Administrator Guide

Moab Accounting Manager®

Version 9.1.0

Administrator Guide : Moab Accounting Manager® ; Version 9.1.0

Table of Contents

List of Tables

Notice

Important: This is a major release of the Moab Accounting Manager Administrator Guide.

Chapter 1. Overview

Moab Accounting Manager is an accounting management system that allows for usage tracking, charge accounting and allocation enforcement for resource usage in technical computing environments. It acts somewhat like a bank in which credits are deposited into funds with constraints designating which entities may access the funds. As resources or services are utilized, funds are charged and usage recorded. It supports familiar operations such as deposits, withdrawals, transfers and refunds. It provides balance and usage feedback to users, managers, and system administrators.

Since the accounting and billing models vary widely from organization to organization, Moab Accounting Manager has been designed to be extremely flexible, featuring customizable usage and fund configurations, and supporting a variety of tracking, charging and allocation models. Attention has been given to scalability, security, and fault tolerance.

Background

Moab Accounting Manager was originally developed as open source software called the Gold Allocation Manager at Pacific Northwest National Laboratory (PNNL) under the Department of Energy (DOE) Scalable Systems Software (SSS) SciDAC project. It has been extended and enhanced by Adaptive Computing Enterprises, Inc. (formerly Cluster Resources, Inc.) and is in production use at many commercial, government and educational sites.

Conceptual Overview

Moab Accounting Manager was designed to be used in technical computing environments for usage tracking, charge accounting and allocation enforcement. Usage tracking involves recording resource usage in customizable usage records. Charge accounting involves calculating and recording charges for usage for invoicing or cost tracking. Allocation enforcement involves establishing limits on the use of system resources by defining separate funds having limited debit or credit balances.

In this overview, we will assume that you want to track or charge for workload resource usage. The use of resources by a job or reservation may result in a usage record. The usage record will track the resources that were used, whom they were used by, and (optionally) how much the usage cost.

With Moab Accounting Manager, it is possible to allocate resource credits to various parties. This is done by associating a cost for the usage by deciding on a currency unit (generically referred to as credits), whether based on a real currency such as dollars, or a reference currency such as billing units or processor seconds. Next you will define charge rates in this currency for the components of your usage (consumable resource costs, multipliers, fees, etc.). Pools of funds called allocations may be created via deposits and can be debit or credit based, finite or infinite, and may be limited to a time frame in which they can be used. These allocations are deposited into logical containers called funds which have constraints that distinguish the conditions under which the funds can be used.

Moab Workload Manager interacts with Moab Accounting Manager to ensure sufficient funds and to track and charge for usage. A typical usage pattern might be as follows. When a job is submitted, a quote is obtained to see how much it will cost and to verify that you have sufficient funds. When it is time for the job to start, a lien (or hold) is placed against your funds for the amount of the requested resources.

When the job ends, the appropriate fund is debited and the lien is removed. A usage record is updated with the charge amount and job usage details. The actual composition of the interactions is very flexible and will be defined by the accounting mode and interaction methods.

Features

- Dynamic Charging -- Rather than post-processing resource usage records on a periodic basis to rectify fund balances, charging can occur incrementally throughout usage or at usage completion.
- Liens -- A hold (called a lien) can be placed against funds for the estimated amount of credits before the usage begins, followed by appropriate charges during and/or at the end of the usage, thereby preventing accounts from using more resources or services than were allocated to them.
- Customizable Usage Records -- Usage record fields can be configured by the site to track custom usage properties.
- Flexible Fund Allocation -- A uniquely flexible design allows resource or service credits to be allocated to arbitrary entities and purposes.
- Expiring Allocations -- Credits may be restricted for use within a designated time period allowing sites to implement a use-it-or-lose-it policy to prevent year-end resource exhaustion and establishing an allocation cycle.
- Flexible Charging -- The billing system can track and charge for composite time-based or non-time-based resource or service usage, and apply flexible charge multipliers and fees.
- Guaranteed Quotes -- Users and resource brokers can determine ahead of time the cost of using resources or services.
- Credit and Debit Allocations -- Allocations feature an optional credit limit allowing support for both debit and credit models. This feature can also be used to enable overdraft protection for specific funds.
- Infinite Allocations -- Deposits can be made with infinite amounts or infinite credit limits when used with a supporting database.
- Powerful Querying -- a powerful querying and update mechanism (based on SQL queries) facilitates flexible reporting and streamlines administrative tasks.
- Nonintrusiveness -- object-level, attribute-level and correlated defaults may be established for arbitrary objects such as users, accounts and organizations. Additionally, these objects may be configured to be automatically created the first time they are seen by the resource management system. These features allow the accounting system to be used with less impact and involvement from users and administrators.
- Consistency -- Moab Accounting Manager has been engineered for robustness, consistency and resiliency. Complex operations are atomic and are automatically rolled back on failure.
- Security -- multiple security mechanisms are supported for strong authentication and encryption.
- Role Based Authorization -- fine-grained (instance-level) Role Based Access Controls are provided for all operations which allows users to view and manipulate only those objects permitted to them.

- Dynamic Customization -- Sites can create or modify record types on the fly enabling them to meet their custom accounting needs. Dynamic object creation allows sites to customize the types of accounting data they collect without modifying the code. This capability turns this system into a generalized information service. This capability is extremely powerful and can be used to manage all varieties of custom configuration data, or to function as a persistence interface for other components.
- Web Interface -- a powerful dynamic web-based GUI is provided for easy remote access for users, managers and administrators which displays only the actions allowed by their role.
- Journaling -- a journaling mechanism preserves the indefinite historical state of all objects and records. This powerful mechanism allows historical bank statements to be generated, provides an undo/redo capability and allows commands to be run as if it were any arbitrary time in the past.
- Event Scheduler -- an event engine can be used to schedule arbitrary MAM commands to run periodically or at a designated time in the future.

Interfaces

Moab Accounting Manager provides a variety of means of interaction, including command-line interfaces, graphical user interfaces, application programming interfaces and communication protocols.

Command-Line Clients

The command-line clients provided feature rich argument sets and built-in documentation. These commands allow scripting and are the preferred way to interact with Moab Accounting Manager for basic usage and administration. Use the --help option for usage information or the --man option for a manual page on any command.

Example 1-1. Listing users using a command-line client

```
mam-list-users
```

Interactive Control Program

The mam-shell command uses a control language to issue object-oriented requests to the server and display the results. The commands may be included directly as command-line arguments or read from stdin. Use the "ShowUsage:=True" option after a valid Object Action combination for usage information on the command.

Example 1-2. Listing users using the mam-shell control program

```
mam-shell User Query
```

Caution

The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. Do not use this command unless you understand the syntax and the potential for unintended results.

Web-based Graphical User Interface

A powerful and easy-to-use web-based GUI permits browser access by users, managers and administrators according to their role definitions.

Example 1-3. Listing users via the web GUI

Click on "Manage Users" -> "List Users"

Perl API

You can access the full functionality via the Perl API. Use perldoc to obtain usage information for the Moab Accounting Manager Perl modules.

Example 1-4. Listing users using the Perl API

```
use MAM;

my $request = new MAM::Request(object => "User", action => "Query");
my $response = $request->getResponse();
foreach my $datum ($response->getData())
{
  print $datum->toString(), "\n";
}
```

SSSRMAP Wire Protocol

It is also possible to interact with Moab Accounting Manager by directly using the SSSRMAP Wire Protocol and Message Format over the network. Documentation for these protocols can be found at *SSS Resource Management and Accounting Protocol Documentation* (http://www.clusterresources.com/products/gold/docs/).

Example 1-5. Listing users via the SSSRMAP wire protocol

```
POST /SSSRMAP HTTP/1.1
Content-Type: text/xml; charset="utf-8"
Transfer-Encoding: chunked

190
<?xml version="1.0" encoding="UTF-8"?>
<Envelope>
 <Body actor="scottmo" chunking="True">
  <Request action="Query" object="User"></Request>
 </Body>
 <Signature>
  <DigestValue>azu4obZswzBt89OgATukBeLyt6Y=</DigestValue>
  <SignatureValue>YXE/C08XX3RX4PMU1bWju+5/E5M=</SignatureValue>
  <SecurityToken type="Symmetric" name="scottmo"></SecurityToken>
 </Signature>
</Envelope>
0
```

Documentation

The documentation for Moab Accounting Manager includes this user's guide, release notes, built-in man pages, module documentation and online documentation.

- Administrator Guide -- The Moab Accounting Manager Administrator guide is a comprehensive manual for users and administrators of Moab Accounting Manager and includes information about features, interfaces, installation, usage, configuration and customization. The Administrator Guide can be found under the $PREFIX/doc directory in pdf and html formats. These documents are also available online.
- Release Notes -- The Release Notes describe the primary features and fixes included in the release, along with notes to aid in migration from previous versions and can be found under the doc directory in the distribution tarball.
- Command Line built-in Man Pages and Usage Synopsis -- All command line clients support a --man option that provides full documentation of the command options and a --help option that provides a brief usage synopsis.
- Module Perl Pod Documentation -- Documentation for Moab Accounting Manager Perl modules can be viewed by changing directory to the $PREFIX/lib directory and running perldoc <modulename>, e.g. perldoc MAM::Request.
- Online Documentation -- The Moab Accounting Manager Administrator Guide can be found online at http://www.adaptivecomputing.com/documentation. The Gold project web page is at

http://www.adaptivecomputing.com/resources/docs/gold/files/index.php and includes the original Gold project documentation.

License

The Moab Accounting Manager software and associated documentation, data and information include parts which are copyrighted by Adaptive Computing Enterprises, Inc., and parts which are copyrighted by Battelle Memorial Institute. The terms and conditions for the use and redistribution of these parts are governed by the Moab Accounting Manager License and the BSD License respectively. Refer to the LICENSE file for details.

Moab Accounting Manager License

The Moab Accounting Manager License specifies the terms and conditions for use and redistribution.

The Moab Accounting Manager License applies to the Moab Accounting Manager software offered by Adaptive Computing Enterprises, Inc. By installing or using this software, Licensee accepts a non-exclusive license from Adaptive Computing Enterprises, Inc. and is bound to accept acknowledgement of and abide by the notices and conditions of the Moab Accounting Manager License.

BSD License

The BSD license specifies the terms and conditions for use and redistribution.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Battelle nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE

Chapter 2. Installation

If you are performing a fresh installation of Moab Accounting Manager, follow the instructions in this chapter. If you are upgrading an existing version of Moab Accounting Manager to a new release, follow the instructions in the Upgrading chapter.

Moab Accounting Manager uses the standard configure, make and make install steps for installation. However, there are a number of preparation, prerequisite, setup and customization steps that need to be performed. This document provides general installation guidance and provides a number of sample steps referenced to a particular installation on a Linux platform using the bash shell. These steps indicate the userid in brackets performing the step. The exact commands to be performed and the user that issues them will vary based on the platform, shell, installation preferences, etc.

In this document, the MAM Server Host refers to the host on which the MAM server is installed. The MAM Database Host refers to the host on which the MAM PostgreSQL database server is installed. This may be the same host as the MAM Server Host or it may be a separate host. The MAM GUI Host refers to the host on which the HTTP server that serves the optional MAM GUI is installed. When installing the MAM GUI, this may be the same host as the MAM Server Host or it may be a separate host. MAM Client Hosts refer to those hosts on which the MAM clients are installed. The MAM Server Host will also be a MAM Client Host, along with any other hosts on which you want to have the MAM client commands available to users or administrators.

Install Prerequisites

Before installing Moab Accounting Manager, you will need to satisfy the following prerequisites:

Open the necessary ports in your firewall

If your site is running firewall software on its server hosts, you will need to configure the firewall to allow connections to the necessary ports.

On the MAM Server Host, if the Moab server or the MAM clients are installed on a separate host from the MAM server, open the MAM server port (defaults to 7112) in the firewall.

For RedHat-6-based or Fedora systems using iptables:

```
[root]#  iptables-save > /tmp/iptables.mod
```

```
[root]#  vi /tmp/iptables.mod
```

```
# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 7112 -j ACCEPT
```

```
[root]#  iptables-restore < /tmp/iptables.mod
```

```
[root]#  service iptables save
```

For RedHat-7-based or Fedora systems using firewalld:

```
[root]#  firewall-cmd --add-port=7112/tcp --permanent
[root]#  firewall-cmd --reload
```

For SUSE-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2
```

```
FW_SERVICES_EXT_TCP="7112"
```

```
[root]#  service SuSEfirewall2_setup restart
```

On the MAM Database Host, if the MAM Postgresql server is on a separate host from the MAM server, open the postgres port in the firewall.

For RedHat-6-based or Fedora systems using iptables:

```
[root]#  iptables-save > /tmp/iptables.mod
[root]#  vi /tmp/iptables.mod
```

```
# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 5432 -j ACCEPT
```

```
[root]#  iptables-restore < /tmp/iptables.mod
[root]#  service iptables save
```

For RedHat-7-based or Fedora systems using firewalld:

```
[root]#  firewall-cmd --add-port=postgres/tcp --permanent
[root]#  firewall-cmd --reload
```

For SUSE-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2
```

```
FW_SERVICES_EXT_TCP="5432"
```

```
[root]#  service SuSEfirewall2_setup restart
```

On the MAM GUI Host, if using the MAM GUI, open the https port in the firewall for secure browser communication.

For RedHat-6-based or Fedora systems using iptables:

```
[root]#  iptables-save > /tmp/iptables.mod
[root]#  vi /tmp/iptables.mod
```

```
# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 443 -j ACCEPT
```

```
[root]#  iptables-restore < /tmp/iptables.mod
[root]#  service iptables save
```

For RedHat-7-based or Fedora systems using firewalld:

```
[root]#  firewall-cmd --add-port=https/tcp --permanent
[root]#  firewall-cmd --reload
```

For SUSE 11-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2
```

```
FW_SERVICES_EXT_TCP="443"
```

```
[root]#  service SuSEfirewall2_setup restart
```

For SUSE 12-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2
```

```
FW_SERVICES_EXT_TCP="443"
```

```
[root]#  systemctl restart SuSEfirewall2.service
```

On the MAM Web Services Host, if using MAM Web Services, open the https port in the firewall.

For RedHat-6-based or Fedora systems using iptables:

```
[root]#  iptables-save > /tmp/iptables.mod
[root]#  vi /tmp/iptables.mod
```

```
# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 443 -j ACCEPT
```

```
[root]#  iptables-restore < /tmp/iptables.mod
[root]#  service iptables save
```

For RedHat-7-based or Fedora systems using firewalld:

```
[root]#  firewall-cmd --add-port=https/tcp --permanent
[root]#  firewall-cmd --reload
```

For SUSE 11-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2
```

```
FW_SERVICES_EXT_TCP="443"
```

```
[root]#  service SuSEfirewall2_setup restart
```

For SUSE 12-based systems using SuSEfirewall2:

```
[root]#  vi /etc/sysconfig/SuSEfirewall2

FW_SERVICES_EXT_TCP="443"


[root]#  systemctl restart SuSEfirewall2.service
```

Enable extra repositories

Some of the required Perl module dependencies listed in subsequent sections may require you to obtain access to packages that are found in the optional add-on repositories for the distribution.

On the MAM Server Host, the MAM Database Host, the MAM GUI Host and the MAM Client Hosts, enable the appropriate extra repositories.

For RedHat-based systems:

Enable the Extra Packages for Enterprise Linux (EPEL) repository.

```
[root]#  yum install epel-release
```

> **Note:** On RHEL systems, your system must be registered to Red Hat Subscription Management. Additionally, some packages are only available in the EPEL repository so we recommend installing and disabling the repository so that we can pull specific rpms from the repository but avoid subsequent general system package updates from upgrading from EPEL.
>
> *For RHEL 6:*
>
> ```
> [root]#  rpm -Uvh
> http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
>
> [root]#  yum install yum-utils
>
> [root]#  yum-config-manager --disable epel
> ```
>
> *For RHEL 7:*
>
> ```
> [root]#  rpm -Uvh
> http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-8.noarch.rpm
>
> [root]#  yum install yum-utils
>
> [root]#  yum-config-manager --disable epel
> ```

For SUSE-based systems:

Enable the devel:languages:perl repository.

```
[root]#  zypper addrepo
http://download.opensuse.org/repositories/devel:languages:perl/SLE_12/devel:languages:perl.

[root]#  zypper refresh
```

C Compiler and LSB Tools [REQUIRED]

A C compiler is required in the configure step and to compile the mamauth promotion method if designated. Additionally, lsb-release is needed to be able to perform distribution-specific configuration.

On the MAM Server Host, the MAM GUI Host and the MAM Client Hosts, install the C Compiler and LSB Tools.

For Fedora 22 or higher systems:

```
[root]#  dnf install gcc redhat-lsb-core
```

For RedHat-based systems:

```
[root]#  yum install gcc redhat-lsb-core
```

For SUSE-based systems:

```
[root]#  zypper install gcc lsb-release
```

For Debian-based systems:

```
[root]#  apt-get install gcc make
```

Perl [REQUIRED]

The Moab Accounting Manager server and clients are written in Perl. Perl 5.8 or higher is required. It is usually at a sufficient level at most modern operating systems. Use 'perl -v' to see what level of Perl is installed.

On the MAM Server Host, the MAM GUI Host and the MAM Client Hosts, install Perl.

For Fedora 22 or higher systems:

```
[root]#  yum install perl perl-open
```

For other RedHat-based systems:

```
[root]#  yum install perl
```

Suidperl [OPTIONAL]

Command-line clients and Perl API scripts use a security promotion method (mamauth or suidperl) to authenticate and encrypt communications with the server. It is recommended that you install and use setuid perl as the security promotion method if it is available for your system. Otherwise configure will compile and use mamauth as the security promotion method. Use 'suidperl -v' to see if suidperl is installed. See the description for the security.promotion configuration parameter in the Client Configuration section for more information about the two security promotion methods.

On the MAM Client Hosts, install Suidperl if applicable.

For RedHat-6-based systems:

```
[root]#  yum install perl-suidperl
```

For SUSE-11-based systems:

```
[root]#  chmod 4755 /usr/bin/sperl*
```

Note: Systems with perl 5.12 or higher do not support suidperl. These systems will need to use the mamauth security promotion method.

OpenSSL [REQUIRED]

OpenSSL is used to encode the secret key and is used in the web interface to encrypt communications with the server. OpenSSL is normally preinstalled on most modern operating systems. The development package is needed if you will be using the mamauth security promotion method.

On the MAM Client Hosts, install the OpenSSL development package if applicable.

For Fedora 22 or higher systems:

```
[root]#  dnf install openssl-devel
```

For RedHat-7-based or Fedora 21 or lower systems:

```
[root]#  yum install openssl-devel
```

For SUSE-12-based systems:

```
[root]#  zypper install libopenssl-devel
```

For Debian-based systems:

```
[root]#  apt-get install libssl-dev
```

PostgreSQL Database Server [REQUIRED]

You will need to install the PostgreSQL database that will be used by Moab Accounting Manager to store current and historical accounting information. It is permissible to have the PostgreSQL server installed on a different host from Moab Accounting Manager, however, the PRODUCT; server host will need to have the postgreSQL libraries and the Perl Pg DBD installed locally.

On the MAM Database Host, install the PostgreSQL server packages.

For Fedora 22 or higher systems:

```
[root]#  dnf install postgresql-server
```

For Redhat-based or Fedora 21 or lower systems:

```
[root]#  yum install postgresql-server
```

For SUSE-11-based systems:

```
[root]#  zypper install postgresql-server
```

For SUSE-12-based systems:

```
[root]#  zypper install postgresql-server
```

For Debian-based systems:

```
[root]#  apt-get install postgresql
```

On the MAM Server Host, install the PostgreSQL client and library packages.

For Fedora 22 or higher systems:

```
[root]#  dnf install postgresql postgresql-libs perl-DBD-Pg
```

For Redhat-based or Fedora 21 or lower systems:

```
[root]#  yum install postgresql postgresql-libs perl-DBD-Pg
```

For SUSE-11-based systems:

```
[root]#  zypper install postgresql postgresql-libs perl-DBD-Pg
```

For SUSE-12-based systems:

```
[root]#  zypper install postgresql postgresql-devel libpq5 perl-DBD-Pg
```

> **Note:** perl-DBD-Pg is provided in the SLES 12 devel:languages:perl repository, but for some reason this is not included in the SLES 12 SP1 devel:languages:perl. So if you are installing on SLES 12 SP1, you may either obtain it from the SLES 12 devel:languages:perl repository, install the perl module from CPAN, or install the rpm from a reputable third-party rpm repository.
>
> ```
> [root]#  zypper addrepo
> http://download.opensuse.org/repositories/devel:/languages:/perl/SLE_12
> devel_languages_perl_SLE_12
> ```
>
> ```
> [root]#  zypper install perl-DBD-Pg
> ```
>
> ```
> [root]#  zypper modifyrepo --disable devel_languages_perl_SLE_12
> ```
>
> —OR—
>
> ```
> [root]#  cpan DBD::Pg
> ```

For Debian-based systems:

```
[root]#  apt-get install postgresql-common postgresql-client libdbd-pg-perl
```

Apache Httpd Server with mod_ssl for MAM GUI [OPTIONAL]

Moab Accounting Manager provides a web-based gui so that managers, users and administrators can interact with the accounting and allocation system. The web interface utilizes Perl CGI and SSL and needs to have an httpd server (preferably apache) installed. mod_ssl is also needed and is often bundled as part of the apache2 server.

On the MAM GUI Host, install the Apache Httpd Server with mod_ssl if you will be using the MAM GUI.

For Fedora 22 or higher systems:

```
[root]#  dnf install httpd mod_ssl
```

For RedHat-based or Fedora 21 or lower systems:

```
[root]#  yum install httpd mod_ssl
```

For SUSE-based systems:

```
[root]#  zypper install apache2
```

For Debian-based systems:

```
[root]#  apt-get install apache2
```

Apache Httpd Server with mod_perl for MAM Web Services [OPTIONAL]

Moab Accounting Manager provides a REST-like web services interface to the Moab Accounting Manager API. The implementation utilizes an apache httpd server with mod_perl and SSL.

On the MAM Web Services Host, install the Apache Httpd Server with mod_ssl and mod_perl if you will be using MAM Web Services.

For Fedora 22 or higher systems:

```
[root]#  dnf install httpd mod_ssl mod_perl
```

For RedHat-based or Fedora 21 or lower systems:

```
[root]#  yum install httpd mod_ssl mod_perl
```

> **Note:** On RHEL systems, mod_perl may be obtained from the EPEL repository.
>
> ```
> [root]#  yum install --enablerepo=epel mod_perl
> ```

For SUSE-based systems:

```
[root]#  zypper install apache2 apache2-mod_perl
```

For Debian-based systems:

```
[root]#  apt-get install apache2 libapache2-mod-perl2
```

Perl Module Dependencies [REQUIRED]

Moab Accounting Manager requires the use of a number of Perl modules. These modules may be installed by the vendor package manager. Alternatively, the required modules can be installed from CPAN (not shown).

On the MAM Server Host, the MAM GUI Host, the MAM Web Services Host and the MAM Client Hosts, install the common Perl modules.

For Fedora 22 or higher systems:

```
[root]#  dnf install perl-Config-Tiny perl-Crypt-CBC perl-Crypt-DES
perl-Crypt-DES_EDE3 perl-Digest-HMAC perl-Error perl-JSON
perl-Log-Dispatch-FileRotate perl-Log-Log4perl perl-XML-LibXML
```

For RedHat-based or Fedora 21 or lower systems:

```
[root]#  yum install perl-Config-Tiny perl-Crypt-CBC perl-Crypt-DES
perl-Crypt-DES_EDE3 perl-Digest-HMAC perl-Digest-SHA perl-Error perl-JSON
perl-Log-Dispatch-FileRotate perl-Log-Log4perl perl-XML-LibXML
```

Note: On RHEL systems, some packages must be obtained from optional repositories.

```
[root]#  yum install --enablerepo=epel,rhel-7-server-optional-rpms perl-Config-Tiny
perl-Crypt-CBC perl-Crypt-DES perl-Crypt-DES_EDE3 perl-Digest-HMAC perl-Digest-SHA
perl-Error perl-JSON perl-Log-Dispatch-FileRotate perl-Log-Log4perl
perl-XML-LibXML
```

For SUSE-based systems:

```
[root]#  zypper install perl-Config-Tiny perl-Crypt-CBC perl-Crypt-DES
perl-Crypt-DES_EDE3 perl-Digest-HMAC perl-Error perl-JSON
perl-Log-Dispatch-FileRotate perl-Log-Log4perl perl-XML-LibXML
```

For Debian-based systems:

```
[root]#  apt-get install libconfig-tiny-perl libcrypt-cbc-perl
libcrypt-des-perl libcrypt-des-ede3-perl libdigest-hmac-perl
libdigest-sha-perl liberror-perl libjson-perl liblog-dispatch-filerotate-perl
liblog-log4perl-perl libxml-libxml-perl
```

Note: The Compress::Zlib package (libcompress-zlib-perl) is required for Ubuntu 12 and earlier.

On the MAM Server Host, install the Perl modules specific to the MAM Server.

For Fedora 22 or higher systems:

```
[root]#  dnf install perl-Date-Manip perl-Time-HiRes perl-DBI
```

For RedHat-based or Fedora 21 or lower systems:

```
[root]#  yum install perl-Date-Manip perl-Time-HiRes perl-DBI
```

For SUSE-based systems:

```
[root]#  zypper install perl-Date-Manip perl-DBI
```

For Debian-based systems:

```
[root]#  apt-get install libdate-manip-perl libdbi-perl
```

On the MAM GUI Host, install the Perl modules specific to the MAM GUI if you are going to be using it.

For Fedora 22 or higher systems:

```
[root]#  dnf install perl-CGI perl-CGI-Session
```

For RedHat-based or Fedora 21 or lower systems:

```
[root]#  yum install perl-CGI perl-CGI-Session
```

For SUSE-based systems:

```
[root]#  zypper install perl-CGI perl-CGI-Session
```

For Debian-based systems:

```
[root]#  apt-get install libcgi-session-perl
```

On the MAM Client Hosts (including the MAM Server Host), install the Perl modules specific to the clients.

For Fedora 22 or higher systems:

```
[root]#  dnf install perl-Term-ReadLine-Gnu perl-TermReadKey perl-XML-LibXML
```

For RedHat-7.0-based systems:

```
[root]#  yum install cpan readline-devel ncurses-devel perl-TermReadKey

[root]#  cpan Term::ReadLine::Gnu
```

For RedHat-6-based, RedHat-7.1-or-higher or older Fedora systems:

```
[root]#  yum install perl-Term-ReadLine-Gnu perl-TermReadKey perl-XML-LibXML
```

For SUSE-based systems:

```
[root]#  zypper install perl-TermReadLine-Gnu perl-TermReadKey
```

For Debian-based systems:

```
[root]#  apt-get install libterm-readline-gnu-perl libterm-readkey-perl
```

> **Note:** If any of the Perl module packages fail to install or are unavailable for your system, you can install it from CPAN by running 'cpan MODULENAME', where MODULENAME is the respective perl module name.

Preparation

To build and install Moab Accounting Manager, you first need to unpack the tar archive and change directory into the top directory of the distribution. For maximum security, it is recommended that you install and run Moab Accounting Manager under its own non-root userid. This user will be referred to as the accounting admin user.

```
[root]#  useradd -m autobuild

[root]#  su - autobuild

[autobuild]$  mkdir src

[autobuild]$  cd src

[autobuild]$  tar -zxvf mam-9.1.0.tar.gz

[autobuild]$  cd mam-9.1.0
```

Configuration

To configure Moab Accounting Manager, run the "configure" script provided with the distribution.

The following is a list of commonly used configure options:

- -h, --help display the list of options

  Run ./configure --help to see the list of configure options.

- --prefix=PREFIX install architecture-independent files in PREFIX [/opt/mam]

  Base installation directory where all subdirectories will be installed unless otherwise designated (defaults to /opt/mam).

- --localstatedir=DIR modifiable single-machine data [PREFIX]

  Home directory where per-configuration subdirectories (such as etc, log, data) will be installed (defaults to PREFIX).

- --with-db-name=NAME database name [mam]

  Name of the SQL database that the server will sync with (defaults to mam).

- --with-user=USER accounting admin user id under which the server will run [mam]

  Use --with-user to specify the accounting admin userid that the server will run under and who will have full administrative privileges (defaults to mam). It is recommended that this be a non-privileged user for the highest security.

- --with-promotion=mamauth|suidperl method used to promote privileges to read the site configuration file

  Command-line clients and scripts using the API need to use a security promotion method to authenticate and encrypt the communication using the symmetric key. The default is suidperl if it is installed on the system, otherwise the default is mamauth. See the description for the security.promotion configuration parameter in the Client Configuration section for more information about the two security promotion methods.

- --with-cgi-bin=DIR directory to install cgi-bin files if using web gui [/var/www/cgi-bin/mam]

  If you will intend to use the web GUI, use with-cgi-bin to specify the directory where you want the Moab Accounting Manager CGI files to reside (defaults to /var/www/cgi-bin/mam).

- --with[out]-gui specifies whether or not to install the CGI GUI

  If you do not intend to use the CGI web GUI, you can specify --without-gui to not install the CGI scripts, otherwise the default is to install the GUI CGI scripts.

- --without-init don't install mam init.d service

  If you do not intend to use the mam init.d service, you can specify --without-init to not install the mam init.d script, otherwise the default is to install the mam init.d script.

- --without-profile don't install mam profile.d scripts

  If you do not intend to use the mam profile.d environment scripts, you can specify --without-profile to not install the mam profile.d scripts, otherwise the default is to install the mam profile.d scripts.

- --with-legacy-links install the legacy links

  Creates symbolic links allowing the use of the old client and server command names. When running a command under its old name, the command will issue a deprecation warning. This warning can be disabled by setting client.deprecationwarning = false in the mam-client.conf file. The default is not to install the legacy links.

To assume the defaults, use:

```
[autobuild]$  ./configure
```

Compilation

To compile the program, type make.

```
[autobuild]$  make
```

> **Note:** If you only need to install the clients on a particular system, you would instead type make clients-only. If you only need to install the web GUI on a particular system, you would instead type make gui-only. If you only need to install the web services on a particular system, replace make with make ws-only.

Installation

Run ‘make install’ as the root user to install Moab Accounting Manager.

```
[autobuild]$  exit
[root]#  cd ~autobuild/src/mam-9.1.0
[root]#  make install
```

> **Note:** If you only need to install the clients on a particular system, you would instead type make install-clients-only. If you only need to install the web GUI on a particular system, you would instead type make install-gui-only. If you only need to install the web services on a particular system, replace make with make install-ws-only.

To delete the files created by the product installation, you can use ’make uninstall’.

Database Setup

You will need to define a database user, create the database, and configure the database server to support transactions and connections from the MAM Server Host.

Initialize the database

On the MAM Database Host, the database must be initialized before it can be configured.

For RedHat-6-based systems:

```
[root]#  service postgresql initdb
```

For RedHat-7-based systems:

```
[root]#  postgresql-setup initdb
```

For Fedora systems:

```
[root]#  postgresql-setup --initdb --unit postgresql
```

For SUSE 11-based and Debian-based systems:

```
[root]#  service postgresql start
```

For SUSE 12-based systems:

```
[root]#  systemctl start postgresql.service
```

Configure trusted connections

On the MAM Database Host, set the host-based client authentication as appropriate. Edit or add a line in the pg_hba.conf file for the interface from which the Moab Accounting Manager server will be connecting to the database.

For RedHat-based and SUSE-based systems:

```
[root]#  vi /var/lib/pgsql/data/pg_hba.conf
```

```
# Replace 127.0.0.1 with the IP address of the MAM Server Host if the
# MAM PostgreSQL server is on a separate host from the MAM server.
host    all             all             127.0.0.1/32            md5
host    all             all             ::1/128                 md5
```

If the MAM PostgreSQL server is on a separate host from the MAM server, you will additionally need to configure PostgreSQL to accept connections from the MAM Server Host.

For RedHat-based and SUSE-based systems:

```
[root]#  vi /var/lib/pgsql/data/postgresql.conf
```

```
# Replace <MAM-Host> with the interface name from which the MAM server
# will be connecting to the database.
listen_addresses = '<MAM-Host>'
```

For Debian-based systems:

```
[root]#  vi /etc/postgresql/9.*/main/postgresql.conf
```

```
# Replace <MAM-Host> with the interface name from which the MAM server
# will be connecting to the database.
listen_addresses = '<MAM-Host>'
```

Start the database

On the MAM Database Host, configure the database to start on system startup. Start (or restart) the database server with the new configurations in effect.

For RedHat-6-based and SUSE-11-based systems:

```
[root]#  chkconfig postgresql on
[root]#  service postgresql restart
```

For RedHat-7-based, SUSE-12-based or Fedora systems:

```
[root]#  systemctl enable postgresql.service
[root]#  systemctl restart postgresql.service
```

For Debian-based systems:

```
[root]#  service postgresql restart
```

Create the database

On the MAM Database Host, create the Moab Accounting Manager database and add the accounting admin user as a database administrator. This must be performed as the postgres user.

```
[root]#  su - postgres
[postgres]$  psql

create database "mam";
create user autobuild with password 'changeme!';
\q


[postgres]$  exit
```

Bootstrap

On the MAM Server Host, use hpc.sql to populate the Moab Accounting Manager database with an sql dump that defines the objects, actions and attributes necessary to function as an Accounting Manager in an HPC context.

```
[root]#  su - autobuild
[autobuild]$  cd src/mam-9.1.0
[autobuild]$  psql mam < hpc.sql
[autobuild]$  exit
```

General Setup

On the MAM Server Host, edit the configuration files as necessary. At a minimum, you should set your database user and password to match the values you selected during the database setup. Most of the other defaults should be sufficient to get you up and running. See the Configuration files chapter for information on the configuration files and parameters.

```
[autobuild]$  vi /opt/mam/etc/mam-server.conf

database.user     = autobuild
database.password = changeme!


[autobuild]$  vi /opt/mam/etc/mam-client.conf
```

Startup

On the MAM Server Host, configure your system to automatically start the MAM server at system startup. Then start the server.

For RedHat-6-based or SUSE-11-based systems:

```
[root]#  chkconfig --add mam

[root]#  service mam start
```

For RedHat-7-based or SUSE-12-based systems:

```
[root]#  systemctl enable mam.service

[root]#  systemctl start mam.service
```

For Debian-based systems:

```
[root]#  update-rc.d mam defaults 95

[root]#  service mam start
```

Web Server Setup

If you want to use the web GUI, you will need to configure your Apache HTTP server to use SSL. The following shows some sample steps to configure the web GUI. The actual steps you will need to use will vary according to your distribution and environment. The web server configuration must be modified to support the invocation of cgi-bin scripts over an SSL connection using a private key and a signed certificate. You may also need to work around the default selinux restrictions if it is enabled on your system.

On the MAM GUI Host, configure the HTTP server.

Configure an apache virtual host to use CGI over SSL

For RedHat-based systems:

[root]# vi /etc/httpd/conf.d/ssl.conf

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  # Configure your cgi-bin directory
  <Directory "/var/www/cgi-bin">
    Options ExecCGI
    AddHandler cgi-script .cgi
    AllowOverride All
    Order allow,deny
    Allow from all
  </Directory>

  # Create an alias for /cgi-bin pointing to your cgi-bin directory.
  # If you chose to install to a cgi-bin subdir, you may want to create an
  # alias for that as well. Comment out any related ScriptAlias entries.
  Alias /cgi-bin/ /var/www/cgi-bin/
  Alias /mam /var/www/cgi-bin/mam/

  # Add index.cgi to the DirectoryIndex so you can use the shorter subdir name
  DirectoryIndex index.cgi

...
</VirtualHost>
```

For SUSE-11-based systems:

[root]# a2enmod perl

[root]# a2enflag SSL

[root]# cp /etc/apache2/vhosts.d/vhost-ssl.template /etc/apache2/vhosts.d/mam-ssl.conf

[root]# vi /etc/apache2/vhosts.d/mam-ssl.conf

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  # Configure your cgi-bin directory
  <Directory "/srv/www/cgi-bin">
    Options ExecCGI
    AddHandler cgi-script .cgi
    AllowOverride All
    Order allow,deny
    Allow from all
```

```
  </Directory>

  # Create an alias for /cgi-bin pointing to your cgi-bin directory.
  # If you chose to install to a cgi-bin subdir, you may want to create an
  # alias for that as well. Comment out any related ScriptAlias entries.
  Alias /cgi-bin/ /srv/www/cgi-bin/
  Alias /mam /srv/www/cgi-bin/mam/

  # Add index.cgi to the DirectoryIndex so you can use the shorter subdir name
  DirectoryIndex index.cgi

...
</VirtualHost>
```

For SUSE-12-based systems:

[root]# a2enflag SSL

[root]# cp /etc/apache2/vhosts.d/vhost-ssl.template /etc/apache2/vhosts.d/mam-ssl.conf

[root]# vi /etc/apache2/vhosts.d/mam-ssl.conf

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  # Configure your cgi-bin directory
  <Directory "/srv/www/cgi-bin">
    Options ExecCGI
    AddHandler cgi-script .cgi
    AllowOverride All
    Require all granted
  </Directory>

  # Create an alias for /cgi-bin pointing to your cgi-bin directory.
  # If you chose to install to a cgi-bin subdir, you may want to create an
  # alias for that as well. Comment out any related ScriptAlias entries.
  Alias /cgi-bin/ /srv/www/cgi-bin/
  Alias /mam /srv/www/cgi-bin/mam/

  # Add index.cgi to the DirectoryIndex so you can use the shorter subdir name
  DirectoryIndex index.cgi

...
</VirtualHost>
```

For Debian-based systems with apache version less than 2.4:

[root]# a2enmod ssl

[root]# a2ensite default-ssl

[root]# vi /etc/apache2/sites-enabled/default-ssl

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  # Configure your cgi-bin directory
  <Directory "/usr/lib/cgi-bin">
    Options ExecCGI
    AddHandler cgi-script .cgi
    AllowOverride All
    Order allow,deny
    Allow from all
  </Directory>

  # Create an alias for /cgi-bin pointing to your cgi-bin directory.
  # If you chose to install to a cgi-bin subdir, you may want to create an
  # alias for that as well. Comment out any related ScriptAlias entries.
  Alias /cgi-bin/ /usr/lib/cgi-bin/
  Alias /mam /usr/lib/cgi-bin/mam/

  # Add index.cgi to the DirectoryIndex so you can use the shorter subdir name
  DirectoryIndex index.cgi

...
</VirtualHost>
```

For Debian-based systems with apache version 2.4 or higher:

[root]# a2enmod cgi

[root]# a2enmod ssl

[root]# a2ensite default-ssl

[root]# vi /etc/apache2/sites-enabled/default-ssl.conf

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  # Configure your cgi-bin directory
  <Directory "/usr/lib/cgi-bin">
    Options ExecCGI
    AddHandler cgi-script .cgi
    AllowOverride All
    Require all granted
  </Directory>

  # Create an alias for /cgi-bin pointing to your cgi-bin directory.
  # If you chose to install to a cgi-bin subdir, you may want to create an
  # alias for that as well. Comment out any related ScriptAlias entries.
  Alias /cgi-bin/ /usr/lib/cgi-bin/
```

```
  Alias /mam /usr/lib/cgi-bin/mam/

  # Add index.cgi to the DirectoryIndex so you can use the shorter subdir name
  DirectoryIndex index.cgi

...
</VirtualHost>
```

Adjust SELinux

For RedHat-based systems, if Security Enhanced Linux (SELinux) is enforced on your system, it will need to be adjusted to allow the web server to make network connections, use setuid for authentication, and to write to the log file.

On the MAM Server Host, the MAM GUI Host and the MAM Client Hosts, adjust SELinux if enabled.

Determine the current mode of SELinux by running 'getenforce'.

```
[root]#  getenforce

Enforcing
```

If the command returns a mode of "Disabled" or "Permissive", or if the getenforce command is not found, you do not need to do anything and you can skip the rest of this sub-section.

If the command returns a mode of "Enforcing", you may choose between the options of customizing SELinux to allow the web GUI to perform its required functions or disabling SELinux on your system.

If you wish to customize SELinux, you should understand that selinux can vary with version and architecture and these instructions may not suffice for all possible environments. Ultimately, it will be your responsibility to determine and apply the necessary adjustments for the web GUI to work under SELinux. The following example is offered as a set of steps that appeared to work acceptably in our testing for supported RedHat-based systems.

For RedHat-6-based systems:

```
[root]#  cat > mamgui.te <<EOF

module mamgui 1.0;
require {
      type httpd_sys_script_t;
      type port_t;
      class capability setuid;
      class tcp_socket name_connect;
}
allow httpd_sys_script_t port_t:tcp_socket name_connect;
allow httpd_sys_script_t self:capability setuid;
EOF
```

```
[root]#  checkmodule -M -m -o mamgui.mod mamgui.te
[root]#  semodule_package -m mamgui.mod -o mamgui.pp
[root]#  semodule -i mamgui.pp
[root]#  setenforce 0
[root]#  chcon -v -t httpd_sys_content_t /opt/mam/log
[root]#  setenforce 1
```

For Fedora 24 or higher systems:

```
[root]#  yum install checkpolicy policycoreutils-python
policycoreutils-python-utils
```

For RedHat-7-based or Fedora 23 or lower systems:

```
[root]#  yum install checkpolicy policycoreutils-python
[root]#  cat > mamgui.te <<EOF
module mamgui 1.0;
require {
        type httpd_sys_script_t;
        type unreserved_port_t;
        class tcp_socket name_connect;
}
allow httpd_sys_script_t unreserved_port_t:tcp_socket name_connect;
EOF
```

```
[root]#  checkmodule -M -m -o mamgui.mod mamgui.te
[root]#  semodule_package -m mamgui.mod -o mamgui.pp
[root]#  semodule -i mamgui.pp
[root]#  setenforce 0
[root]#  chcon -v -t httpd_sys_rw_content_t /opt/mam/log
[root]#  setenforce 1
```

> **Note:** If you used the --prefix configure option when configuring Moab Accounting Manager, substitute the PREFIX you used for references to /opt/mam in the chcon command.

Alternatively, if you instead choose to disable SELinux, the following steps can be followed.

```
[root]#  vi /etc/sysconfig/selinux
SELINUX=disabled
[root]#  setenforce 0
```

Install a Signed Certificate

For the highest security, it is recommended that you install a public key certificate that has been signed by a certificate authority. The exact steps to do this will be specific to your distribution and the chosen certificate authority. An overview of this process for CentOS 5 is documented at http://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html. Alternatively, if your network domain can be secured from man-in-the-middle attacks, you could use a self-signed certificate. Often this does not require any additional steps since in many distributions (e.g. RedHat-based), the Apache SSL configuration provides self-signed certificates by default.

The following steps assume you are using self-signed certificates:

Create self-signed SSL certificate and key files. Some distributions (e.g. RedHat) ship with ready-made certificates.

For SUSE-based systems:

```
[root]#  cd /etc/apache2
[root]#  openssl genrsa -out ssl.key/server.key 1024
[root]#  openssl req -new -key ssl.key/server.key -x509 -out
ssl.crt/server.crt
```

Start the HTTP Server

Start or restart the Apache HTTP server.

For RedHat-6-based systems:

```
[root]#  chkconfig httpd on
[root]#  service httpd restart
```

For RedHat-7-based or Fedora systems:

```
[root]#  systemctl enable httpd.service
[root]#  systemctl restart httpd.service
```

For SUSE-11-based systems:

```
[root]#  chkconfig apache2 on
[root]#  service apache2 restart
```

For SUSE-12-based systems:

```
[root]#  systemctl enable apache2.service
[root]#  systemctl restart apache2.service
```

For Debian-based systems:

```
[root]#  update-rc.d apache2 defaults
[root]#  service apache2 restart
```

Web Services Setup

If you want to use MAM Web Services, you will need to configure your Apache HTTP server to use mod_perl and SSL. The following shows some sample steps to configure web services. The actual steps you will need to use will vary according to your distribution and environment. The web server configuration must be modified to add the mamws location within an SSL connection using a basic authentication and a signed certificate. You may also need to work around the default selinux restrictions if it is enabled on your system.

On the MAM Web Services Host, configure the HTTP server.

Configure an apache virtual host to use mod_perl over SSL

For RedHat-based systems:

```
[root]#  vi /etc/httpd/conf.d/ssl.conf
```

Add or edit the SSL virtual host definition to include the mamws location as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  PerlOptions +Parent
  PerlSwitches -Mlib=/opt/mam/lib
  PerlModule MAM::WSResponseHandler
  PerlModule MAM::WSAuthenHandler
  <Location /mamws>
    SetHandler perl-script
    PerlResponseHandler MAM::WSResponseHandler
    Options +ExecCGI

    AuthName MAM
    PerlAuthenHandler MAM::WSAuthenHandler
    Require valid-user

    Order allow,deny
    Allow from all
  </Location>

...
</VirtualHost>
```

For SUSE-11-based systems:

```
[root]#  a2enflag SSL

[root]#  cp /etc/apache2/vhosts.d/vhost-ssl.template
/etc/apache2/vhosts.d/mam-ssl.conf

[root]#  vi /etc/apache2/vhosts.d/mam-ssl.conf
```

Add or edit the SSL virtual host definition to include the mamws location as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  PerlOptions +Parent
  PerlSwitches -Mlib=/opt/mam/lib
  PerlModule MAM::WSResponseHandler
  PerlModule MAM::WSAuthenHandler
  <Location /mamws>
    SetHandler perl-script
    PerlResponseHandler MAM::WSResponseHandler
    Options +ExecCGI

    AuthName MAM
    PerlAuthenHandler MAM::WSAuthenHandler
    Require valid-user

    Order allow,deny
    Allow from all
  </Location>

...
</VirtualHost>
```

For SUSE-12-based systems:

[root]# a2enflag SSL

[root]# cp /etc/apache2/vhosts.d/vhost-ssl.template /etc/apache2/vhosts.d/mam-ssl.conf

[root]# vi /etc/apache2/vhosts.d/mam-ssl.conf

Adjust the SSLCertificate entries if necessary

```
SSLCertificateFile /etc/apache2/ssl.crt/server.crt
SSLCertificateKeyFile /etc/apache2/ssl.key/server.key
```

Add or edit the SSL virtual host definition to include the mamws location as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  PerlOptions +Parent
  PerlSwitches -Mlib=/opt/mam/lib
  PerlModule MAM::WSResponseHandler
  PerlModule MAM::WSAuthenHandler
  <Location /mamws>
```

```
    SetHandler perl-script
    PerlResponseHandler MAM::WSResponseHandler
    Options +ExecCGI

    AuthName MAM
    PerlAuthenHandler MAM::WSAuthenHandler
    Require valid-user

    AllowOverride All
    Require all granted
  </Location>

...
</VirtualHost>
```

For Debian-based systems with apache version less than 2.4:

```
[root]#  a2enmod ssl
```

```
[root]#  a2ensite default-ssl
```

```
[root]#  vi /etc/apache2/sites-enabled/default-ssl
```

Add or edit the SSL virtual host definition to include the mamws location as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  PerlOptions +Parent
  PerlSwitches -Mlib=/opt/mam/lib
  PerlModule MAM::WSResponseHandler
  PerlModule MAM::WSAuthenHandler
  <Location /mamws>
    SetHandler perl-script
    PerlResponseHandler MAM::WSResponseHandler
    Options +ExecCGI

    AuthName MAM
    PerlAuthenHandler MAM::WSAuthenHandler
    Require valid-user

    Order allow,deny
    Allow from all
  </Location>

...
</VirtualHost>
```

For Debian-based systems with apache version 2.4 or higher:

```
[root]#  a2enmod cgi
```

```
[root]#  a2enmod ssl

[root]#  a2ensite default-ssl

[root]#  vi /etc/apache2/sites-enabled/default-ssl.conf
```

Add or edit the SSL virtual host definition as appropriate for your environment.

```
<VirtualHost _default_:443>
...

  PerlOptions +Parent
  PerlSwitches -Mlib=/opt/mam/lib
  PerlModule MAM::WSResponseHandler
  PerlModule MAM::WSAuthenHandler
  <Location /mamws>
    SetHandler perl-script
    PerlResponseHandler MAM::WSResponseHandler
    Options +ExecCGI

    AuthName MAM
    PerlAuthenHandler MAM::WSAuthenHandler
    Require valid-user

    AllowOverride All
    Require all granted
  </Location>

...
</VirtualHost>
```

Adjust SELinux

For RedHat-based systems, if Security Enhanced Linux (SELinux) is enforced on your system, it will need to be adjusted to allow the web server to make network connections, use setuid for authentication, and to write to the log file.

On the MAM Server Host, the MAM GUI Host and the MAM Client Hosts, adjust SELinux if enabled.

Determine the current mode of SELinux by running 'getenforce'.

```
[root]#  getenforce

Enforcing
```

If the command returns a mode of "Disabled" or "Permissive", or if the getenforce command is not found, you do not need to do anything and you can skip the rest of this sub-section.

If the command returns a mode of "Enforcing", you may choose between the options of customizing SELinux to allow the web GUI to perform its required functions or disabling SELinux on your system.

If you wish to customize SELinux, you should understand that selinux can vary with version and architecture and these instructions may not suffice for all possible environments. Ultimately, it will be your responsibility to determine and apply the necessary adjustments for the web GUI to work under SELinux. The following example is offered as a set of steps that appeared to work acceptably in our testing for supported RedHat-based systems.

For RedHat-6-based systems:

```
[root]#  cat > mamws.te <<EOF

module mamws 1.0;
require {
        type httpd_t;
        type port_t;
        type usr_t;
        class tcp_socket name_connect;
        class file { create append };
}
allow httpd_t port_t:tcp_socket name_connect;
allow httpd_t usr_t:file { create append };
EOF



[root]#  checkmodule -M -m -o mamws.mod mamws.te

[root]#  semodule_package -m mamws.mod -o mamws.pp

[root]#  semodule -i mamws.pp

[root]#  setenforce 0

[root]#  chcon -v -t httpd_sys_content_t /opt/mam/log

[root]#  setenforce 1
```

For RedHat-7-based or Fedora systems:

```
[root]#  yum install checkpolicy policycoreutils-python

[root]#  cat > mamws.te <<EOF

module mamws 1.0;
require {
        type httpd_t;
        type unreserved_port_t;
        type usr_t;
        class tcp_socket name_connect;
        class file { create unlink append };
}
allow httpd_t unreserved_port_t:tcp_socket name_connect;
allow httpd_t usr_t:file { create unlink append };
EOF
```

```
[root]#  checkmodule -M -m -o mamws.mod mamws.te

[root]#  semodule_package -m mamws.mod -o mamws.pp

[root]#  semodule -i mamws.pp

[root]#  setenforce 0

[root]#  chcon -v -t httpd_sys_rw_content_t /opt/mam/log

[root]#  setenforce 1
```

> **Note:** If you used the --prefix configure option when configuring Moab Accounting Manager, substitute the PREFIX you used for references to /opt/mam in the chcon command.

Alternatively, if you instead choose to disable SELinux, the following steps can be followed.

```
[root]#  vi /etc/sysconfig/selinux

SELINUX=disabled

[root]#  setenforce 0
```

Install a Signed Certificate

For the highest security, it is recommended that you install a public key certificate that has been signed by a certificate authority. The exact steps to do this will be specific to your distribution and the chosen certificate authority. An overview of this process for CentOS 5 is documented at http://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html. Alternatively, if your network domain can be secured from man-in-the-middle attacks, you could use a self-signed certificate. Often this does not require any additional steps since in many distributions (e.g. RedHat-based), the Apache SSL configuration provides self-signed certificates by default.

The following steps assume you are using self-signed certificates:

Create self-signed SSL certificate and key files. Some distributions (e.g. RedHat) ship with ready-made certificates.

For SUSE-based systems:

```
[root]#  cd /etc/apache2

[root]#  openssl genrsa -out ssl.key/server.key 1024

[root]#  openssl req -new -key ssl.key/server.key -x509 -out
ssl.crt/server.crt
```

Start the HTTP Server

Start or restart the Apache HTTP server.

For RedHat-6-based systems:

```
[root]#  chkconfig httpd on
[root]#  service httpd restart
```

For RedHat-7-based or Fedora systems:

```
[root]#  systemctl enable httpd.service
[root]#  systemctl restart httpd.service
```

For SUSE-11-based systems:

```
[root]#  chkconfig apache2 on
[root]#  service apache2 restart
```

For SUSE-12-based systems:

```
[root]#  systemctl enable apache2.service
[root]#  systemctl restart apache2.service
```

For Debian-based systems:

```
[root]#  update-rc.d apache2 defaults
[root]#  service apache2 restart
```

Accessing the GUI

Note: In order to use the web gui, users will have to generate passwords for themselves using the mam-set-password client command. Moab Accounting Manager may have to be restarted in order for role privileges to be reflected in the GUI.

On a MAM Client Host, set a password for the autobuild user to be able to login to the MAM GUI.

```
[root]#  su - autobuild
[autobuild]$  mam-set-password
```

To access the web gui, open a browser with url: https://localhost/mam.

Accessing MAM Web Services

Note: In order to use MAM Web Services, users will have to generate passwords for themselves using the mam-set-password client command. Moab Accounting Manager may have to be restarted in order for role privileges to be reflected in the GUI.

On a MAM Client Host, set a password for the user that you wish to access MAM Web Services.

```
[root]#  su - autobuild
[autobuild]$  mam-set-password
```

Invoke the web services interface.

```
[root]#  curl -k -X GET --basic -u mam:changeme!
'https://localhost/mamws/system'
```

Alternatively for queries, you can use the browser to access the URL, e.g. 'https://localhost/mamws/system'.

Initialization

You are now ready to perform the initial setup in Moab Accounting Manager as necessary for your site. Refer to the Initial Setup chapter to set up Moab Accounting Manager according to your desired accounting mode.

Chapter 3. Upgrading

This chapter describes the process of updating Moab Accounting Manager to a new release version. It includes instructions for migrating your database schema to a new version if this is necessary.

Moab Accounting Manager uses the standard configure, make and make install steps for upgrades. This document provides a number of sample steps referenced to a particular installation on a Linux platform using the bash shell. These steps indicate the userid in brackets performing the step. The exact commands to be performed and the user that issues them will vary based on the platform, shell, installation preferences, etc.

Determine if a database migration is necessary

Check the current database schema version by running ‘mam-shell System Query‘ (or ‘goldsh System Query‘ if you are upgrading from a version earlier than 9.0). If the current version is less than 9.1, then your database will need to be migrated. Follow all instructions in this chapter for migrating your database to 9.1 as necessary.

```
[root]#  su - autobuild
[autobuild]$  mam-shell System Query
```

Note: Use goldsh rather than mam-shell if you are upgrading from a version of Moab Accounting Manager that is earlier than 9.0.

Server Shutdown

Stop the server daemon.

```
[autobuild]$  mam-server -k
```

Note: Use goldd rather than mam-server if you are upgrading from a version of Moab Accounting Manager that is earlier than 9.0.

Database Backup

If you need to migrate your database, create a database backup.

For PostgreSQL database:

```
[autobuild]$  pg_dump -U autobuild -W prior_db_name > /tmp/prior_db_name.sql
```

For MySQL database:

```
[autobuild]$  mysqldump -u autobuild -p prior_db_name > /tmp/prior_db_name.sql
```

Install Prerequisites

Any new prerequisites pertaining to the current release must be applied. Refer to the Install Prerequisites section in the Installation chapter and ensure that all current install prerequisites are satisfied.

Preparation

To build and update Moab Accounting Manager, you first need to unpack the tar archive and change directory into the top directory of the distribution.

```
[autobuild]$  cd ~/src
[autobuild]$  tar -zxvf mam-9.1.0.tar.gz
[autobuild]$  cd mam-9.1.0
```

Configuration

To configure Moab Accounting Manager, run the "configure" script provided with the distribution with the desired options. It is recommended that you use the same configure options that were used in the previous installation. You can examine the config.log file where you unpacked your previous distribution to help determine the configuration options that were used to install the prior version of MAM.

> **Important:** The client and server command names have changed in this release. If you want to create symbolic links to enable you to continue to use the old client and server command names, use the --with-legacy-links option with configure. When running a command under its old name, the command will issue a deprecation warning. This warning can be disabled by setting client.deprecationwarning = false in the mam-client.conf file.

```
[autobuild]$  ./configure
```

Compilation

To compile the program, type make:

```
[autobuild]$  make
```

Installation

Run ‘make install‘ as root to install Moab Accounting Manager.

```
[autobuild]$  su -c "make install"
```

General Setup

Edit the configuration files as necessary. You may want to compare your existing configuration files with those distributed with the new release to determine if you want to merge and change any of the new options within your configuration files.

> **Important:** If you are upgrading from a version of Moab Accounting Manager prior to 9.0, the install process will have saved your goldd.conf to goldd.conf.pre-9.0 and written a new default server configuration file as mam-server.conf. You will need to merge any non-default parameters from goldd.conf.pre-9.0 to the new mam-server.conf file.

```
[autobuild]$  diff /opt/mam/etc/goldd.conf.pre-9.0 /opt/mam/etc/mam-server.conf
[autobuild]$  vi /opt/mam/etc/mam-server.conf
```

> **Note:** If you are upgrading from a version of Moab Accounting Manager that is 9.0 or later, merge and change any of the new options in the mam-server.conf.dist file into your mam-server.conf file.
>
> ```
> [autobuild]$  diff /opt/mam/etc/mam-server.conf /opt/mam/etc/mam-server.conf.dist
> ```

Make sure your current path points to your newly installed clients and server.

```
[autobuild]$  which mam-server
/opt/mam/sbin/mam-server
```

Server Startup

Start the server daemon back up.

```
[autobuild]$  mam-server
```

Database Migration

If you need to migrate your database to 9.1, you will do so by running one or more migration scripts. You must run every incremental migration script between the version you are currently at and the version you are trying to migrate to (9.1). These scripts are designed to be rerunnable, so if you encounter a failure, resolve the failure and rerun the migration script. If you are unable to resolve the failure and complete the migration, contact support.

For example, if you are migrating from Moab Accounting Manager version 7.2, you will need to run five migration scripts: the first to migrate the database schema from 7.2 to 7.3, the second to migrate the database schema from 7.3 to 7.5, the third to migrate the database schema from 7.5 to 8.0, the fourth to migrate the database schema from 8.0 to 8.1 and the fifth to migrate the database schema from 8.1 to 9.0.

```
[autobuild]$  sbin/migrate_7.2-7.3.pl

[autobuild]$  sbin/migrate_7.3-7.5.pl

[autobuild]$  sbin/migrate_7.5-8.0.pl

[autobuild]$  sbin/migrate_8.0-8.1.pl

[autobuild]$  sbin/migrate_8.1-9.0.pl
```

Verify Upgrade

Verify that the resulting database schema version is 9.1.

```
[autobuild]$  mam-shell System Query

Name                    Version Description
----------------------- ------- ------------------
Moab Accounting Manager 9.1     Commercial Release
```

Verify that the executables have been upgraded to 9.1.0.

```
[autobuild]$  mam-server -v

Moab Accounting Manager version 9.1.0
```

Web Services

Note: If you are upgrading from a version of MAM prior to 9.1 and you want to use MAM Web Wervices, follow the instructions in the Installation chapter to install and configure MAM Web Services.

Chapter 4. Initial Setup

After installation, certain steps will need to be performed to prepare Moab Accounting Manager to fulfill its desired role in your environment. Moab Accounting Manager can be configured in a myriad of use cases. It can be used in different accounting modes such as for usage tracking, notional charging or allocation enforcement. This chapter will walk you through some of the steps you need to take to integrate and initialize the accounting manager.

Integrate Moab Accounting Manager with the Moab Workload Manager

If you have not done so already, you will need to configure the Moab suite to interact with Moab Accounting Manager (see Integrating with Moab Workload Manager).

Select an appropriate accounting mode

Moab Accounting Manager can be configured to be used in a variety of different accounting modes. Some sites may wish to create and enforce resource usage limits through allocations. Others may want to impute a charge amount to their workload, but never deny workload based on availability of funds. Still others may not wish to calculate a charge at all, but simply record the usage details of the workload. Select the accounting mode from the following options that best matches your requirements.

- strict-allocation -- Use this mode if you wish to strictly enforce allocation limits. Under this mode, workload can be prevented from running if the end-users do not have sufficient funds. This mode is supported by funds, allocations, quotes, liens, charge rates and usage records. Before a job runs, a lien (or hold) is placed against the user's funds to prevent overcommitment of their allocation. When a job completes, the lien is removed, their allocation is debited, and the workload usage details and charge are recorded in a usage record. This is the normal default.
- fast-allocation -- Use this mode if you wish to debit allocations, but need higher throughput by eliminating the lien and quote of strict-allocation mode. If implemented properly through scripts, the lien and quote of strict-allocation mode can be replaced with an asynchronous balance check, causing accounts to be disabled from further use after the first job that causes the fund to become negative. This mode is supported by funds, allocations, balance checks, charge rates and usage records.
- notional-charging -- Use this mode if you wish to calculate and record charges for workload usage, but not keep track of fund balances or allocation limits. This mode is supported by charge rates and usage records. The workload usage details and charge are recorded in a usage record.
- usage-tracking -- Use this mode if you wish to simply record workload usage details, but not to calculate a charge nor keep track of fund balances or allocation limits. This mode is supported by usage records.

Follow the Setup Guide for your selected accounting mode

Use the respective setup guide to prepare Moab Accounting Manager and Moab Workload Manager to handle accounting according to your chosen accounting mode.

- If you have selected the strict-allocation accounting mode, then refer to the Strict Allocation Setup Guide.
- If you have selected the fast-allocation accounting mode, then refer to the Fast Allocation Setup Guide.
- If you have selected the notional-charging accounting mode, then refer to the Notional Charging Setup Guide.
- If you have selected the usage-tracking accounting mode, then refer to the Usage Tracking Setup Guide.

Chapter 5. Strict Allocation Setup Guide

This chapter will walk you through the typical steps needed to set up Moab Workload Manager and Moab Accounting Manager to use the strict allocation accounting mode.

With the strict allocation accounting mode, you can establish rigorous limits on the use of compute resources by your various parties. This is done by associating a cost for the usage by deciding on a currency unit, generically referred to as credits, whether based on a real currency such as dollars, or a reference currency such as billing units or processor-seconds, and then creating charge rates based on this currency. Funds are established to contain credit allocations attributed to specific accounts. Users are designated as members of the accounts. Deposits are made into funds associated with the accounts creating allocations. An allocation cycle may be established whereby allocations are considered for renewal on a regular periodic basis (such as yearly, quarterly or monthly).

Before a job is started, Moab Workload Manager will verify that the user has sufficient credits to run the job by attempting to place a hold against their funds (referred to as a lien). When a job completes, the user's funds will be debited via a charge, usage information will be recorded for the job and the lien will be removed. Users or managers can query the status of their allocations or details of their job charges and resource utilization.

> **Important:** You will need to be a Moab Accounting Manager System Administrator to perform many of the tasks in this chapter. It is assumed that you have already installed Moab Workload Manager and installed, bootstrapped and started Moab Accounting Manager before performing the steps discussed in this chapter.

Set the accounting mode

Set the AMCFG[mam] MODE parameter to strict-allocation in moab.cfg and set the accounting.mode parameter to strict-allocation in both the mam-server.conf and mam-client.conf files. Since strict allocation is the default accounting mode in both Moab Workload Manager and Moab Accounting Manager, it may not be necessary to do anything here unless you were previously using a different accounting mode.

Example 5-1. Setting the accounting mode to strict-allocation

The AMCFG[] MODE parameter must be set in the Moab server configuration file (moab.cfg). After editing the moab.cfg file, you will need to restart moab.

```
#  vi /opt/moab/etc/moab.cfg
AMCFG[mam] MODE=strict-allocation

#  mschedctl -R
```

The accounting.mode parameter must be set in the server and client configuration files (mam-server.conf and mam-client.conf). After editing the mam-server.conf file, you will need to restart mam-server.

```
$  su - mam
```

```
$ vi /opt/mam/etc/mam-server.conf
accounting.mode = strict-allocation

$ vi /opt/mam/etc/mam-client.conf
accounting.mode = strict-allocation

$ mam-server -r
```

Decide on a currency and set the currency precision

Since we will be calculating charges, we will need to decide what currency unit a MAM credit represents and set the currency precision accordingly. For this example we will define a currency in which one credit represents the value of using one processor core for one hour. We will assume for simplicity that a processor-hour on one machine will have the same value as a processor-hour on another machine. Charge rates will be specified relative to this currency unit. Monetary transactions such as deposits and charges will be specified in terms of this currency. Since we want to be able to track and account for short jobs, we will specify a currency precision of two so that our currency credits will be repesented as a floating point number with two decimal places. If instead we were to have chosen to use processor-seconds as the currency base, we would want to set the currency.precision value to zero since processor seconds can easily be represented as an integer with no decimal places. If we were to have chosen to use dollars as the currency base, we would have set the currency.precision value to two.

Example 5-2. Setting the currency precision to two

The currency precision value must be set in the server and client configuration files (mam-server.conf and mam-client.conf). It must also be set in the GUI configuration file (mam-gui.conf) if you will be using the web GUI. If made changes in mam-server.conf, you will need to restart mam-server.

```
$ vi /opt/mam/etc/mam-server.conf
currency.precision = 2

$ vi /opt/mam/etc/mam-client.conf
currency.precision = 2

$ mam-server -r
```

Customize the Usage Record

Add any additional properties you would like to track in the usage record. The usage properties that you can track will be limited by the properties sent by your resource manager to MAM. If you are using the

Moab Workload Manager, see "Accounting Properties Reported to Moab Accounting Manager" in the Moab Administrator Guide for the list of usage record properties included with the accounting calls to MAM. Refer to the section on Customizing the UsageRecord Object for information on customizing the properties tracked in the usage record.

Define Charge Rates

Since we are charging, we must establish the charge rates for the usage. In our example, we will define a charge rate that charges 1 credit for each processor-hour utilized by the job. See the chapter on Managing Charge Rates for more detailed information on setting up charge rates.

Example 5-3. Define a Charge Rate for Processors

```
$  mam-create-chargerate -n Processors -z 1/h -d "1 credit per
processor-hour"
Successfully created 1 charge rate


$  mam-list-chargerates
Name       Value Amount Description
---------- ----- ------ --------------------------
Processors       1/h    1 credit per processor-hour
```

Define Accounts

Next we will define some accounts and assign users to the accounts. We will also associate each account with an organization so that usage reports can be generated for the organization level as well as the account and user level. We will create accounts for biology, chemistry and film and assign them some users. The biology and chemistry account will be associated with the sciences organization while the film account will be associated with the arts organization. See the chapter on Managing Accounts for more information on setting up accounts.

Example 5-4. Define the biology, chemistry and film accounts.

```
$  mam-create-account -a biology -o sciences -u amy,bob -d "Biology
Department"
Successfully created 1 account


$  mam-create-account -a chemistry -o sciences -u amy,dave -d "Chemistry
Department"
Successfully created 1 account


$  mam-create-account -a film -o arts -u bob,dave -d "Film Department"
```

```
Successfully created 1 account
```

```
$  mam-list-accounts
```

```
Name      Active Users    Organization Description
--------- ------ -------- ------------ --------------------
biology   True   amy,bob  sciences     Biology Department
chemistry True   amy,dave sciences     Chemistry Department
film      True   bob,dave arts         Film Department
```

Create Funds

The next task will be to create the funds which will hold the allocated credits. A fund is much like a numbered bank account, where credits can be deposited and are defined by constraints that distinguish who or what can use the contained credits and for what purposes. In this example, we will create a fund for each of the three accounts. The reason that funds are defined separately from accounts is that it is possible to create multiple funds for the same account. For example, you might have a fund that can be used for the chemistry account only when running the red cluster, and another fund that is used for the chemistry account when using a certain quality of service. See the chapter on Managing Funds for more detailed information on setting up funds.

In this example, we will assume that we want to establish a periodic allocation cycle with predesignated allocation amounts being deposited on a quarterly schedule. In order to facilitate this, we will associate a default deposit amount with the science funds. For the biology fund, we will configure it to make a resetting deposit of 5000 credits for each period. The chemistry fund is going to be disabled at the end of the allocation period. The film account will remain unaffected by allocation renewals. See the chapter on Managing Allocations for more information on periodic allocations.

Example 5-5. Create a fund for each of the three accounts.

```
$  mam-create-fund -a biology -n "biology" --default-deposit 5000
```

```
Successfully created 1 fund with id 1 and 1 constraint
```

```
$  mam-create-fund -a chemistry -n "chemistry" --default-deposit 0
```

```
Successfully created 1 fund with id 2 and 1 constraint
```

```
$  mam-create-fund -a film -n "film"
```

```
Successfully created 1 fund with id 3 and 1 constraint
```

```
$  mam-list-funds
```

```
Id Name      Constraints       Allocated Balance   DefaultDeposit Description
-- --------- ----------------- --------- --------- -------------- -----------
1  biology   Account=biology        0.00      0.00        5000.00
2  chemistry Account=chemistry      0.00      0.00           0.00
3  film      Account=film           0.00      0.00
```

Make Deposits

Now we need to allocate credits to these funds by making deposits to them. An allocation has a start and end time associated with it declaring the time frame in which it can be used (defaulting to a start time of the present and an end time of infinity). It can also have a credit limit which defines the extent to which the allocation is allowed to go negative. Allocations can be reset on a periodic basis or future allocations with different time frames can be precreated within a fund to establish an allocation cycle and set expectations for credit expenditure. See the sections on Managing Allocations and Making Deposits for additional information.

In this example, we will allocate 5000 and 3000 credits to the biology and chemistry accounts respectively. The film account will be given a credit limit of 2000 credits which allows them to charge up to 2000 credits before settling their fund. When making a deposit we must specify the fund we are depositing into unless the fund can be unambiguously determined by its constraint references (i.e. there is only a single fund associated with the account biology). In the next example, we will utilize the fund's default deposit amount in the first deposit, specify the amount explicitly in the second deposit and establish a credit allocation in the third deposit.

Example 5-6. Making Deposits

```
$  mam-deposit -a biology
```

```
Successfully deposited 5000.00 credits into fund 1
Successfully created 1 allocation
```

```
$  mam-deposit -z 3000 -a chemistry
```

```
Successfully deposited 3000.00 credits into fund 2
Successfully created 1 allocation
```

```
$  mam-deposit -L 2000 -a film
```

```
No credits were deposited into fund 3
Successfully created 1 allocation
```

Let's examine the allocations we just created and its effect on the funds.

```
$  mam-list-allocations
```

```
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
1  1    2017-08-09 18:18:56 Infinity        5000.00   5000.00        0.00   5000.00
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   3000.00
3  3    2017-08-09 18:18:57 Infinity           0.00      0.00     2000.00      0.00
```

```
$  mam-list-funds
```

```
Id Name      Constraints       Allocated Balance DefaultDeposit Description
```

```
-- --------- ----------------- --------- ------- -------------- -----------
1  biology   Account=biology     5000.00 5000.00        5000.00
2  chemistry Account=chemistry   3000.00 3000.00           0.00
3  film      Account=film           0.00    0.00
```

Check The Balance

We can verify the resulting balance (see Querying The Balance).

Example 5-7. Let's look at amy's balance

```
$  mam-balance -u amy
Id Name      Balance Reserved Effective CreditLimit Available
-- --------- ------- -------- --------- ----------- ---------
1  biology   5000.00     0.00   5000.00        0.00   5000.00
1  chemistry 3000.00     0.00   3000.00        0.00   3000.00
```

Automate Allocation Renewal

To facilitate the automatic renewal of our allocations, we will create a repeating event that resets all funds (see Creating Events) at the beginning of each new quarter.

Example 5-8. Create an automatic allocation renewal event

```
#  vi /opt/mam/etc/mam-server.conf
event.scheduler = true

$  mam-server -r
$  mam-create-event --fire-command "Fund Reset" -s "2018-01-01"
--rearm-period "3 months^"
Successfully created 1 event

$  mam-list-events
Id FireCommand FireTime   ArmTime             RearmPeriod EndTime Notify RearmOnFailure Fai
-- ----------- ---------- ------------------- ----------- ------- ------ -------------- ---
1  Fund Reset  2018-01-01 2017-08-09 18:21:28 3 months^                 False
```

Run a job

Now, let's submit a job and examine the effects on the accounting system.

Example 5-9. Submit a job

```
$  echo sleep 300 | msub -A chemistry -l procs=12,walltime=600
```

The Usage Lien

When a job starts, Moab Workload Manager typically creates a lien (or hold) against the appropriate allocations based on the estimated duration of the job. We will examine the effect of a running job on the accounting system (see Managing Liens).

Example 5-10. Examine the effect of a running job on the accounting system

```
$  mam-list-liens
Id Instance Amount StartTime           EndTime             Duration UsageRecord Funds Descr
-- -------- ------ ------------------- ------------------- -------- ----------- ----- -----
1  74         2.00 2017-08-09 18:22:42 2017-08-09 18:22:42 600      1           2
```

This lien will decrease our available balance by the amount reserved.

```
$  mam-balance -u amy -a chemistry
Id Name      Balance Reserved Effective CreditLimit Available
-- --------- ------- -------- --------- ----------- ---------
2  chemistry 3000.00     2.00   2998.00        0.00   2998.00
```

Although our actual allocation has not changed.

```
$  mam-list-allocations -u amy -a chemistry
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   3000.00
```

Note that the lien resulted in the initial creation of a usage record for the job with Stage Start.

```
$  mam-list-usagerecords
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         0.00 Start amy  faculty chemistry sciences     batch normal           co
```

The Usage Charge

After a job completes, any associated liens are removed and a charge is issued against the appropriate allocations based on the resources and actual wallclock time used by the job. An allocation is debited and the usage record is modified with the charge and usage information.

Example 5-11. Examine the effect of a completed job on the accounting system

Your allocation will now have gone down by the amount of the charge.

```
$ mam-list-allocations -u amy -a chemistry
```

```
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   2999.00
```

However, your balance actually goes up (because the lien that was removed was larger than the actual charge).

```
$ mam-balance -u amy -a chemistry
```

```
Id Name      Balance Reserved Effective CreditLimit Available
-- --------- ------- -------- --------- ---------- ---------
2  chemistry 2999.00    0.00   2999.00       0.00   2999.00
```

The usage record for the job was updated as a side-effect of the charge (see Querying Usage).

```
$ mam-list-usagerecords
```

```
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         1.00 End   amy  faculty chemistry sciences     batch normal           co
```

Usage Refund

Now, we will illustrate the effect of issuing a refund for the user's job (see Issuing Usage Refunds).

Example 5-12. Refund the Job

```
$ mam-refund -J 74
```

```
Successfully refunded 1.00 credits to usage record 1 for instance 74
```

Our balance is back as it was before the job ran.

```
$ mam-balance -u amy -a chemistry
```

```
Id Name      Balance Reserved Effective CreditLimit Available
-- --------- ------- -------- --------- ---------- ---------
2  chemistry 3000.00    0.00   3000.00       0.00   3000.00
```

The allocation, of course, is likewise restored.

```
$  mam-list-allocations -u amy -a chemistry
```

```
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   3000.00
```

Notice that the usage charge is now zero because the job has been fully refunded.

```
$  mam-list-usagerecords
```

```
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         0.00 End   amy  faculty chemistry sciences     batch normal           co
```

List Transactions

Let's list the transactions relating to this job (see Querying Transactions).

Example 5-13. Listing transaction details for this job

```
$  mam-list-transactions -J 74 --full
```

```
Id   Object      Action  Actor   Key Child Instance Count Amount Delta Balance Remaining Us
---- ----------- ------- ------- --- ----- -------- ----- ------ ----- ------- --------- --
6481 UsageRecord Create  root    1         74       1
6484 UsageRecord Reserve root    1         74       1       2.00                         am
6489 UsageRecord Charge  root    1   74    74       1       1.00 -1.00 2999.00   2999.00 am
6495 UsageRecord Refund  root    1         74       1       1.00  1.00 3000.00   3000.00
```

Examine Fund Statement

Finally, you can examine the fund statement for our activities (see Obtaining a Fund Statement).

Example 5-14. We can request an itemized fund statement over all time for user amy and the chemistry account (fund 2)

```
$  mam-statement -u amy -a chemistry
```

```
################################################################################
#
# Includes fund 2 (chemistry)
# Generated on Tue Aug 9 18:29:53 2017.
# Reporting fund activity from -Infinity to Now.
#
################################################################################
```

```
Beginning Balance:                 0.00
------------------ --------------------
Total Credits:                  3001.00
Total Debits:                     -1.00
------------------ --------------------
Ending Balance:                 3000.00

############################### Credit Detail ###################################

Object      Action  Instance Amount  Balance Time
----------- ------- -------- ------- ------- -------------------
Fund        Deposit          3000.00 3000.00 2017-08-09 18:18:56
UsageRecord Refund  74          1.00 3000.00 2017-08-09 18:28:58

################################ Debit Detail ###################################

Object      Action Instance Account   User Machine Amount Balance Time
----------- ------ -------- --------- ---- ------- ------ ------- -------------------
UsageRecord Charge 74       chemistry amy  colony   -1.00 2999.00 2017-08-09 18:27:42

############################### End of Report ###################################
```

Strict Allocation Initialization Script

Running the hpc-strict-allocation.sh script supplied with your Moab Accounting Manager package has a similar affect to running the example commands in this chapter and will result in MAM being minimally set up for the strict allocation accounting mode with a small amount of dummy sample data. It will not perform the Moab configuration steps described in this chapter. It can be cleaned up by running the hpc-cleanup.sh script.

Example 5-15. Running the hpc-strict-allocation.sh script

```
$ ./hpc-strict-allocation.sh
```

Chapter 6. Fast Allocation Setup Guide

This chapter will walk you through the typical steps needed to set up Moab Workload Manager and Moab Accounting Manager to use the fast allocation accounting mode.

If you want to debit funds in Moab Accounting Manager, but need higher throughput by eliminating the lien and quote operations of the strict allocation accounting mode. With the fast allocation accounting mode, Moab Workload Manager checks a cached account balance, and jobs or reservations may be prevented from starting or continuing after the balance has become zero or negative. As with the strict allocation accounting mode, you establish limits on the use of compute resources by your various parties. This is done by associating a cost for the usage by deciding on a currency unit, generically referred to as credits, whether based on a real currency such as dollars, or a reference currency such as billing units or processor-seconds, and then creating charge rates based on this currency. Funds are established to contain credit allocations attributed to specific accounts. Users are designated as members of the accounts. Deposits are made into funds associated with the accounts creating allocations. An allocation cycle may be established whereby allocations are considered for renewal on a regular periodic basis (such as yearly, quarterly or monthly).

Before a job is started, Moab Workload Manager will check its internal cache to verify that the user has a positive account balance. When a job completes, the user's funds will be debited via a charge, usage information will be recorded for the job and Moab's account balance cache is updated. Since Moab Accounting Manager is not contacted at job submission or start time in order to verify account membership, additional configuration is needed in Moab to synchronize account information with Moab Accounting Manager. Additionally, since the cache in Moab Workload Manager supports only account based funds, when using the fast allocation accounting mode, funds having no constraints or having non-account constraints should not be used.

> **Important:** You will need to be a Moab Accounting Manager System Administrator to perform many of the tasks in this chapter. It is assumed that you have already installed Moab Workload Manager and installed, bootstrapped and started Moab Accounting Manager before performing the steps discussed in this chapter.

Set the accounting mode

Set the AMCFG[mam] MODE parameter to fast-allocation in moab.cfg and set the accounting.mode parameter to fast-allocation in both the mam-server.conf and mam-client.conf files.

Example 6-1. Setting the accounting mode to fast-allocation

The AMCFG[] MODE parameter must be set in the Moab server configuration file (moab.cfg). After editing the moab.cfg file, you will need to restart moab.

```
#  vi /opt/moab/etc/moab.cfg
AMCFG[mam] MODE=fast-allocation

#  mschedctl -R
```

The accounting.mode parameter must be set in the server and client configuration files (mam-server.conf and mam-client.conf). After editing the mam-server.conf file, you will need to restart mam-server.

```
$  su - mam
$  vi /opt/mam/etc/mam-server.conf
accounting.mode = fast-allocation

$  vi /opt/mam/etc/mam-client.conf
accounting.mode = fast-allocation

$  mam-server -r
```

Additional Moab configuration

Since Moab will be checking an internal account balance cache when starting jobs and reservations instead of contacting Moab Accounting Manager, we need to periodically update Moab Workload Manager with account information from Moab Accounting Manager so that Moab can correctly apply default accounts and enforce account memberships. Additionally, it is beneficial to poll the account balances periodically so that external actions such as new deposits, transfers, etc. will be reflected properly in Moab's account balance cache.

Example 6-2. Configuring Moab to synchronize account information

We will set AMCFG[] CREATECRED=TRUE in order to enable Moab to query accounts, users, user membership in accounts, and users' default accounts from Moab Accounting Manager and define them in Moab. We will set the AMCFG[] REFRESHPERIOD parameter to the interval that we want Moab to update these credential updates as well as its account balance cache. We will also set the ENFORCEACCOUNTACCESS parameter to TRUE in order to tell Moab to restrict users to use only accounts that they belong to.

```
#  vi /opt/moab/etc/moab.cfg
AMCFG[mam] CREATECRED=TRUE
AMCFG[mam] REFRESHPERIOD=2:00:00
ENFORCEACCOUNTACCESS TRUE

#  mschedctl -R
```

Decide on a currency and set the currency precision

Since we will be calculating charges, we will need to decide what currency unit a MAM credit represents and set the currency precision accordingly. For this example we will define a currency in which one

credit represents the value of using one processor core for one hour. We will assume for simplicity that a processor-hour on one machine will have the same value as a processor-hour on another machine. Charge rates will be specified relative to this currency unit. Monetary transactions such as deposits and charges will be specified in terms of this currency. Since we want to be able to track and account for short jobs, we will specify a currency precision of two so that our currency credits will be repesented as a floating point number with two decimal places. If instead we were to have chosen to use processor-seconds as the currency base, we would want to set the currency.precision value to zero since processor seconds can easily be represented as an integer with no decimal places. If we were to have chosen to use dollars as the currency base, we would have set the currency.precision value to two.

Example 6-3. Setting the currency precision to two

The currency precision value must be set in the server and client configuration files (mam-server.conf and mam-client.conf). It must also be set in the GUI configuration file (mam-gui.conf) if you will be using the web GUI. If made changes in mam-server.conf, you will need to restart mam-server.

```
$  vi /opt/mam/etc/mam-server.conf
currency.precision = 2

$  vi /opt/mam/etc/mam-client.conf
currency.precision = 2

$  mam-server -r
```

Customize the Usage Record

Add any additional properties you would like to track in the usage record. The usage properties that you can track will be limited by the properties sent by your resource manager to MAM. If you are using the Moab Workload Manager, see "Accounting Properties Reported to Moab Accounting Manager" in the Moab Administrator Guide for the list of usage record properties included with the accounting calls to MAM. Refer to the section on Customizing the UsageRecord Object for information on customizing the properties tracked in the usage record.

Define Charge Rates

Since we are charging, we must establish the charge rates for the usage. In our example, we will define a charge rate that charges 1 credit for each processor-hour utilized by the job. See the chapter on Managing Charge Rates for more detailed information on setting up charge rates.

Example 6-4. Define a Charge Rate for Processors

```
$  mam-create-chargerate -n Processors -z 1/h -d "1 credit per
processor-hour"
```

```
Successfully created 1 charge rate
```

```
$ mam-list-chargerates
```

```
Name       Value Amount Description
---------- ----- ------ --------------------------
Processors       1/h    1 credit per processor-hour
```

Define Accounts

Next we will define some accounts and assign users to the accounts. We will also associate each account with an organization so that usage reports can be generated for the organization level as well as the account and user level. We will create accounts for biology, chemistry and film and assign them some users. The biology and chemistry account will be associated with the sciences organization while the film account will be associated with the arts organization. See the chapter on Managing Accounts for more information on setting up accounts.

Example 6-5. Define the biology, chemistry and film accounts.

```
$ mam-create-account -a biology -o sciences -u amy,bob -d "Biology
Department"
```

```
Successfully created 1 account
```

```
$ mam-create-account -a chemistry -o sciences -u amy,dave -d "Chemistry
Department"
```

```
Successfully created 1 account
```

```
$ mam-create-account -a film -o arts -u bob,dave -d "Film Department"
```

```
Successfully created 1 account
```

```
$ mam-list-accounts
```

```
Name      Active Users    Organization Description
--------- ------ -------- ------------ --------------------
biology   True   amy,bob  sciences     Biology Department
chemistry True   amy,dave sciences     Chemistry Department
film      True   bob,dave arts         Film Department
```

Create Funds

The next task will be to create the funds which will hold the allocated credits. A fund is much like a numbered bank account, where credits can be deposited and are defined by constraints that distinguish

who or what can use the contained credits and for what purposes. In this example, we will create a fund for each of the three accounts. The reason that funds are defined separately from accounts is that it is possible to create multiple funds for the same account. For example, you might have a fund that can be used for the chemistry account only when running the red cluster, and another fund that is used for the chemistry account when using a certain quality of service. See the chapter on Managing Funds for more detailed information on setting up funds.

In this example, we will assume that we want to establish a periodic allocation cycle with predesignated allocation amounts being deposited on a quarterly schedule. In order to facilitate this, we will associate a default deposit amount with the science funds. For the biology fund, we will configure it to make a resetting deposit of 5000 credits for each period. The chemistry fund is going to be disabled at the end of the allocation period. The film account will remain unaffected by allocation renewals. See the chapter on Managing Allocations for more information on periodic allocations.

Example 6-6. Create a fund for each of the three accounts.

```
$ mam-create-fund -a biology -n "biology" --default-deposit 5000
Successfully created 1 fund with id 1 and 1 constraint

$ mam-create-fund -a chemistry -n "chemistry" --default-deposit 0
Successfully created 1 fund with id 2 and 1 constraint

$ mam-create-fund -a film -n "film"
Successfully created 1 fund with id 3 and 1 constraint

$ mam-list-funds
Id Name      Constraints       Allocated Balance   DefaultDeposit Description
-- --------- ----------------- --------- --------- -------------- -----------
1  biology   Account=biology        0.00      0.00        5000.00
2  chemistry Account=chemistry      0.00      0.00           0.00
3  film      Account=film           0.00      0.00
```

Make Deposits

Now we need to allocate credits to these funds by making deposits to them. An allocation has a start and end time associated with it declaring the time frame in which it can be used (defaulting to a start time of the present and an end time of infinity). It can also have a credit limit which defines the extent to which the allocation is allowed to go negative. Allocations can be reset on a periodic basis or future allocations with different time frames can be precreated within a fund to establish an allocation cycle and set expectations for credit expenditure. See the sections on Managing Allocations and Making Deposits for additional information.

In this example, we will allocate 5000 and 3000 credits to the biology and chemistry accounts respectively. The film account will be given a credit limit of 2000 credits which allows them to charge up to 2000 credits before settling their fund. When making a deposit we must specify the fund we are

depositing into unless the fund can be unambiguously determined by its constraint references (i.e. there is only a single fund associated with the account biology). In the next example, we will utilize the fund's default deposit amount in the first deposit, specify the amount explicitly in the second deposit and establish a credit allocation in the third deposit.

Example 6-7. Making Deposits

```
$  mam-deposit -a biology
Successfully deposited 5000.00 credits into fund 1
Successfully created 1 allocation

$  mam-deposit -z 3000 -a chemistry
Successfully deposited 3000.00 credits into fund 2
Successfully created 1 allocation

$  mam-deposit -L 2000 -a film
No credits were deposited into fund 3
Successfully created 1 allocation
```

Let's examine the allocations we just created and its effect on the funds.

```
$  mam-list-allocations
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
1  1    2017-08-09 18:18:56 Infinity        5000.00   5000.00        0.00   5000.00
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   3000.00
3  3    2017-08-09 18:18:57 Infinity           0.00      0.00     2000.00      0.00

$  mam-list-funds
Id Name      Constraints       Allocated Balance DefaultDeposit Description
-- --------- ----------------- --------- ------- -------------- -----------
1  biology   Account=biology     5000.00 5000.00        5000.00
2  chemistry Account=chemistry   3000.00 3000.00           0.00
3  film      Account=film           0.00    0.00
```

Check The Balance

We can verify the resulting balance (see Querying The Balance).

Example 6-8. Let's look at amy's balance

```
$  mam-balance -u amy
Id Name      Balance CreditLimit Available
-- --------- ------- ----------- ---------
```

```
1  biology   5000.00        0.00   5000.00
1  chemistry 3000.00        0.00   3000.00
```

Automate Allocation Renewal

To facilitate the automatic renewal of our allocations, we will create a repeating event that resets all funds (see Creating Events) at the beginning of each new quarter.

Example 6-9. Create an automatic allocation renewal event

```
#  vi /opt/mam/etc/mam-server.conf
event.scheduler = true

$  mam-server -r
$  mam-create-event --fire-command "Fund Reset" -s "2018-01-01"
--rearm-period "3 months^"
Successfully created 1 event

$  mam-list-events
Id FireCommand FireTime   ArmTime             RearmPeriod EndTime Notify RearmOnFailure Fai
-- ----------- ---------- ------------------- ----------- ------- ------ -------------- ---
1  Fund Reset  2018-01-01 2017-08-09 18:21:28 3 months^                  False
```

Run a job

Now, let's submit a job and examine the effects on the accounting system.

Example 6-10. Submit a job

```
$  echo sleep 300 | msub -A chemistry -l procs=12,walltime=600
```

The Usage Charge

After a job completes, a charge is issued against the appropriate allocations based on the resources and actual wallclock time used by the job. An allocation is debited and the usage record is modified with the charge and usage information.

Example 6-11. Examine the effect of a completed job on the accounting system

Your allocation and balance will have gone down by the amount of the charge.

```
$  mam-list-allocations -u amy -a chemistry
```

```
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   2999.00
```

```
$  mam-balance -u amy -a chemistry
```

```
Id Name      Balance CreditLimit Available
-- --------- ------- ----------- ---------
2  chemistry 2999.00        0.00   2999.00
```

The usage record for the job was updated as a side-effect of the charge (see Querying Usage).

```
$  mam-list-usagerecords
```

```
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         1.00 End   amy  faculty chemistry sciences     batch normal           co
```

Usage Refund

Now, we will illustrate the effect of issuing a refund for the user's job (see Issuing Usage Refunds).

Example 6-12. Refund the Job

```
$  mam-refund -J 74
```

```
Successfully refunded 1.00 credits to usage record 1 for instance 74
```

Our balance is back as it was before the job ran.

```
$  mam-balance -u amy -a chemistry
```

```
Id Name      Balance CreditLimit Available
-- --------- ------- ----------- ---------
2  chemistry 3000.00        0.00   3000.00
```

The allocation, of course, is likewise restored.

```
$  mam-list-allocations -u amy -a chemistry
```

```
Id Fund StartTime           EndTime  InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------------------- -------- -------------- --------- ----------- --------- -------
2  2    2017-08-09 18:18:56 Infinity        3000.00   3000.00        0.00   3000.00
```

Notice that the usage charge is now zero because the job has been fully refunded.

```
$  mam-list-usagerecords
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         0.00 End   amy  faculty chemistry sciences     batch normal           co
```

List Transactions

Let's list the transactions relating to this job (see Querying Transactions).

Example 6-13. Listing transaction details for this job

```
$  mam-list-transactions -J 74 --full
Id   Object      Action  Actor   Key Child Instance Count Amount Delta Balance Remaining Us
---- ----------- ------- ------- --- ----- -------- ----- ------ ----- ------- --------- --
6489 UsageRecord Charge  root    1   74    74       1       1.00 -1.00 2999.00   2999.00 am
6495 UsageRecord Refund  root    1         74       1       1.00  1.00 3000.00   3000.00
```

Examine Fund Statement

Finally, you can examine the fund statement for our activities (see Obtaining a Fund Statement).

Example 6-14. We can request an itemized fund statement over all time for user amy and the chemistry account (fund 2)

```
$  mam-statement -u amy -a chemistry
################################################################################
#
# Includes fund 2 (chemistry)
# Generated on Tue Aug 9 18:29:53 2017.
# Reporting fund activity from -Infinity to Now.
#
################################################################################

Beginning Balance:                 0.00
------------------ --------------------
Total Credits:                  3001.00
Total Debits:                     -1.00
------------------ --------------------
Ending Balance:                 3000.00

################################ Credit Detail ###################################

Object      Action  Instance Amount  Balance Time
----------- ------- -------- ------- ------- -------------------
```

```
Fund        Deposit          3000.00 3000.00 2017-08-09 18:18:56
UsageRecord Refund  74          1.00 3000.00 2017-08-09 18:28:58

############################### Debit Detail ###################################

Object      Action Instance Account   User Machine Amount Balance Time
----------- ------ -------- --------- ---- ------- ------ ------- -------------------
UsageRecord Charge 74       chemistry amy  colony   -1.00 2999.00 2017-08-09 18:27:42

############################### End of Report ##################################
```

Fast Allocation Initialization Script

Running the hpc-fast-allocation.sh script supplied with your Moab Accounting Manager package has a similar affect to running the example commands in this chapter and will result in MAM being minimally set up for the fast allocation accounting mode with a small amount of dummy sample data. It will not perform the Moab configuration steps described in this chapter. It can be cleaned up by running the hpc-cleanup.sh script.

Example 6-15. Running the hpc-fast-allocation.sh script

```
$ ./hpc-fast-allocation.sh
```

Chapter 7. Notional Charging Setup Guide

This chapter will walk you through the typical steps needed to set up Moab Workload Manager and Moab Accounting Manager to use the notional charging accounting mode.

Some sites may want to use Moab Accounting Manager to calculate and record charges, but not to impose allocation limits or prevent any workload from running. With notional charging, charge rates will be used to calculate a cost for using resources, but there is no need to make deposits, debit funds or keep track of allocation limits. Although it would be possible to set up accounts and assign users to specific accounts, this chapter will assume that account membership is not going to be enforced. If you would prefer to enforce account membership, you can continue to use the notional charging accounting setup as described in this chapter, but you will need to additionally define accounts and account memberships as well as configure Moab to synchronize account information from Moab Accounting Manager as described in the Fast Allocation Setup Guide. Liens, balance queries and quotes are not needed. Our main task is to define charge rates.

At the end of a job, Moab Workload Manager will send usage information to the accounting manager. Moab Accounting Manager will calculate a charge and store this with the job usage record.

> **Important:** You will need to be a Moab Accounting Manager System Administrator to perform many of the tasks in this chapter. It is assumed that you have already installed Moab Workload Manager and installed, bootstrapped and started Moab Accounting Manager before performing the steps discussed in this chapter.

Set the accounting mode

Set the AMCFG[mam] MODE parameter to notional-charging in moab.cfg and set the accounting.mode parameter to notional-charging in both the mam-server.conf and mam-client.conf files.

Example 7-1. Setting the accounting mode to notional-charging

The AMCFG[] MODE parameter must be set in the Moab server configuration file (moab.cfg). After editing the moab.cfg file, you will need to restart moab.

```
#  vi /opt/moab/etc/moab.cfg
AMCFG[mam] MODE=notional-charging

#  mschedctl -R
```

The accounting.mode parameter must be set in the server and client configuration files (mam-server.conf and mam-client.conf). After editing the mam-server.conf file, you will need to restart mam-server.

```
$  su - mam
$  vi /opt/mam/etc/mam-server.conf
accounting.mode = notional-charging
```

```
$ vi /opt/mam/etc/mam-client.conf
accounting.mode = notional-charging

$ mam-server -r
```

Decide on a currency and set the currency precision

Since we will be calculating charges, we will need to decide what currency unit a MAM credit represents and set the currency precision accordingly. For this example we will define a currency in which one credit represents the value of using one processor core for one hour. We will assume for simplicity that a processor-hour on one machine will have the same value as a processor-hour on another machine. Charge rates will be specified relative to this currency unit. Monetary transactions such as deposits and charges will be specified in terms of this currency. Since we want to be able to track and account for short jobs, we will specify a currency precision of two so that our currency credits will be repesented as a floating point number with two decimal places. If instead we were to have chosen to use processor-seconds as the currency base, we would want to set the currency.precision value to zero since processor seconds can easily be represented as an integer with no decimal places. If we were to have chosen to use dollars as the currency base, we would have set the currency.precision value to two.

Example 7-2. Setting the currency precision to two

The currency precision value must be set in the server and client configuration files (mam-server.conf and mam-client.conf). It must also be set in the GUI configuration file (mam-gui.conf) if you will be using the web GUI. If made changes in mam-server.conf, you will need to restart mam-server.

```
$ vi /opt/mam/etc/mam-server.conf
currency.precision = 2

$ vi /opt/mam/etc/mam-client.conf
currency.precision = 2

$ mam-server -r
```

Customize the Usage Record

Add any additional properties you would like to track in the usage record. The usage properties that you can track will be limited by the properties sent by your resource manager to MAM. If you are using the Moab Workload Manager, see "Accounting Properties Reported to Moab Accounting Manager" in the Moab Administrator Guide for the list of usage record properties included with the accounting calls to

MAM. Refer to the section on Customizing the UsageRecord Object for information on customizing the properties tracked in the usage record.

Define Charge Rates

Since we are charging, we must establish the charge rates for the usage. In our example, we will define a charge rate that charges 1 credit for each processor-hour utilized by the job. See the chapter on Managing Charge Rates for more detailed information on setting up charge rates.

Example 7-3. Define a Charge Rate for Processors

```
$  mam-create-chargerate -n Processors -z 1/h -d "1 credit per
processor-hour"
Successfully created 1 charge rate

$  mam-list-chargerates
Name       Value Amount Description
---------- ----- ------ --------------------------
Processors       1/h    1 credit per processor-hour
```

Run a job

Now, let's submit a job and examine the effects on the accounting system.

Example 7-4. Submit a job

```
$  echo sleep 300 | msub -A chemistry -l procs=12,walltime=600
```

The Usage Charge

After a job completes, a usage record is generated with the charge and resource usage information.

Example 7-5. List the usage and charge for our job

```
$  mam-list-usagerecords
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         1.00 End   amy  faculty chemistry sciences     batch normal           co
```

Usage Refund

Now, we will illustrate the effect of issuing a refund for the user's job (see Issuing Usage Refunds).

Example 7-6. Refund the Job

```
$ mam-refund -J 74
Successfully refunded 1.00 credits to usage record 1 for instance 74
```

Notice that the usage charge is now zero because the job has been fully refunded.

```
$ mam-list-usagerecords
Id Type Instance Charge Stage User Group   Account   Organization Class QualityOfService Ma
-- ---- -------- ------ ----- ---- ------- --------- ------------ ----- ---------------- --
1  Job  74         0.00 End   amy  faculty chemistry sciences     batch normal           co
```

List Transactions

Let's list the transactions relating to this job (see Querying Transactions).

Example 7-7. Listing transaction details for this job

```
$ mam-list-transactions -J 74 --full
Id   Object      Action  Actor   Key Child Instance Count Amount User Account   Machine Fun
---- ----------- ------- ------- --- ----- -------- ----- ------ ---- --------- ------- ---
6489 UsageRecord Charge  root    1   74    74       1       1.00 amy  chemistry colony
6495 UsageRecord Refund  root    1         74       1       1.00
```

Notional Charging Initialization Script

Running the hpc-notional-charging.sh script supplied with your Moab Accounting Manager package has a similar affect to running the example commands in this chapter and will result in MAM being minimally set up for the notional charging accounting mode with a small amount of dummy sample data. It will not perform the Moab configuration steps described in this chapter. It can be cleaned up by running the hpc-cleanup.sh script.

Example 7-8. Running the hpc-notional-charging.sh script

```
$ ./hpc-notional-charging.sh
```

Chapter 8. Usage Tracking Setup Guide

This chapter will walk you through the typical steps needed to set up Moab Workload Manager and Moab Accounting Manager to use the usage tracking accounting mode.

When used solely for usage tracking, Moab Accounting Manager logs resource usage in usage records. This usage can be queried to report what resources were used when and by whom. In this case, there is no need for charge rates, funds, allocations, liens or quotes. There is no need to define account membership.

At the end of a job, Moab Workload Manager will send usage information to the accounting manager. Moab Accounting Manager will store this information in a job usage record.

> **Important:** You will need to be a Moab Accounting Manager System Administrator to perform many of the tasks in this chapter. It is assumed that you have already installed Moab Workload Manager and installed, bootstrapped and started Moab Accounting Manager before performing the steps discussed in this chapter.

Set the accounting mode

Set the AMCFG[mam] MODE parameter to usage-tracking in moab.cfg and set the accounting.mode parameter to usage-tracking in both the mam-server.conf and mam-client.conf files.

Example 8-1. Setting the accounting mode to usage-tracking

The AMCFG[] MODE parameter must be set in the Moab server configuration file (moab.cfg). After editing the moab.cfg file, you will need to restart moab.

```
#  vi /opt/moab/etc/moab.cfg
AMCFG[mam] MODE=usage-tracking

#  mschedctl -R
```

The accounting.mode parameter must be set in the server and client configuration files (mam-server.conf and mam-client.conf). After editing the mam-server.conf file, you will need to restart mam-server.

```
$  su - mam
$  vi /opt/mam/etc/mam-server.conf
accounting.mode = usage-tracking

$  vi /opt/mam/etc/mam-client.conf
accounting.mode = usage-tracking

$  mam-server -r
```

Customize the Usage Record

Add any additional properties you would like to track in the usage record. The usage properties that you can track will be limited by the properties sent by your resource manager to MAM. If you are using the Moab Workload Manager, see "Accounting Properties Reported to Moab Accounting Manager" in the Moab Administrator Guide for the list of usage record properties included with the accounting calls to MAM. Refer to the section on Customizing the UsageRecord Object for information on customizing the properties tracked in the usage record.

Run a job

Now, let's submit a job and examine the effects on the accounting system.

Example 8-2. Submit a job

```
$  echo sleep 300 | msub -A chemistry -l procs=12,walltime=600
```

Query job usage information

After a job completes, usage information is recorded. Let's examine the usage record that was created (see Querying Usage).

Example 8-3. List usage records

```
$  mam-list-usagerecords
Id Type Instance Stage User Group   Account   Organization Class QualityOfService Machine N
-- ---- -------- ----- ---- ------- --------- ------------ ----- ---------------- ------- -
1  Job  74       End   amy  faculty chemistry sciences     batch normal           colony  1
```

Usage Tracking Initialization Script

Running the hpc-usage-tracking.sh script supplied with your Moab Accounting Manager package has a similar affect to running the example commands in this chapter and will result in MAM being minimally set up for the usage tracking accounting mode with a small amount of dummy sample data. It will not perform the Moab configuration steps described in this chapter. It can be cleaned up by running the hpc-cleanup.sh script.

Example 8-4. Running the hpc-usage-tracking.sh script

```
$  ./hpc-usage-tracking.sh
```

Chapter 9. Managing Users

A user is a person authorized to use a resource or service. Default user properties include the common name, phone number, email address, default account, and description for that person. A user can be created, queried, modified and deleted. By default, a standard user may only query their own user record.

User queries allow the specification of filter options which narrow down the users that will be returned to those belonging to the specified account.

Creating Users

To create a new user, use the command **mam-create-user**:

mam-create-user {[-u] user_name} [-A | -I] [-n common_name] [--phone phone_number] [--email email_address] [-a default_account] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 9-1. Creating a user

```
$ mam-create-user -n "Smith, Robert F." --email "bob@bank.com" --phone
"(509) 555-1234" bob
Successfully created 1 user
```

Querying Users

To display user information, use the command **mam-list-users**:

mam-list-users [[-u] user_pattern] [-A | -I] [-X, --extension property=value]... [-a account_name] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 9-2. Listing standard info about active users

```
$ mam-list-users -A
Name Active CommonName       PhoneNumber    EmailAddress     DefaultAccount Description
---- ------ ---------------- -------------- ---------------- -------------- -----------
amy  True   Wilkes, Amy      (509) 555-8765 amy@bank.com
bob  True   Smith, Robert F. (509) 555-1234 bob@bank.com
```

Example 9-3. Displaying bob's phone number

```
$ mam-list-users --show PhoneNumber bob --quiet
(509) 555-1234
```

Example 9-4. Listing amy's accounts

```
$  mam-list-users --show Accounts amy --long -q
chemistry
biology
```

Example 9-5. Listing all users belonging to the chemistry account

```
$  mam-list-users --show Name -a chemistry -q
amy
dave
```

Modifying Users

To modify a user, use the command **mam-modify-user**:

mam-modify-user {[-u] user_name} [-A | -I] [-n common_name] [--phone phone_number] [--email email_address] [-a default_account] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 9-6. Deactivating a user

```
$  mam-modify-user -I bob
Successfully modified 1 user
```

> **Note:** In order for user validity enforcement to occur, the Values property for the UsageRecord User attribute must be set to '@User'.
>
> ```
> $  mam-shell Attribute Modify Object==UsageRecord Name==User Values=@User
> ```

Example 9-7. Changing a user's email address

```
$  mam-modify-user --email "rsmith@cs.univ.edu" bob
Successfully modified 1 user
```

Example 9-8. Setting a user's default account

```
$ mam-modify-user -a chemistry amy
Successfully modified 1 user
```

Deleting Users

To delete a user, use the command **mam-delete-user**:

mam-delete-user {[-u] user_name} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 9-9. Deleting a user

```
$ mam-delete-user bob
Successfully deleted 1 user
```

User Auto-Generation

If user auto-generation is enabled (this is the default), users will automatically be created when first added as a member to an account or role. It is also possible to have users be created automatically when first encountered in a usage function (charge, reserve or quote). In order for user auto-generation to occur, the AutoGen property for the User object must be set to 'True'. This is the default. Additionally, for user auto-generation to occur when a user is added as a member of another object (such as Account) via an association table (e.g. AccountUser), the Values property for the user attribute of the Association (e.g. Name) must be set to '@User', indicating that that value should be constrained to be a valid instance of the User object. For user auto-generation to occur when initially encountered in a usage function, the Values property of the user attribute of the UsageRecord object must be similarly set to '@User'. The auto-creation of users can be completely disabled by setting the AutoGen property for the User object to 'False'.

Example 9-10. Enable auto-generation of users when initially seen in a charge

```
$ mam-shell Attribute Modify Object==UsageRecord Name==User Values=@User
Successfully modified 1 attribute
```

Example 9-11. Disable all auto-generation of users

```
$ mam-shell Object Modify Name==User AutoGen=False
Successfully modified 1 object
```

See Object Auto-Generation for more information about the auto-generation of objects.

Default User

It is possible to set a global default user to which usage would be ascribed in quotes, liens or charges where no user is specified. This can be accomplished by setting the DefaultValue property for the User object to the desired user.

It is also possible to set a custom user default for a specific object, which will result in usage being ascribed to the specified user when the object is attributed to the usage. This is done by creating a default usage override modifier. For example, to specify that acmeuser be the default user for usage associated with the acme organization, you might first create an attribute called DefaultUser for the Organization Object with the Values property of @?=User. Then you would populate the new DefaultUser property for the acme organization with the value of acmeuser. See the chapter on Customizing Objects for more information on default and other usage override modifiers.

Example 9-12. Assign a global default user

```
$  mam-shell Object Modify Name==User DefaultValue=anonymous
Successfully modified 1 object
```

Chapter 10. Managing Accounts

An account represents a work entity requiring the use of resources or services for a common purpose such as a project or cost-center. Users may be designated as members of an account and may be allowed to share its allocations. If the special 'ANY' user is added to an account, then any user may use funds allocated to the account. The user members may be designated as active or inactive, and as an administrator for the account. Default account properties include the description, the organization it is part of, and whether or not it is active. An account can be created, queried, modified and deleted. An account's user membership can also be adjusted. By default, a standard user may only query accounts they belong to.

Account queries allow the specification of filter options which narrow down the accounts that will be returned to those having the specified users in them.

Creating Accounts

To create a new account, use the command **mam-create-account**:

mam-create-account {[-a] account_name} [-A | -I] [-o organization_name] [-d description] [-X, --extension property=value]... [-u [^|!][+|-]user_name,...]... [--create-fund True\|False] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

When defining users, the optional caret or exclamation symbol indicates whether the user should be created as an admin (^) or not (!) for the account. The optional plus or minus sign can precede each member to indicate whether the member should be created in the active (+) or inactive (-) state. By default, a user will be created in the active state but not an admin. Multiple users may be passed to the -u option in a comma-delimited list. Alternatively, multiple -u options may be specified.

> **Note:** If the Fund object's AutoGen property is set to true (see Fund Auto-Generation), a fund will be automatically created for the account (unless overridden with the --create-fund option). The auto-generated fund will be associated with the new account.

Example 10-1. Creating an account

```
$  mam-create-account -d "Chemistry Department" chemistry
Successfully created 1 account
```

Example 10-2. Creating an account and specifying user members at the same time

In this example, we make amy the account admin and associate the account with the sciences organization.

```
$  mam-create-account -d "Chemistry Department" -u ^amy,bob,dave chemistry -o
sciences
Successfully created 1 account
```

Example 10-3. Creating an account that can be used by any user

```
$  mam-create-account -d "Common Account" -u ANY common
Successfully created 1 account
```

Querying Accounts

To display account information, use the command **mam-list-accounts**:

mam-list-accounts [[-a] account_pattern] [-A | -I] [-o organization_name] [-X, --extension property=value]... [-u user_name] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 10-4. Listing all info about all accounts

```
$  mam-list-accounts
Name      Active Users     Organization Description
--------- ------ --------- ------------ --------------------
biology   True   amy,^bob  sciences     Biology Department
chemistry True   ^amy,dave sciences     Chemistry Department
film      True   amy,^dave arts         Film Department
```

Example 10-5. Displaying the name and user members of an account in long format

```
$  mam-list-accounts --show Name,Users --long chemistry
Name      Users
--------- -----
chemistry ^amy
          dave
```

Example 10-6. Listing all account names

```
$  mam-list-accounts --show Name --quiet
biology
chemistry
film
```

Example 10-7. Listing all accounts that have dave as a member

```
$  mam-list-accounts --show Name -u dave --quiet
chemistry
film
```

Modifying Accounts

To modify an account, use the command **mam-modify-account**:

mam-modify-account {[-a] account_name} [-A | -I] [-o organization_name] [-d description] [-X, --extension property=value]... [--add-user(s) [^|!][+|-]user_name,...]... [--del-user(s) user_name,...]... [--mod-user(s) [^|!][+|-]user_name,...]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

User members may be added, removed or modified in an account. When adding user members to an account, the optional caret or exclamation symbol indicates whether the user should be created as an admin (^) or not (!) for the account. The optional plus or minus signs can precede each member to indicate whether the member should be created in the active (+) or inactive (-) state. When modifying user members of an account, the caret symbol or exclamation symbol indicates the user should be changed to become an admin (^) or not (!) for the account. The plus or minus signs indicate whether the user should be changed to become active (+) or inactive (-). If an active or admin modifier is not specified, that aspect of the user member will remain unchanged. If the user.firstaccountdefault server parameter is set to true, the first account that a user is added to will additionally become the default account for that user.

Example 10-8. Deactivating an account

```
$  mam-modify-account -I chemistry
Successfully modified 1 account
```

> **Note:** In order for account validity enforcement to occur, the Values property for the UsageRecord Account attribute must be set to '@Account'.
>
> ```
> $  mam-shell Attribute Modify Object==UsageRecord Name==Account Values=@Account
> ```

Example 10-9. Adding users as members of an account

```
$  mam-modify-account --add-users jsmith,barney chemistry
Successfully added 2 users
```

Example 10-10. Deactivating a user in an account

```
$  mam-modify-account --mod-user -dave chemistry
Successfully modified 1 user
```

Deleting Accounts

To delete an account, use the command **mam-delete-account**:

mam-delete-account {[-a] account_name} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 10-11. Deleting an account

```
$  mam-delete-account chemistry
Successfully deleted 1 account
```

Account Auto-Generation

It is possible to have accounts be created automatically when first encountered in a usage function (charge, reserve or quote). It is also possible for accounts to be automatically created when initially added as a member of another object. In order for account auto-generation to occur, the AutoGen property for the Account object must be set to 'True'. This is the default. For account auto-generation to occur when initially encountered in a usage function, the Values property of the account attribute of the UsageRecord object must be set to '@Account'. Additionally, for account auto-generation to occur when an account is added as a member of another object (such as the Organization object) via an association table (e.g. OrganizationAccount), the Values property for the account attribute of the Association (e.g. Name) must be set to '@Account', indicating that that value should be constrained to be a valid instance of the Account object. The auto-creation of accounts can be completely disabled by setting the AutoGen property for the Account object to 'False'.

Example 10-12. Enable auto-generation of accounts when initially seen in a charge

```
$  mam-shell Attribute Modify Object==UsageRecord Name==Account
Values=@Account
Successfully modified 1 attribute
```

Example 10-13. Disable all auto-generation of accounts

```
$  mam-shell Object Modify Name==Account AutoGen=False
Successfully modified 1 object
```

See Object Auto-Generation for more information about the auto-generation of objects.

Default Account

It is possible to set a global default account to which usage would be ascribed in quotes, liens or charges where no account is specified. This can be accomplished by setting the DefaultValue property for the Account object to the desired account name.

A per-user default account can be established by setting the DefaultAccount property for the user. If the user.firstaccountdefault server parameter is set to true (the default), the first account that a user is added to will automatically become the default account for that user. Otherwise, one can use the **mam-modify-user** command along with the -a option to set or change the default account for the user.

It is also possible to set a custom account default for a specific object, which will result in usage being ascribed to the specified account when the object is attributed to the usage. This is done by creating a default usage override modifier. For example, to specify a default account of testing for the beta organization, you might first create an attribute called DefaultAccount for the Organization Object with the Values property of @?=Account. Then you would populate the new DefaultAccount property for the beta organization with the value of testing. See the chapter on Customizing Objects for more information on default and other usage override modifiers.

Example 10-14. Assign a global default account

```
$  mam-shell Object Modify Name==Account DefaultValue=common
Successfully modified 1 object
```

Chapter 11. Managing Organization

An organization is virtual organization in which accounts are grouped. An account may only belong to a single organization while an organization may have multiple accounts. For example, an account may represent a project or cost-center while an organization may represent an institutional department or business division. The purpose of defining organizations is to support the ability to produce reporting for higher-order organizational entities beyond the individual account. Default organization properties include a name and a description. An organization can be created, queried, modified and deleted.

Creating Organizations

To create a new organization, use the command **mam-create-organization**:

mam-create-organization {[-o] organization_name} [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 11-1. Creating an organization

```
$  mam-create-organization -d "Sciences Department" sciences
Successfully created 1 organization
```

Querying Organizations

To display organization information, use the command **mam-list-organizations**:

mam-list-organizations [[-o] organization_pattern] [-X, --extension property=value]... [--full] [--show attribute_name,...] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 11-2. Listing all organization names

```
$  mam-list-organizations --show Name -q
arts
sciences
```

Modifying Organizations

To modify an organization, use the command **mam-modify-organization**:

mam-modify-organization {[-o] organization_name} [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 11-3. Changing an organization's description

```
$ mam-modify-organization -d "Art College" art
Successfully modified 1 organization
```

Deleting Organizations

To delete an organization, use the command **mam-delete-organization**:

mam-delete-organization {[-o] organization_name} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 11-4. Deleting an organization

```
$ mam-delete-organization arts
Successfully deleted 1 organization
```

Organization Auto-Generation

It is possible to have organizations be created automatically when first encountered in a usage function (charge, reserve or quote). It is also possible to have organizations automatically be created when initially added as a member of another object. In order for organization auto-generation to occur, the AutoGen property for the Organization object must be set to 'True'. This is the default. For organization auto-generation to occur when initially encountered in a usage function, the Values property of the organization attribute of the UsageRecord object must be set to '@Organization'. Additionally, for organization auto-generation to occur when an organization is added as a member of another object (such as a hypothetical Site object) via an association table (e.g. SiteOrganization), the Values property for the organization attribute of the Association (e.g. Name) must be set to '@Organization', indicating that that value should be constrained to be a valid instance of the Organization object. The auto-creation of organizations can be completely disabled by setting the AutoGen property for the Organization object to 'False'.

Example 11-5. Enable auto-generation of organizations when initially seen in a charge

```
$ mam-shell Attribute Modify Object==UsageRecord Name==Organization
Values=@Organization
Successfully modified 1 attribute
```

Example 11-6. Disable all auto-generation of organizations

```
$ mam-shell Object Modify Name==Organization AutoGen=False
```

```
Successfully modified 1 object
```

See Object Auto-Generation for more information about the auto-generation of objects.

Default Organization

It is possible to set a global default organization to which usage would be ascribed in quotes, liens or charges where no organization is specified. This can be accomplished by setting the DefaultValue property for the Organization object to the desired organization name.

It is also possible to set an organization default for a specific object, which will result in usage being ascribed to the specified organization when the object is attributed to the usage. This is done by creating a default usage override modifier. For example, to specify that retail be the default organization for usage associated with the user amy, you might first create an attribute called DefaultOrganization for the User Object with the Values property of @?=Organization. Then you would populate the new DefaultOrganization property for the amy user with the value of retail. See the chapter on Customizing Objects for more information on default and other usage override modifiers.

Example 11-7. Assign a global default organization

```
$  mam-shell Object Modify Name==Organization DefaultValue=sciences
Successfully modified 1 object
```

Chapter 12. Managing Funds

A fund is a container for a time-bounded reference currency called credits, the usage of which is restricted by constraints that define how the credits must be used. A fund is like a bank account for resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

Funds may have a name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging. A default deposit amount can be set for a fund which is used when the amount is not specified for a deposit. Derived properties such as Allocated, Balance, Effective, Available, Capacity, PercentRemaining, PercentUsed and Used can be displayed via the **mam-list-funds** or **mam-balance** commands (see the commands reference for mam-list-funds or mam-balance for more details). Operations include creating, querying, modifying, deleting and resetting funds as well as making deposits, withdrawals, transfers and balance queries. By default, a standard user may only query and view the balance for funds which pertain to them.

Credits are added to a fund via a deposit. If no amount is specified for the deposit, the fund's default deposit value is used for the deposit amount. When credits are deposited into a fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. The initial deposit into a fund will create a new allocation having the specified or default time boundaries.

A fund may be reset, causing the currently active allocation to end and creating a new allocation. When a fund is reset, the default deposit amount will be used to determine the amount of the new allocation. A zero default deposit amount will result in the creation of an allocation with a zero balance. A negative default deposit amount can be used to stipulate that the allocations in the fund should be ended if the fund is reset. An empty default deposit amount stipulates that no change will be made to the allocations if the fund is reset. As an alternative to resetting funds, allocations with predesignated start and end times may be created ahead of time. By using one of these methods to implement periodic allocations, it is possible to establish an allocation cycle. See the chapter on Allocations for more information on periodic allocations as well as credit limits and infinite allocations.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for credits deposited into higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Some fund operations (Query, Deposit, Withdraw and Refund) allow the specification of filter options which narrow down the funds that will be acted on for that operation. There are three funds filter types that can be employed that modify the behavior of the filtering: ExactMatch, Exclusive and NonExclusive. If an exact-match filter type is used, the query will return only the funds for which the specified filters exactly match the fund constraints. For example, Allocation Query FilterType:=ExactMatch Filter:=User=amy would only return a fund with the sole constraint User=amy. If an exclusive filter type is used, the query will return only the funds for which the specified filters meet all constraints for usage. Another way to think of an exclusive filter is to ask if usage were to be posted given only the specified filter options as ACLs, which funds would be eligible for charging? For example, Fund Query FilterType:=Exclusive Filter:=User=amy would not return a fund with the sole

constraint Machine=blue because Machine=blue was not included in the filters. Not only must the filters be a non-conflicting superset of the fund constraints, but all constraint dependencies must also be satisfied (for example, an appropriate user may need to be specified with the account). If a non-exclusive filter type is used, the query will return all funds for which the filters do not specifically exclude the constraints. The query assumes that if constraints are not specified within the filters, they can be assumed as a wildcard and will return all funds that are not specifically excluded by the filter. For example, Fund Query FilterType:=NonExclusive Filter:=User=amy would return a fund whose only constraint was Machine=blue (because it does not conflict) but would not return a fund with the constraint User=bob (because it does conflict). The NonExclusive filter type will be used by default if no filter type is specified.

Creating Funds

mam-create-fund is used to create a new fund. One can specify a fund name, a description, and any number of fund constraints. If a name is not specified and constraints are specified, a name will be automatically generated based on the constraints. A new unique id is automatically generated for the fund.

mam-create-fund [-n fund_name] [--priority fund_priority] [--default-deposit deposit_amount] [-d description] [-X, --extension property=value]... [-u user_name,...]... [-g group_name,...]... [-a account_name,...]... [-o organization_name,...]... [-c class_name,...]... [-m machine_name,...]... [--constraint constraint_name=[!]constraint_value,...]... [--parent parent_fund_id] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

> **Note:** It is possible to have funds be created automatically when accounts are created by setting the Fund object's AutoGen property to true (see Fund Auto-Generation). The auto-generated fund will be associated with the new account.

Example 12-1. Creating a fund valid for the chemistry account

```
$  mam-create-fund -a chemistry -n "Chemistry"
Successfully created 1 fund with id 7 and 1 constraint
```

Example 12-2. Creating a wide-open fund that can be used by anyone for anything

```
$  mam-create-fund -n "Windfall"
Successfully created 1 fund with id 8
```

Example 12-3. Creating a fund valid toward all biology account members except for dave and just the machine colony

```
$  mam-create-fund --constraint Account=biology,User=!dave,Machine=colony -n
"Biology on Colony not for Dave"
```

```
Successfully created 1 fund with id 9 and 3 constraints
```

Querying Funds

To display fund information, use the command **mam-list-funds**:

mam-list-funds [[-f] fund_id] [-A | -I] [-n fund_name] [-X, --extension property=value]... [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\|Exclusive\|NonExclusive] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv\|raw\|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 12-4. Listing all info about all funds with multi-valued fields displayed in a multi-line format

```
$ mam-list-funds --long
```

```
Id Name              Constraints       Allocated Balance  DefaultDeposit Description
-- ----------------- ----------------- --------- -------- -------------- -----------
1  biology           Account=biology    25000000 25000000       25000000
2  chemistry for amy User=amy           35000000 34802392       35000000
                     Account=chemistry
3  chemistry not amy User=!amy          50000000 50000000       50000000
                     Account=chemistry
4  film on colony    Account=film             0         0
                     Machine=colony
```

Example 12-5. Wide listing all info about all funds useable by amy

```
$ mam-list-funds -u amy
```

```
Id Name              Constraints                  Allocated Balance  DefaultDeposit Descrip
-- ----------------- ---------------------------- --------- -------- -------------- -------
1  biology           Account=biology               25000000 25000000       25000000
2  chemistry for amy Account=chemistry,User=amy    35000000 34802392       35000000
4  film on colony    Machine=colony,Account=film          0        0
```

Modifying Funds

To modify a fund, use the command **mam-modify-fund**:

mam-modify-fund [[-f] fund_id] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\|Exclusive\|NonExclusive] {{[-n fund_name] [--priority fund_priority] [--default-deposit deposit_amount] [-d description] [-X, --extension property=value]... [--add-constraint

constraint_name=[!]constraint_value,...]... [--del-constraint(s) constraint_name[=constraint_value],...]... [--parent parent_fund_id]} | {--reset [--all]}} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 12-6. Adding a constraint to a fund that it can only be used by the acme organization

```
$ mam-modify-fund --add-constraint Organization=acme 7
Successfully added 1 constraint
```

Example 12-7. Setting the default deposit amount for a fund

```
$ mam-modify-fund --default=deposit 5000000 -f 1
Successfully modified 1 fund
```

Example 12-8. Resetting a fund

```
$ mam-modify-fund --reset 1
Successfully deposited 5000000 credits into fund 1
Successfully stopped 1 allocation
Successfully created 1 allocation
```

Making Deposits

mam-deposit is used to deposit time-bounded credits into a fund resulting in the creation or increase of an allocation. (See Allocations for managing allocations). The start time will default to -infinity and the end time will default to infinity if not specified. Filter options can be specified to help select a unique fund for the deposit. If multiple funds are matched by the filters, the matching funds will be listed and you will be prompted to respecify the deposit with one of the fund ids. If an allocation for the deposit fund is found having the start and end times for the deposit, the amount of the allocation will be increased by the deposit amount. Otherwise, a new allocation will be created for the fund with the amount of the deposit. If no funds match your criteria, if fund auto-generation is enabled, a fund will be created and the deposit made into it, otherwise the deposit will fail (the fund will need to be first created using **mam-create-fund**).

Deposits may be used to extend the debit ceiling by specifying an amount for the deposit (with the -z option) or extend the credit floor by specifying a credit limit for the deposit (with the -L option) or a combination of both options may be used. Additionally, Infinity may be used for either of these option values when Moab Accounting Manager is coupled with a database that supports IEEE Standard 754 for Floating-Point Arithmetic (e.g. PostgreSQL).

To make a deposit, use the command **mam-deposit**:

mam-deposit [-f fund_id] [-i allocation_id] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]...

[--filter-type ExactMatch\Exclusive\NonExclusive] [[-z] deposit_amount] [-L credit_limit] [-s start_time] [-e end_time] [--reset] [-d description] [--create-fund True\False] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 12-9. Making a deposit into fund 1

```
$  mam-deposit -z 360000000 -f 1
```

```
Successfully deposited 360000000 credits into fund 1
Successfully created 1 allocation
```

Example 12-10. Making a deposit "into" an account

If an account has a single fund then a deposit can be made against the account.

```
$  mam-deposit -z 360000000 -a chemistry
```

```
Successfully deposited 360000000 credits into fund 2
Successfully created 1 allocation
```

Example 12-11. Creating a credit allocation

```
$  mam-deposit -L 10000000000 -f 3
```

```
Successfully deposited 0 credits into fund 3
Successfully created 1 allocation
```

Example 12-12. Making a reset deposit

Stop the active allocation within a fund and create a new allocation.

```
$  mam-deposit -f 4 -z 36000000 --reset
```

```
Successfully deposited 36000000 credits into fund 4
Successfully stopped 1 allocation
Successfully created 1 allocation
```

Example 12-13. Creating an infinite allocation

```
$  mam-deposit -z Infinity -f 5
```

```
Successfully deposited inf credits into fund 5
Successfully created 1 allocation
```

> **Note:** The use of infinite allocations requires the use of a database that supports the IEEE Standard 754 for Floating-Point Arithmetic (e.g. PostgreSQL).

Example 12-14. Creating a future quarterly allocation

```
$  mam-deposit -s 2017-10-01 -e 2018-01-01 -z 25000000 -a biology
Successfully deposited 25000000 credits into fund 6
Successfully created 1 allocation
```

Querying The Balance

To display balance information, use the command **mam-balance**:

mam-balance [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\|Exclusive\|NonExclusive] [--ignore-ancestors] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv\|raw\|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

> **Note:** The fields which are displayed by default by the mam-balance command can be customized by setting the balance.show configuration parameter in mam-client.conf.

Example 12-15. Querying amy's balance

```
$  mam-balance -u amy
Id Name      Balance Reserved Effective CreditLimit Available
-- --------- ------- -------- --------- ----------- ---------
1  biology   2785.87   103.22   2682.65        0.00   2682.65
1  chemistry 1785.87     0.00   1785.87        0.00   1785.87
```

Example 12-16. List the available balances that amy can charge against along with the constraints on those balances

```
$  mam-balance -u amy --show Available,Constraints
Available Constraints
--------- --------------------------
 25000000 Account=biology
 34802392 Account=chemistry,User=amy
        0 Machine=colony,Account=film
```

Personal Balance

The **mybalance** has been provided as a wrapper script to show users their personal balance. It provides a list of balances for the funds that they can charge to:

mybalance [--hours] [--help] [--man]

Example 12-17. List my fund balances

```
$ mybalance
Name              Available
----------------- ---------
biology            25000000
chemistry for amy  34802392
```

Example 12-18. List my balance in (Processor) hours

```
$ mybalance -h
Name              Available
----------------- ---------
biology              6944.44
chemistry for amy    9667.33
```

Making Withdrawals

A withdrawal can be used to debit a fund without being associated with the usage charge from some item. To issue a withdrawal, use the command **mam-withdraw**:

mam-withdraw [-f fund_id] [-i allocation_id] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\Exclusive\NonExclusive] {[-z] withdrawal_amount} [-d description] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 12-19. Making a withdrawal

```
$ mam-withdraw -z 12800 -f 1 -d "Grid Tax"
Successfully withdrew 12800 credits from fund 1
```

Example 12-20. Making a withdrawal "from" an account

If an account has a single fund then a withdrawal can be made against the account.

```
$ mam-withdraw -z 12800 -a biology
Successfully withdrew 12800 credits from fund 1
```

If more than one fund exist for the account or filter, you will be asked to be more specific:

```
$  mam-withdraw -z 12800 -a chemistry
Multiple funds were matched for the withdrawal.
Please respecify using one of the following fund ids:
2   [chemistry for amy]
3   [chemistry not amy]
```

Making Transfers

To issue a transfer between funds, use the command **mam-transfer**. If the allocation id is specified, then only credits associated with the specified allocation will be transferred, otherwise, only active credits will be transferred. Fund transfers preserve the allocation time periods associated with the resource or service credits from the source to the destination funds. Source and destination filters may be used if they result in a single source fund and single destination fund.

mam-transfer {--from-fund source_fund_id &| --from-allocation source_allocation_id &| --from-filter filter_name=filter_value...} {--to-fund destination_fund &| --to-allocation destination_allocation_id &| --to-filter filter_name=filter_value...} [--filter-type ExactMatch|Exclusive|NonExclusive] {[-z] transfer_amount} [-d description] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 12-21. Transferring credits between two funds

```
$  mam-transfer --from-fund 1 --to-fund 2 10000
Successfully transferred 10000 credits from fund 1 to fund 2
```

Example 12-22. Transferring credits between two single-fund accounts

```
$  mam-transfer --from-filter Account=biology --to-filter Account=chemistry
10000
Successfully transferred 10000 credits from fund 1 to fund 2
```

Obtaining a Fund Statement

To generate a fund statement, use the command **mam-statement**. For a specified time frame it displays the beginning and ending balances as well as the total credits and debits to the fund over that period. This is followed by an itemized report of the debits and credits. Summaries of the debits and credits will be displayed instead of the itemized report if the --summarize option is specified. If filter options are

specified instead of a fund, then the statement will consist of information merged from all funds valid toward the specified entities.

mam-statement [[-f] fund_id] [-n fund_name] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\|Exclusive\|NonExclusive] [-s start_time] [-e end_time] [--summarize] [--hours] [--debug] [--site site_name] [--help] [--man] [--version] [--about]

Example 12-23. Generating a fund statement for all chemistry funds for the fourth quarter of 2016

```
$  mam-statement -a chemistry -s 2016-10-01 -e 2017-01-01 --summarize
################################################################################
#
# Includes fund 3 (chemistry not amy)
# Includes fund 2 (chemistry for amy)
# Generated on Mon Feb  7 18:44:23 2017.
# Reporting fund activity from 2016-10-01 to 2017-01-01.
#
################################################################################

Beginning Balance:                    0
------------------ --------------------
Total Credits:                 90122212
Total Debits:                  -5308668
------------------ --------------------
Ending Balance:                84813544

################################ Credit Summary #################################

Object      Action  Amount
----------- ------- --------
Fund        Deposit 90100000
UsageRecord Refund     22212

################################ Debit Summary ##################################

Object      Action Account   User Machine Amount   Count
----------- ------ --------- ---- ------- -------- -----
UsageRecord Charge chemistry amy  colony  -5219820 239

################################ End of Report ##################################
```

> **Note:** The fields which are used as default discriminators in the detail section of the mam-statement command (which are by default Account, User and Machine) can be customized by setting the statement.show configuration parameter in mam-client.conf.

Deleting Funds

To delete a fund, use the command **mam-delete-fund**:

mam-delete-fund {[-f] fund_id} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 12-24. Deleting a fund

```
$  mam-delete-fund 2
Successfully deleted 1 fund
```

Fund Auto-Generation

It is possible to enable the auto-generation of funds by setting the AutoGen property of the Fund object to True. When creating a new account, if fund auto-generation is enabled, a fund will automatically be created for the account (unless overridden with the --create-fund option). The fund will be usable only by usage attributed to the new account. Additionally, if fund auto-generation is set, a deposit that does not match an existing fund will automatically generate a fund using the filters as constraint options. Objects associated with the constraint that have AutoGen set to True will be auto-generated as well (unless overridden with thee --create-fund option).

Example 12-25. Enable auto-generation of funds

```
$  mam-shell Object Modify Name==Fund AutoGen=True
Successfully modified 1 object
```

Hierarchical Funds

A hierarchy can be established between funds. When creating a fund or by modifying it later, one can specify a parent fund id via the --parent option to establish the object fund as a child of the specified parent fund. A fund may have multiple children funds but only a single parent fund.

Example 12-26. Establishing a child relationship with another fund

```
$  mam-modify-fund --parent 3 -f 6
Successfully added 1 parent
```

Deposit shares may be established between the parent fund and its children that assist to automate a trickle-down effect for funds deposited at higher level funds (DepositShare is an attribute of the FundFund association object). Deposit shares are integers and are treated as a percentage of each deposit and the sum of the deposit shares for a fund's children may not exceed 100. If the deposit shares for the

children of a fund totals less than 100, the difference is taken to be the share of the deposit that will be allocated to the parent. When a deposit is made into a parent fund, for each child fund that has a non-zero deposit share a recursive deposit amounting to the designated percentage of the parent deposit is issued to that child. After the share amounts have been deposited to each of the child funds, the remaining percentage of the deposit is allocated to the parent fund. This effect is recursive with each child. If a start time and/or end time are specified in the original deposit, these time frames will be recursively applied to all descendant deposits. You have to use the mam-shell interactive control program to manage deposit shares. For the FundFund association object, the Fund is the parent and the Id is the child.

Example 12-27. Establishing a 10% deposit share between a parent and a child fund

```
$  mam-shell FundFund Modify Fund==3 Id==6 DepositShare=10
Fund Id DepositShare Overflow
---- -- ------------ --------
3    6  10           False
Successfully modified 1 fundFund
```

An overflow policy may be established between the parent fund and its children to enable a trickle-up effect for charges, liens and quotes from the lower level funds (Overflow is an attribute of the FundFund association object). The Overflow attribute is a boolean value (True or False). If the overflow value between a child and its parent is set to True, any charges, liens or quotes issued against the child fund that cannot be satisfied by the balance in the child fund, will recursively issue the unsatisfied portion of the charge, lien or quote against the parent fund. If the charge, lien or quote cannot be satisfied by the ancestors, no charges, liens or quotes will result against any of funds. The balance in the descendant funds will be depleted before ancestor funds. This effect is recursive with each parent. If a parent fund is linked with overflow to a child fund and a charge, lien or quote overflows to the parent fund, the constraints of the parent fund will not be checked against the properties of the item. One must use the mam-shell interactive control program to manage the overflow policy. For the FundFund association object, the Fund is the parent and the Id is the child.

Example 12-28. Enabling overflow between a parent and a child fund

```
$  mam-shell FundFund Modify Fund==3 Id==6 Overflow=True
Fund Id DepositShare Overflow
---- -- ------------ --------
3    6  10           True
Successfully modified 1 fundFund
```

Fund Priority

By default, when an item can charge to multiple funds, funds with more constraints are chosen over funds with fewer constraints. For example, if the user amy is charging against the chemistry account for usage of an item and there are two viable funds, one with a single constraint (e.g. Account=chemistry) and another with two constraints (e.g. Account=chemistry and User=amy), credits will be taken from the

more specific fund (with 2 constraints) before they are taken from the more general fund (with 1 constraint). To override this behavior, it is possible to give a priority to a fund. The priority factor of a fund has higher precedence than the specificity (constraint count) of the fund. Thus, all else being equal, if a fund with a lower number of constraints is given a higher priority than a fund with a higher number of constraints, the higher priority fund will be depleted first. Other factors, such as the end time of the allocation or whether there is an existing lien for the item against a fund, have a higher precedence than the specificity of the fund. If you want the allocations in a particular fund to be chosen before allocations that expire sooner or that have a lien, you may need to specify fund priorities that are in the millions (See Allocation Precedence for a discussion of the manner of sorting allocations for charging).

Example 12-29. Setting a fund priority

```
$  mam-modify-fund -f 3 --priority 1
Successfully modified 1 fund
```

Chapter 13. Managing Allocations

An allocation is a time-bounded pool of credits associated with a fund. A fund may have multiple allocations, each for use during a different time period. Normally, only a single allocation will be active within a fund at any given time.

Allocations are normally created via a fund deposit. An allocation has an amount, an initial deposit and an allocated value. The Amount attribute tracks the current amount of credits in the allocation. The InitialDeposit attribute stores the amount originally deposited into an allocation when it is initially created. The Allocated attribute stores the current adjusted allocated amount. It is initially set to the initial deposit amount and is incremented with each crediting deposit or incoming transfer and decremented with each withdrawal or outgoing transfer. When a deposit is made to a fund, if a matching allocation already exists with the appropriate time period, the existing allocation is modified. Otherwise, a new allocation is created. A resetting deposit will end the currently active allocation and create a new allocation.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. If a start time or end time is specified when making a deposit, an existing allocation having the specified boundary times will be credited. If no start time or end time is specified, the active allocation will be credited. If no matching or active allocations can be found, a new allocation will be created with the specified or default start and end time (the start time defaults to the present and the end time defaults to infinity). An active flag is automatically updated to True if the allocation is within its valid timeframe or False if it is not. An allocation that becomes active because the current time is greater than its start time undergoes an activation which normally registers as a credit to the fund. An allocation that becomes inactive because the current time is greater than its end time undergoes a deactivation which normally registers as a debit to the fund.

By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle. There are two primary methods to implement periodic allocations. In the first method, called Resetting Allocations, funds are reset (ending the current allocation and creating a new one) at the beginning of each allocation period. By setting and maintaining an appropriate default deposit amount for each fund, the process of resetting funds can be simplified. The periodic reset can be performed either by making a resetting deposit for each fund (e.g. **mam-deposit -f 1 --reset**) which allows you to override default deposit amounts, by calling the reset action for each fund (e.g. **mam-modify-fund -f 1 --reset**) which allows you to select which funds to reset, or by invoking a reset across all funds (e.g. **mam-modify-fund --reset --all**). The effect of any of these commands is to end the currently active allocation in the fund and then make a fresh deposit. The fund's default deposit amount is used any time the amount is not specified in a deposit (as in the case of a fund reset command). If the default deposit amount is positive, the currently active allocation is ended and a new allocation is created with the default amount. If the default deposit amount is set to a value of zero, the active allocation is ended and no new allocation is created. If the default deposit amount is not set, the fund's allocations are not affected. The reset can be performed via a scheduled event or via a cron script. If default deposit amounts are kept up-to-date (including being zeroed out for funds that are slated to end and being unset for funds that you do not want affected by the reset), automation of this method can be as simple as creating a single periodic event with a FireCommand of "Fund Reset" (see Creating Events). In the second method, called Expiring Allocations, funds with predesignated start and end times are created ahead of time. When the beginning of an allocation period is reached, the currently active allocation automatically expires and the next one automatically becomes active. A future allocation is created by making a deposit while specifying a start time and an end time in the future (e.g.

mam-deposit -f 1 -s 2017-10-01 -e 2018-01-01). This method can also take advantage of default deposit amounts. The overall effect of either of these methods is very similar.

By default, Moab Accounting Manager attempts to enforce Discrete Allocations, i.e. ensure that allocations within a fund are non-overlapping (in time) and non-reusable (each allocation period should use a distinct allocation). This behavior is designated by the allocation.enforcediscrete server configuration parameter. If set to true, this policy prevents new allocations within a fund from overlapping existing ones. Enabling this policy helps to improve clarity when reporting on allocation usage during a particular period. If set to false, overlapping allocations within a fund can be created. This might be useful if you want to allow the remaining balance from a prior allocation period to carry over into the new allocation period. With overlapping allocations, it is harder to describe what percentage of a group's allocation has been used. This policy is applied when making deposits that create new allocations, when making transfers that create new allocations, or when modifying the start and end times of an existing allocation. It is possible to override the configured policy for an individual command by specifying the EnforceDiscrete option (e.g. **mam-deposit --option name=EnforceDiscrete value=False**).

An allocation may have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit account, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit account, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

It is possible for the allocation Amount or CreditLimit to be set to Infinity (via a deposit). If the amount is infinite, debits will not decrease the balance. An infinite deposit will result in an infinite Allocated amount. If the credit limit is infinite, there will be no negative limit for debits. It is not possible to have infinite charges, liens, quotes, withdrawals, refunds or transfers. However, it is possible to have infinite allocation activations, deactivations and deletions. This capability is only available when using a database that supports IEEE Standard 754 for Floating-Point Arithmetic (e.g. PostgreSQL).

Operations include querying, modifying, and deleting allocations. Allocations can be created by a fund deposit, creating a fund with allocation auto-generation enabled, refunding a usage record, or a transfer between funds. Allocations may also be indirectly modified via charges, withdrawals, transfers or refunds. By default, a standard user may only query allocations which pertain to them.

Allocation queries allow the specification of filter options which filter the allocations to those with funds meeting the specified fund constraints. There are three allocation filter types that can be employed: ExactMatch, Exclusive and NonExclusive. If an exact-match filter type is used, the query will return only the allocations relating to funds for which the specified filters exactly match the constraints. For example, Allocation Query FilterType:=ExactMatch Filter:=User=bob would only return an allocation for a fund with the sole constraint User=bob. If an exclusive filter type is used, the query will return only allocations relating to funds for which the specified filters meet all constraints. For example, Allocation Query FilterType:=Exclusive Filter:=User=amy would not return an allocation for a fund with the sole constraint Machine=blue. If a non-exclusive filter type is used, the query will return all allocations relating to funds for which the filters do not specifically exclude the constraints. The query assumes that if constraints are not specified within the filters, they can be assumed as a wildcard and will return all allocations involving funds that are not specifically excluded by the filter. For example, Allocation Query FilterType:=NonExclusive Filter:=User=amy would return an allocation with a fund whose only constraint was Machine=blue but would not return an allocation with a fund with the constraint User=bob. The NonExclusive filter type will be used if no filter type is specified.

Creating Allocations

Allocations are normally created by making fund deposits via the mam-deposit command (see Making Deposits).

Querying Allocations

To display allocation information, use the command **mam-list-allocations**:

mam-list-allocations [[-i] allocation_id] [-f fund_id] [-A | -I | {[-s start_time] [-e end_time]}] [-X, --extension property=value]... [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type ExactMatch\Exclusive\NonExclusive] [--include-ancestors] [--full] [--show attribute_name,...] [--format csv\raw\standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 13-1. Listing allocations for fund 1

```
$  mam-list-allocations -f 1
Id Fund Active StartTime  EndTime    InitialDeposit Allocated CreditLimit Remaining Percent
-- ---- ------ ---------- ---------- -------------- --------- ----------- --------- -------
1  1    True   2017-01-01 2017-04-01       25000000  25000000           0  24974400
2  1    False  2017-04-01 2017-07-01       25000000  25000000           0  25000000
3  1    False  2017-07-01 2017-10-01       25000000  25000000           0  25000000
4  1    False  2017-10-01 2018-01-01       25000000  25000000           0  25000000
```

Modifying Allocations

To modify an allocation, use the command **mam-modify-allocation**:

mam-modify-allocation {[-i] allocation_id} [-s start_time] [-e end_time] [-L credit_limit] [-d description] [-X, --extension property=value]... [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 13-2. Changing the end time for an allocation

```
$  mam-modify-allocation -e "2018-01-01" 4
Successfully modified 1 allocation
```

Example 13-3. Changing the credit limit for an allocation

```
$  mam-modify-allocation -L 500000000000 -i 2
Successfully modified 1 allocation
```

Deleting Allocations

To delete an allocation, use the command **mam-delete-allocation**:

mam-delete-allocation {-I | {[-i] allocation_id}} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 13-4. Deleting an allocation

```
$ mam-delete-allocation 4
Successfully deleted 1 allocation
```

Example 13-5. Purging inactive allocations

```
$ mam-delete-allocation -I
Successfully deleted 2 allocations
```

Allocation Auto-Generation

It is possible to enable the auto-generation of allocations by setting the AutoGen property of the Allocation object to True. When creating a new fund, if allocation auto-generation is enabled, an allocation will automatically be created for the fund via a deposit. The deposit will use the default amount and default credit limit (defined in the DefaultValue property of the Allocation Amount and Allocation CreditLimit attributes). The default action for allocation auto-generation is to create an allocation with an infinite credit limit.

Example 13-6. Enable auto-generation of allocations

```
$ mam-shell Object Modify Name==Allocation AutoGen=True
Successfully modified 1 object
```

Allocation Precedence

When issuing a charge (or a lien or quote) for the usage of a resource or service, the feasible allocations are sorted according to a weight given to them for that transaction. The weight for each allocation is calculated as follows: Independent of precedence, if the instance has current liens against one or more allocations, the reserved allocations will be debited first in order to avoid double booking. For the

remaining non-nested funds, allocations will be given a value of 100 * int((2147483647 - <end_epoch_time>) / 86400) + 10 * <fund_priority> + <constraint_count>. Thus, sooner expiring allocations will be used before later expiring allocations, fund priority will be the next highest factor (assuming small priority values of 1-10), followed by the number of constraints on the fund (more specific funds will be used before more general funds). Of course, since priority is configurable, a sufficiently large priority (in the millions) can be used to override the precedence of earlier expiring allocations. Lastly, nested funds that become feasible because of overflow to ancestor funds have a negative weighting and are used last, with the earliest expiring allocations being used before later expiring allocations and closer level ancestors being depleted before ancestor funds that are at more distant levels. These allocations are given a weight of <distance * 100000> - <end_epoch_time>. After all feasible allocations are sorted according to the above rules, the charge (or lien or quote) will be applied against the allocations one by one in sorted order (highest value first) until the request is fulfilled, or until it fails due to insufficient funds. If a transaction is not able to be satisfied in whole, in the case of a charge, partial debits will be applied and the entire transaction will succeed regardless of the amount successfully debited, whereas for a quote or lien, the entire transaction will fail and no partial debits will be applied.

Chapter 14. Managing Liens

A lien is a reservation or hold placed against an allocation. Before usage of a resource or service begins, a lien is placed against one or more allocations within the requesting user's applicable funds. Subsequent usage requests will also post liens while the available balance (active allocations minus liens) allows. When the usage ends, the lien is removed and the actual charge is made to the allocation(s). This procedure ensures that usage will only be permitted so long as the requestors have sufficient funds.

Associated with a lien is the instance name (name of the item being used such as the job id), the usage record (which contains the item details), a start time and end time for the lien and a description. The lien will automatically expire and no longer count against the user's balance after the end time passes. Each lien will be associated with held amounts from one or more allocations. Operations include creating, querying, modifying and deleting liens. By default, a standard user may only query liens attributed to them.

Lien queries allow the specification of filter options which narrow down the liens that will be returned. There are two lien filter types that can be employed: AttributedTo and ImpingesUpon. If ImpingesUpon is used, the query will return all liens associated with funds satisfying the filters. For example, Lien Query FilterType:=ImpingesUpon Filter:=User=scottmo will return all liens impinging on Funds usable by scottmo. If AttributedTo is used, the query will return all liens associated with usage records satisfying the filters. For example, Lien Query FilterType:=AttributedTo Filter:=User=scottmo will return all liens for resources or services allocated to scottmo.

When a lien is created via the UsageRecord Reserve action (such as via mam-reserve), if another lien exists with the same instance name, the default behavior is to leave the old lien in place (and create the new one along side it). This behavior assumes that the other lien is probably a separate lien created by a resource or service manager that reuses instance ids. However, alternate behaviors may be specified via the mutually exclusive Modify or Replace options. If the Replace option is specified, any pre-existing liens with matching instance names will first be deleted, thereby ensuring only one lien per instance name at a time. If the Modify option is specified, a pre-existing lien with matching instance name will be modified to have the new properties (but keeping the same lien id), and can be used to extend a lien. This might be used with incremental charging to dynamically stretch liens along a little at a time as needed. (See Making Usage Liens for a description of the action using these options).

Liens may be granted a grace period (in seconds), which is defined as the difference between the validity period of the lien (end time minus start time) and the expected duration of the usage. The purpose of a grace period is to account for the fact that we may not know precisely when the usage will begin and the lien needs to be remain in force during the lifetime of the usage. One can apply a desired grace period for a lien by setting the end time longer than the specified duration. Alternatively, a grace duration option can be specified with the duration when creating a lien via **mam-reserve** as a helper to computing a relatively adjusted end time.

Creating Liens

Liens are normally created with the mam-reserve command (see Making Usage Liens).

However, it is also possible to create a manual lien against specified allocations using the **mam-create-lien** command. A lien object and its allocation associations will be created. Unlike **mam-reserve**, no calculated lien amount will be returned nor will a usage record be created or updated

with the lien. Furthermore, **mam-create-lien** will not perform any checking to ensure that the specified allocations have a sufficient active balance to support the lien.

mam-create-lien [-J instance_name] [-s start_time] {-e end_time | -t lien_duration} [-d description] [-X, --extension property=value]... {-A allocation_id<-fund_id=sublien_amount,...}... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 14-1. Creating a manual lien

```
$ mam-create-lien -J weekend_run -t 84600 -A "5<-2=3600"
Successfully created 1 lien
```

> **Warning**
>
> Use of the **mam-create-lien** command bypasses the normal mechanisms that prevent more liens from being placed against an allocation than it can support. Use **mam-reserve** instead if you wish to avoid the possibility of oversubscribing the allocations.

Querying Liens

To display lien information, use the command **mam-list-liens**:

mam-list-liens [[-l] lien_id] [-A| -I] [-J instance_pattern] [-X, --extension property=value]... [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--filter-type AttributedTo|ImpingesUpon] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 14-2. Listing all info about all liens for amy

```
$ mam-list-liens -u amy
Id Instance   Amount StartTime           EndTime             UsageRecord Funds Description
-- ---------- ------ ------------------- ------------------- ----------- ----- -----------
3  PBS.1234.4  57600 2017-04-06 21:21:48 2017-04-06 22:31:48 7           2
```

Example 14-3. Listing all info about all liens that impinge against dave's balance

```
$ mam-list-liens -u dave --filter-type ImpingesUpon
Id Instance Amount StartTime           EndTime             UsageRecord Funds Description
-- -------- ------ ------------------- ------------------- ----------- ----- -----------
4  batch.12   7600 2017-04-06 15:30:34 2017-04-06 15:41:50 244         3
```

Example 14-4. Listing total of lien amounts broken down by attributed account

```
$ mam-list-liens --show "GroupBy(Account),Sum(Amount)=Reserved"
```

```
Account   Reserved
--------- --------
biology       1.00
chemistry     4.00
```

Modifying Liens

To modify a lien, use the command **mam-modify-lien**:

mam-modify-lien {[-l] lien_id} [-s start_time] [-e end_time] [-t lien_duration] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 14-5. Changing the expiration time of a lien

```
$ mam-modify-lien -e "2017-06-04 14:43:02" 1
```

```
Successfully modified 1 lien
```

Deleting Liens

To delete a lien, use the command **mam-delete-lien**:

mam-delete-lien {-I | {-J instance_name} | {[-l] lien_id}} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 14-6. Deleting a lien by instance (or job id)

```
$ mam-delete-lien -J PBS.1234.0
```

```
Successfully deleted 1 lien
```

Example 14-7. Deleting a lien by Lien Id

```
$ mam-delete-lien 1
```

```
Successfully deleted 1 lien
```

Example 14-8. Purging stale liens

```
$  mam-delete-lien -I
Successfully deleted 2 liens
```

Chapter 15. Managing Quotes

A quote provides a way to determine beforehand the cost of using resources or services. When a guaranteed quote is requested, the charge rates applicable to the usage request are saved and a quote id is returned. Charge rates may be specified with the quote or the standard rates may be used in the quote calculation. When the lien and the final charge are issued, the quote id can be referenced to ensure that the saved quote charge rates are used instead of current standard values. A quote has an expiration time after which it cannot be used. A quote may also be used to verify that the given usage request has sufficient funds and meets the policies necessary for the charge to succeed. Additionally, it is possible to request a cost-only quote to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

Associated with a quote is the id, the instance name (name of the item being used such as the job id), the amount quoted (assuming full use of the quoted resources or services), the usage record (which contains the usage details), a start and end time for the quote, a duration (how long the item is expected to be used), a boolean indicating whether the quote is pinned or unpinned, and a description. Each guaranteed quote will be associated with one or more saved charge rates. Operations include creating, querying, modifying and deleting quotes. By default, a standard user may only query quotes attributed to them.

Quote queries allow the specification of filter options which narrow down the quotes that will be returned. The query will return all quotes associated with usage records satisfying the filters. For example, Quote Query Filter:=User=scottmo will return all quotes for resources or services allocated to scottmo.

A quote may be pinned (restricted to a particular instance) or unpinned (allowed to be used by any number of different instances). If a quote is pinned and has not been tied to a particular instance when initially created, it will be tied to the first instance that claims it. Once pinned to an instance, it can then be used repeatedly by that same instance until the quote expires, but not by any other instance. If a quote is not pinned, any instances may use the quoted rates while the quote is active.

A quote may be granted a grace period, which is defined as the difference between the validity period of the quote (end time minus start time) and the expected duration of the usage in seconds. The purpose of a grace period is to account for the fact that we may not know precisely when the usage will begin and the quote needs to be valid during the time of completion of the usage in order for the guaranteed charge rates to be applied. One can apply a desired grace period for a quote by setting the end time longer than the specified duration. Alternatively, a grace duration option can be specified with the duration when creating a quote via **mam-quote** as a helper to computing a relatively adjusted end time.

A distinction may be made between quotes and quote templates, both of which use the Quote object. A quote will always return a cost estimate and will be associated with a specific usage record. A quote template provides a way to bundle together a package of special charge rates that can be applied to quotes, liens and charges. Quote templates use the same Quote object as regular quotes but they are not associated with a usage record and do not generate a quote amount.

In calculating a price, a quote will use (in order of lower to higher precedence) the standard charge rates, the charge rates from a specified quote template, the specified override charge rates, or an externally specified charge amount. In saving guaranteed charge rates, the standard charge rates pertaining to the specified usage record properties will be used unless overridden by a specified quote template or specified charge rates.

There are several key purposes for using quotes and quote templates. First, a quote may be requested to discover the cost of using a resource or service. If this is your sole purpose, then you may want to use the

mam-quote command with the --costOnly option. Second, a quote can be used to check whether the requestor has sufficient access and funds to use the requested resource. This may be accomplished by invoking the mam-quote command without the --costOnly option. Third, a quote or a quote template can be used to lock-in current or specified charge rates for use in future liens and charges. If the details of the usage are known and you would like to get a quote amount with a quote id that can be referenced to guarantee the quoted charge rates, you may use the mam-quote command with the --guarantee option. Override charge rates may be factored in to the cost estimate of the quote by using the mam-quote command with the --rate option. If specific override charge rates need to be saved or guaranteed for future use within a quote, lien or charge without generating a cost estimate, create a pinned quote template by using the mam-create-quote command with the --pin and --rate options. If it is necessary to create a quote template that can be used to override the standard charge rates for multiple instances, use the mam-create-quote command with the --nopin and --rate options.

Creating Quotes

Quotes are normally generated by the resource management system with the mam-quote command before an instance uses requested resources or services (see Making Usage Quotes).

Creating Quote Templates

Quote templates may be created by using the **mam-create-quote** command. Quote templates provide a way to bundle together a package of special charge rates that can be applied to quotes, liens and charges.

mam-create-quote [[--pin] [-J instance_name] | --nopin] [-s start_time] {-e end_time | -t quote_duration} [-d description] [-X, --extension property=value]... {--rate charge_rate_name[{charge_rate_value}]=charge_rate_amount,...}... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 15-1. Creating a pinned quote template

```
$  mam-create-quote --pin -J vpc.1 -t 86400 --rate
Processors=1.5/s,QualityOfService{Premium}=*1.7
Successfully created 1 quote template with id 17
```

Example 15-2. Creating an unpinned quote template

```
$  mam-create-quote --nopin -t 86400 --rate Disk=2.5/s,License{Matlab}=4/s
Successfully created 1 quote template with id 18
```

Note: Use of the **mam-create-quote** command will not result in a cost estimate or the creation of a usage record. Use **mam-quote** instead if you wish to obtain a quote for usage.

Querying Quotes

To display quote information, use the command **mam-list-quotes**:

mam-list-quotes [[-q] quote_id] [-J instance_name] [-A | -I] [-X, --extension property=value]... [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-m machine_name] [--filter filter_name=filter_value]... [--full] [--show attribute_name,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 15-3. Listing all quotes for user amy on machine colony

```
$ mam-list-quotes -u amy -m colony
Id Amount Pinned Instance UsageRecord StartTime           EndTime             Duration Char
-- ------ ------ -------- ----------- ------------------- ------------------- -------- ----
1   57600 True            242         2017-04-06 12:49:53 2017-04-13 13:49:53 3600     Proc
```

Modifying Quotes

To modify a quote, use the command **mam-modify-quote**:

mam-modify-quote {[-q] quote_id} [-s start_time] [-e end_time] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 15-4. Changing the expiration time of a quote

```
$ mam-modify-quote -e "2017-05-01" 1
Successfully modified 1 quote
```

Deleting Quotes

To delete a quote, use the command **mam-delete-quote**:

mam-delete-quote {-I | {[-q] quote_id}} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 15-5. Deleting a quote

```
$ mam-delete-quote 1
Successfully deleted 1 quote
```

Example 15-6. Purging stale quotes

```
$  mam-delete-quote -I
Successfully deleted 2 quotes
```

Chapter 16. Managing Usage Records

Moab Accounting Manager can track the usage of resources and services on your system, recording the charge and the details of the usage in a usage record. A usage record is created when a resource or service manager requests a guaranteed quote for usage, places a lien for usage, or charges for the usage of an item. Usage records can also be created directly via UsageRecord Create (mam-create-usagerecord). A refund can be invoked to credit a charge amount back to the originating fund. Usage records can also be queried, modified or deleted. By default, a standard user may only query usage records attributed to them.

In a typical use case, a quote might be used to discover how much it would cost to use an item (resource or service) and to verify the user had sufficient access to the item and funds to cover the requested usage. Just before the item is about to be used, a lien (or hold) might be placed against the user's allocated credits for the requested usage. After the usage is complete, a charge for the actual usage can be debited from their fund and the lien removed.

As is the case for other Moab Accounting Manager objects, usage records are highly customizable. One may remove most usage record properties and add new usage record properties. Refer to the section Customizing the UsageRecord Object for examples of customizing usage records.

Creating a Usage Record

In most cases, usage records will be created by the resource management system via the API or with the **mam-quote**, the **mam-reserve** or the **mam-charge** command.

However, it is also possible to create usage records directly using the **mam-create-usagerecord** command:

mam-create-usagerecord {-J instance_name} [-n designated_name] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-Q quality_of_service] [-m machine_name] [-N nodes] [-P processors] [-C cpu_time] [-M memory] [-D disk] [-E energy] [-F "{\"feature_name\":feature_count,...}"] [-R "{\"resource_name\":resource_count,...}"] [-L "{\"license_name\":license_count,...}"] [-Z "{\"metric_name\":metric_amount,...}"] [-V "{\"variable_name\":\"variable_value\",...}"] [-W requested_duration] [-t actual_duration] [-s start_time] [-e end_time] [-x exit_code] [--stage lifecycle_stage] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-1. Creating a usage record

```
$  mam-create-usagerecord -u jsmith -a chem -m cluster -X Charge=2468 -P 2 -t
1234 -J PBS.1234.0
Successfully created 1 usage record with id 246
```

Note: Use of the **mam-create-usagerecord** command to record usage will not result in the debiting of a user's allocation. Use **mam-charge** instead if you wish to charge for the usage.

Querying Usage Records

To display usage record information, use the command **mam-list-usagerecords**:

mam-list-usagerecords [[-j] usage_record_id] [-J instance_name_pattern] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-Q quality_of_service] [-m machine_name] [--stage lifecycle_stage] [-X, --extension property=value]... [-s start_time] [-e end_time] [--full] [--show attribute_name,...] [--format csv|raw|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 16-2. Show specific info about usage tallied by amy

```
$ mam-list-usagerecords --show=Type,Instance,Account,Machine,Charge -u amy
Type Instance   Account   Machine Charge
---- ---------- --------- ------- ------
Job  PBS.1234.0 chemistry colony   22212
```

Example 16-3. Show breakdown of charges by account and user

```
$ mam-list-usagerecords --show "GroupBy(Account),GroupBy(User),Sum(Charge)"
Account   User    Charge
--------- ------- ------
biology   bob       5.00
chemistry amy       5.00
chemistry bob       1.00
```

Example 16-4. Show number of jobs per quality of service

```
$ mam-list-usagerecords --show
"Count(Instance)=Jobs,GroupBy(QualityOfService)"
Jobs QualityOfService
---- ----------------
40
1    premimum
9    windfall
```

Example 16-5. Show number of jobs using the bigmem node feature

```
$ mam-list-usagerecords --show "Count(Features{bigmem})"
bigmem
------
147
```

Example 16-6. Show number of matlab licenses used by the chemistry account

```
$ mam-list-usagerecords -a chemistry --show "Sum(Licenses{matlab})"
matlab
------
407
```

Modifying a Usage Record

It is possible to modify a usage record by using the command **mam-modify-usagerecord**:

mam-modify-usagerecord {[-j] usage_record_id | -J instance_name} [-n designated_name] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-Q quality_of_service] [-m machine_name] [-N nodes] [-P processors] [-C cpu_time] [-M memory] [-D disk] [-E energy] [-F "{\"feature_name\":feature_count,...}"] [-R "{\"resource_name\":resource_count,...}"] [-L "{\"license_name\":license_count,...}"] [-Z "{\"metric_name\":metric_amount,...}"] [-V "{\"variable_name\":\"variable_value\",...}"] [-W requested_duration] [-t actual_duration] [-s start_time] [-e end_time] [-x exit_code] [--stage lifecycle_stage] [-d description] [-X, --extension property=value]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-7. Changing a usage record

```
$ mam-modify-usagerecord -Q HalfPrice -X Charge=1234 -d "Benchmark" -J
PBS.1234.0
Successfully modified 1 usage record
```

> **Note:** Changing a recorded charge in this manner will not change the allocated balance (see Issuing Usage Refunds to refund a charge).

Deleting a Usage Record

To delete a usage record, use the command **mam-delete-usagerecord**:

mam-delete-usagerecord {[-j] usage_record_id | -J instance_name} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-8. Deleting a usage record

```
$ mam-delete-usagerecord -J PBS.1234.0
Successfully deleted 1 usage record
```

Obtaining Usage Quotes

Usage quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed. A breakdown of the charges in the quote can be returned by specifying the --itemize option with the --verbose option.

To request a usage quote, use the command **mam-quote**:

mam-quote [-J instance_name] [[-j] usage_record_id] [-q quote_template_id] [-n designated_name] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization] [-c class_name] [-Q quality_of_service] [-m machine_name] [-N nodes] [-P processors] [-C cpu_time] [-M memory] [-D disk] [-E energy] [-F "{\"feature_name\":feature_count,...}"] [-R "{\"resource_name\":resource_count,...}"] [-L "{\"license_name\":license_count,...}"] [-Z "{\"metric_name\":metric_amount,...}"] [-V "{\"variable_name\":\"variable_value\",...}"] [-W requested_duration] [--stage lifecycle_stage] [-d description] [-X, --extension property=value]... [-zt quote_duration [-G grace_duration]] [-zs quote_start_time] [-z quote_amount] [--cost-only | --guarantee] [--rate charge_rate_name[{charge_rate_value}]=charge_rate_amount,...]... [--hours] [--itemize] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-9. Requesting a quote

```
$  mam-quote -a chemistry -u amy -m colony -P 2 -W 3600
Successfully quoted 7200 credits
```

Example 16-10. Requesting a guaranteed quote

```
$  mam-quote -a chemistry -u amy -m colony -P 16 -W 3600 --guarantee
Successfully quoted 57600 credits with quote id 1 and usage record id 86
```

```
$  mam-list-quotes
Id Amount UsageRecord StartTime           EndTime             Duration Used ChargeRates
-- ------ ----------- ------------------- ------------------- -------- ---- ---------------
1  57600  86          2017-04-06 10:09:58 2017-04-06 11:09:58 3600     0    Processors=1/s
```

Making Usage Reservations

A usage lien can be used to place a hold on the user's funds before usage starts to ensure that the credits will be there when it completes. The replace option may be specified if you want the new lien to replace existing liens of the same instance name (associated with the same usage record). The modify option may be specified to dynamically extend any existing lien with the same instance name with the specified characteristics instead of creating a new one. See Managing Liens for more information about these options.

To create a usage lien use the command **mam-reserve**:

mam-reserve {-J instance_name} [[-j] usage_record_id] [-q quote_id] [-n designated_name] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization] [-c class_name] [-Q quality_of_service] [-m machine_name] [-N nodes] [-P processors] [-C cpu_time] [-M memory] [-D disk] [-E energy] [-F "{\"feature_name\":feature_count,...}"] [-R "{\"resource_name\":resource_count,...}"] [-L "{\"license_name\":license_count,...}"] [-Z "{\"metric_name\":metric_amount,...}"] [-V "{\"variable_name\":\"variable_value\",...}"] [-W requested_duration] [-s start_time] [--stage lifecycle_stage] [-d description] [-X, --extension property=value]... [-zt lien_duration] [-zs lien_start_time [-G grace_duration]] [-z lien_amount] [--modify | --replace] [--rate charge_rate_name[{charge_rate_value}]=charge_rate_amount,...]... [--hours] [--itemize] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-11. Creating a lien

```
$  mam-reserve -J PBS.1234.0 -a chemistry -u amy -m colony -P 2 -W 3600
Successfully reserved 7200 credits with lien id 37 for instance PBS.1234.0 and created usag
```

Charging for Usage

A usage charge debits the appropriate allocations based on the attributes of the usage. The charge is calculated based on factors including the resources and services used, the usage time, and other quality-based factors (see Managing Charge Rates). By default, any liens associated with the charge will be removed. The incremental option may be specified if you want associated liens to be reduced instead of removed. If a usage record already exists for the instance being charged it will be updated with the data properties passed in with the charge request, otherwise a new usage record will be created.

A quote id can be specified to use a previously quoted set of charge rates. This will also ensure the charge will update the usage record instantiated with the quote. A lien id can be specified to help match up a charge with its lien (this may assist in deleting the correct lien if instance ids are not unique). This will also ensure the charge will update the usage record that may have been instantiated by the lien.

Although, by default, Moab Accounting Manager will calculate the charge for the usage using its default charge rates or using the charge rates saved by a referenced quote or quote template, it is possible to specify override charge rates via the rate option. Alternatively, it is possible to designate an externally calculated charge by specifying the charge amount with the Charge option (-z option to mam-charge).

To charge for a usage use the command **mam-charge**:

mam-charge {-J instance_name} [[-j] usage_record_id] [-n designated_name] [-q quote_id] [-l lien_id] [-T usage_record_type] [-u user_name] [-g group_name] [-a account_name] [-o organization_name] [-c class_name] [-Q quality_of_service] [-m machine_name] [-N nodes] [-P processors] [-C cpu_time] [-M memory] [-D disk] [-E energy] [-F "{\"feature_name\":feature_count,...}"] [-R "{\"resource_name\":resource_count,...}"] [-L "{\"license_name\":license_count,...}"] [-Z "{\"metric_name\":metric_amount,...}"] [-V "{\"variable_name\":\"variable_value\",...}"] [-W requested_duration] [-t actual_duration] [-s start_time] [-e end_time] [-x exit_code] [--stage lifecycle_stage] [-d description] [-X, --extension property=value]... [-zt charge_duration] [-zs charge_start_time] [-z charge_amount] [-f fund_id] [--incremental] [--rate charge_rate_name[{charge_rate_value}]=charge_rate_amount,...]... [--hours] [--itemize] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-12. Issuing a usage charge

```
$  mam-charge -J PBS.1234.0 -a chemistry -u amy -m colony -P 2 -t 1234
```

```
Successfully charged 2468 credits for instance PBS.1234.0
1 lien was removed
```

Issuing Usage Refunds

A charged amount can be credited back in part or in whole by issuing a usage refund. This action attempts to lookup the referenced usage record to ensure that the refund does not exceed the original charge and so that the charge entry can be updated. If multiple matches are found (such as the case when instance names (such as job ids) are non-unique), this command will return the list of matched usage records with unique ids so that the correct usage record can be specified for the refund.

To issue a refund for a usage charge, use the command **mam-refund**:

mam-refund {-J instance_name | [-j] usage_record_id} [-z refund_amount] [-i allocation_id] [-d description] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 16-13. Issuing a usage refund

```
$  mam-refund -J PBS.1234.0
```

```
Successfully refunded 19744 credits for instance PBS.1234.0
```

Customizing the UsageRecord Object

The stock usage record object can be customized with the attributes you want to track in your use cases. The chapter on Customizing Objects goes into some detail on the customization syntax. However, since this may be a common requirement, this section will provide a few examples on modifying, adding and deleting usage record attributes and getting them to be tracked and show up in queries.

Important: In order to be able to track a custom usage record property, your resource manager must pass this property to MAM at the creation of the usage record (e.g. with the charge) or subsequent modification. If you are using the Moab Workload Manager, see "Accounting Properties Reported to Moab Accounting Manager" in the Moab Administor Guide for the list of usage record properties included with the Job and Reservation charges.

Usage record discriminators are those properties which are considered primary differentiators between usage, lien and quote records. Usage record discriminators are used in the dynamic web portal as filters for the listing, modification and deletion of usage records, liens and quotes. The default usage record discriminators are Type, User, Group, Account, Organization, Class, QualityOfService and Machine. Any new attributes added to the usage record object will become usage record discriminators. Removing a discriminator attribute from the usage record object will necessarily remove it as a usage record discriminator as well. It will be necessary to log out and back in after adding or removing a discriminator in order for it to be reflected in the web GUI.

Example 16-14. Adding an Application Field (and discriminator)

Let's say you would like to track the application run by the job. First, you would add Application as an Attribute of the UsageRecord Object.

```
$  mam-shell Attribute Create Object=UsageRecord Name=Application
DataType=String
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$  vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Application,Charge,Stage,User,Group,Account,Organizatio
```

If you want to filter the usage records by Application, (such as listing all usage records associated with the specified application), use the -X (or --extension) option in mam-list-usagerecords.

```
$  mam-list-usagerecords -X Application=foo
--show=Type,Instance,Charge,User,Application
Type Instance   Charge User Application
---- ---------- ------ ---- -----------
Job  PBS.1234.0  19744 amy  foo
```

You could also use Application as the basis of a ChargeRate. See the Managing Charge Rates chapter for details on how to do this.

Although the initial step above allows the application value to be tracked in the usage record, it is also possible to add it as an attribute of the Transaction table so that it will be automatically populated from actions having assignments, conditions, options and data values referring to the Application.

```
$  mam-shell Attribute Create Object=Transaction Name=Application
DataType=String
```

```
Successfully created 1 attribute
```

Additionally, the mam-statement client command can show Application as one of its discriminators (which are Account, User and Machine by default) in its debit detail. These statement discriminators are specified by the --show argument to mam-statement and can be configured with the statement.show configuration parameter in mam-client.conf.

Example 16-15. Tracking the user-specified job name

The following example demonstrates how to add a Name attribute to the usage record.

```
$ mam-shell Attribute Create Object=UsageRecord Name=Name DataType=String
Description="\"User-Specified Name\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Name,Charge,Stage,User,Group,Account,Organization,Class
```

Example 16-16. Tracking Accelerator Usage

The following example demonstrates how to track hardware accelerator usage (e.g. GPUs and/or MICs) within the usage record.

To track GPUs:

```
$ mam-shell Attribute Create Object=UsageRecord Name=GPUs DataType=Integer
Description="\"Number of GPUs Allocated\""
Successfully created 1 attribute
```

To track MICs:

```
$ mam-shell Attribute Create Object=UsageRecord Name=MICs DataType=Integer
Description="\"Number of MICs Allocated\""
Successfully created 1 attribute
```

If you want the new attributes to show up in mam-list-usagerecords, you must add them to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

Once you have added them to the usage record, you may charge for them by adding an affiliated charge rate. See the example entitled "Charging for GPUs and/or MICs" within the Creating Charge Rates section for sample steps on how to do this.

Example 16-17. Tracking energy used

The following example demonstrates how to add an Energy attribute to the usage record.

```
$ mam-shell Attribute Create Object=UsageRecord Name=Energy DataType=Float
Description="\"Energy Used\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

Example 16-18. Tracking Node Features

The following example demonstrates how to add a Features attribute to the usage record.

```
$ mam-shell Attribute Create Object=UsageRecord Name=Features
DataType=JSON:Integer Description="\"Node Features Allocated\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

Example 16-19. Tracking NUMA Properties

The following example demonstrates how to track NUMA properties (e.g. Sockets, NumaNodes, Cores, Threads) with the usage record.

```
$ mam-shell Attribute Create Object=UsageRecord Name=Sockets
DataType=Integer Description="\"Number of NUMA Sockets Allocated\""
Successfully created 1 attribute

$ mam-shell Attribute Create Object=UsageRecord Name=NumaNodes
DataType=Integer Description="\"Number of NUMA Nodes Allocated\""
Successfully created 1 attribute
```

```
$ mam-shell Attribute Create Object=UsageRecord Name=Cores DataType=Integer
Description="\"Number of NUMA Cores Allocated\""
Successfully created 1 attribute

$ mam-shell Attribute Create Object=UsageRecord Name=Threads
DataType=Integer Description="\"Number of NUMA Threads Allocated\""
Successfully created 1 attribute
```

If you want the new attributes to show up in mam-list-usagerecords, you must add them to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

Example 16-20. Adding a ProcessorEquivalents Field

The following example demonstrates how to track processor equivalents with the usage record. For a description of what processor equivalents mean, see the "PE" topic in the "Scheduling Environment" section of the Moab Workload Manager Administrator Guide.

```
$ mam-shell Attribute Create Object=UsageRecord Name=ProcessorEquivalents
DataType=Float Description="\"Processor Equivalents\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$ vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

You could also use ProcessorEquivalents as the basis of a ChargeRate. See the Managing Charge Rates chapter for details on how to do this.

Example 16-21. Adding a BlockedProcessors Field

The following example demonstrates how to track blocked processors with the usage record.

```
$ mam-shell Attribute Create Object=UsageRecord Name=BlockedProcessors
DataType=Integer Description="\"Number of Processors Blocked by the Job\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$  vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

You could also use BlockedProcessors as the basis of a ChargeRate. See the Managing Charge Rates chapter for details on how to do this.

Example 16-22. Tracking queued duration

The following example demonstrates how to track the effective duration that a job was in the idle state by adding a QueueDuration attribute to the usage record.

```
$  mam-shell Attribute Create Object=UsageRecord Name=QueueDuration
DataType=Integer Description="\"Queue Duration\""
Successfully created 1 attribute
```

If you want the new attribute to show up in mam-list-usagerecords, you must add it to the usagerecord.show string in mam-client.conf.

```
$  vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,Qual
```

Example 16-23. Enabling Reservation Statistics

The following example demonstrates how to track reservation statistics with the usage record. In this example, we will show how to track reserved processor seconds and idle processor seconds within a reservation.

```
$  mam-shell Attribute Create Object=UsageRecord
Name=ReservedProcessorSeconds DataType=Integer Description="\"Reserved
Processor Seconds\""
Successfully created 1 attribute

$  mam-shell Attribute Create Object=UsageRecord Name=IdleProcessorSeconds
DataType=Integer Description="\"Unused Processor Seconds\""
Successfully created 1 attribute
```

Once you are able to track idle processor seconds, you may use the IdleProcessorSeconds property to charge for the unused cycles in a reservation. See the example entitled "Charging for unused cycles in reservations" within the Creating Charge Rates section for sample steps on how to do this.

Example 16-24. Removing the UsageRecord Class Field

Let's say you you were not interested in tracking the class. First, you would delete Class as an Attribute of the UsageRecord Object.

```
$  mam-shell Attribute Delete Object==UsageRecord Name==Class
Successfully deleted 1 attribute
```

Next, we need to make sure mam-list-usagerecords doesn't try to list the class.

```
$  vi /opt/mam/etc/mam-client.conf
usagerecord.show = Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,QualityOfS
```

If the attribute you want to delete is also an attribute in the Transaction table, you could delete it from there as well.

Example 16-25. Setting VM as the default Usage Record Type

As installed, the usage record type defaults to "Job". The default value can be set to NULL if there should be no default value, or to any other default value. This example will demonstrate how to set the default usage record type to "VM".

```
$  mam-shell Attribute Modify Object==UsageRecord Name==Type DefaultValue=VM
Successfully modified 1 attribute
```

Usage Record Property Verification

If a usage record property has an object associated with it, you may want to verify that when that usage record property is specified in a scheduling action (Charge, Reserve, Quote), that it verifies that that property is a valid instance of its object type. You can apply a simple verification to a usage record property by setting the property's Values attribute to an '@' sign followed by the name of the object.

Example 16-26. Ensure that an organization specified in a charge actually exists

```
$  mam-shell Attribute Modify Object==UsageRecord Name==Organization
Values=@Organization
Successfully modified 1 attribute
```

See Managing Attributes for more information about setting the Values attribute.

Usage Record Property Defaults

It is possible to set defaults for usage record properties when they are not specified in the usage data for a charge, lien or quote. There are two cases which must be considered -- when the property has an object associated with it and when the property does not.

If a property does not have an object associated with it, simply set the DefaultValue attribute for the property's UsageRecord Attribute object to the desired value.

Example 16-27. Setting a system-wide simple default class of batch for usage functions

```
$  mam-shell Attribute Modify Object==UsageRecord Name==Class
DefaultValue=batch
Successfully modified 1 attribute
```

If a property does have an object associated with it, you will need to both set the DefaultValue attribute for the property's UsageRecord Attribute object to the desired value AND set the DefaultValue attribute for the corresponding object to the desired value.

Example 16-28. Setting a system-wide simple default user of anonymous for usage functions

```
$  mam-shell Attribute Modify Object==UsageRecord Name==User
DefaultValue=anonymous
Successfully modified 1 attribute

$  mam-shell Object Modify Name==User DefaultValue=anonymous
Successfully modified 1 object
```

See Global Object-based Defaults for more information about setting default values for objects.

See Local Attribute-based Defaults for more information about setting default values for attributes.

Usage Record Property Auto-Generation

It is possible for usage record properties which have object definitions to automatically create the referenced objects the first time they are encountered in a usage function (charge, reserve or quote). To do this, the referenced object must be set to AutoGen=True and the Values attribute for the UsageRecord attribute corresponding to the object must be set to a string consisting of the '@' sign followed by the object name.

Example 16-29. Setting the Usage Record Type to auto-generate Items for usage functions

For example, let's assume there were many usage record types that could be charged for (Food, Book, Haircut) and that you had already created an Item object. It would be possible to automatically generate a new Item instance each time a new usage record type was referenced in a charge operation.

```
$  mam-shell Object Modify Name==Item AutoGen=True
Successfully modified 1 object

$  mam-shell Attribute Modify Object==UsageRecord Name==Type Values=@Item
Successfully modified 1 attribute
```

See Object Auto-Generation for more information about the auto-generation of objects.

Usage Record Property Instantiators

It is possible to establish a dynamic correlation between usage record properties in which one usage record property can instantiate another. For example, if a user is specified in a charge but no account is specified then the user's default account should be applied to the fund constraints and logged; or if an account is specified in a charge but not its organization then the organization corresponding to that account should be looked up and applied to the fund constraints and logged. Three usage record property instantiator types are currently supported and are configured by prefixing the property instance's Values foreign object reference with the appropriate characters: Assign if not defined (@?=), Assign if not different (@!=), Assign always (@:=). We shall look at each of these individually and in different terms.

- Applying a correlated default (@?=) -- If property X is specified with the value x in the usage record and property Y is not specified in the usage record and if the object instance referred to by x has a correlated default value of y' for property Y', then y' will be applied as the default value for property Y in the usage record. For example, we could establish the notion of a default account for a user.

  Example 16-30. Establishing a default account for a user

  First we add a DefaultAccount attribute (the name is arbitrary) to the User object and give it a Values property of @?=Account.

  ```
  $  mam-shell Attribute Create Object=User Name=DefaultAccount
  DataType=String Values="\"@?=Account\"" Description="\"Default Account\""
  Successfully created 1 attribute
  ```

  Then we can establish the default account for user scottmo to be chemistry.

  ```
  $  mam-shell User Modify Name==scottmo DefaultAccount=chemistry
  Successfully modified 1 user
  ```

  Subsequently, when a Charge, Reserve or Quote is issued that specifies the User scottmo but does not specify the Account, the chemistry Account will be applied to the charge as if originally specified in the usage record charge data.

- Applying a correlated verification (@!=) -- If property X is specified with the value x in the usage record and property Y is specified with the value y in the usage record and if the object instance referred to by x has a correlated verification value of y' for the property Y' and if y' does not equal y,

then fail with an error message. Additionally, if property X is specified with the value x in the usage record and property Y is not specified in the usage record and if the object instance referred to by x has a correlated verification value of y' for property Y', then y' will be applied as the default value for property Y in the usage record. For example, we could establish a parent-child relationship between organizations and accounts in which explicitly specified incongruities result in a failure.

Example 16-31. Establishing a verification hierarchy with accounts and organizations

First we add a VerifyOrganization attribute (the name is arbitrary) to the Account object and give it a Values property of @!=Organization.

```
$  mam-shell Attribute Create Object=Account Name=VerifyOrganization
DataType=String Values="\"@!=Organization\"" Description="\"Verify
Organization\""
Successfully created 1 attribute
```

Then we can establish the verify organization for account chemistry to be sciences.

```
$  mam-shell Account Modify Name==chemistry VerifyOrganization=sciences
Successfully modified 1 account
```

Subsequently, when a Charge, Reserve or Quote is issued that specifies the Account chemistry and specifies the wrong Organization (e.g. arts), the transaction will fail with an error message. Additionally, when a Charge, Reserve or Quote is issued that specifies the Account chemistry but does not specify the Organization, the Organization sciences will be applied to the charge as if originally specified in the usage record charge data.

- Applying a correlated override (@:=) -- If property X is specified with the value x in the usage record and if the object instance referred to by x has a correlated override value of y' for property Y', then y' will be applied as the override value for property Y in the usage record. For example, we could establish a parent-child relationship between organizations and accounts in which explicitly specified incongruities are silently overridden with the value from the child.

Example 16-32. Establishing an override hierarchy with accounts and organizations

First we add an OverrideOrganization attribute (the name is arbitrary) to the Account object and give it a Values property of @:=Organization.

```
$  mam-shell Attribute Create Object=Account Name=OverrideOrganization
DataType=String Values="\"@:=Organization\"" Description="\"Override
Organization\""
Successfully created 1 attribute
```

Then we can establish the override organization for Account chemistry to be sciences.

```
$  mam-shell Account Modify Name==chemistry OverrideOrganization=sciences
Successfully modified 1 account
```

Subsequently, when a Charge, Reserve or Quote is issued that specifies the Account chemistry and specifies either the wrong Organization (e.g. arts) or no Organization, the Organization sciences will be silently applied to the charge as if originally specified in the usage record charge data.

Chapter 17. Managing Itemized Charges

The itemized charge table provides an ability to display the components of a composite charge in a line item format. Each charge transaction will write the components of its charge into the charge record so that you can get a line-item breakdown of each charge for usage including the names, values, rates, scaling factors, charge amounts and details listed for each component of the charge. This capability is enabled by setting charge.itemization = true in the mam-server.conf (it is false by default).

Itemized charges may only be queried. They are created automatically in charge transactions and there are no command line clients to change or remove them.

Additionally, an itemize option can be specified for quotes, liens and charges to include an itemized charge breakdown in the response data instead of a single line with the amount.

Querying Itemized Charges

To display itemized charge information, use the command **mam-list-itemizedcharges**:

mam-list-itemizedcharges [-j usage_record_id] [-J instance_name] [-n usage_property_name] [-s start_time] [-e end_time] [--full] [--show attribute_name,...] [--format csv|raw|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 17-1. Listing all itemized charge information

```
$  mam-list-itemizedcharges
```

```
UsageRecord Instance Name       Value Duration Rate      ScalingFactor Amount CreationTime
----------- -------- ---------- ----- -------- --------- ------------- ------ -------------
24          job.1    Storage    100   86400    1.157e-07 1                  1 2017-04-05 17
25          job.2    Processors 4     86400    5.787e-05 1                 20 2017-04-05 17
25          job.2    Memory     4096  86400    1.13e-08  1                  4 2017-04-05 17
26          job.3    Processors 1     86400    5.787e-05 1                  5 2017-04-05 17
26          job.3    Memory     1024  86400    1.13e-08  1                  1 2017-04-05 17
```

Displaying Itemized Charges for a Transaction

In addition to the itemized charge table, Moab Accounting Manager captures the itemized charges for usage record charges, liens and guaranteed quotes in the details of the transaction. The itemized charges show the details for the formula used to calculate the charge for the transaction. To display the itemized charges for a scheduling transaction, parse the details from the command **mam-list-transactions --full -A Charge|Reserve|Quote**:

Example 17-2. Extract the itemized charges for a job charge

```
$  mam-list-transactions -A Charge -J job.2 -q --show Details | perl -pe
's/.*(ItemizedCharges[^,]*).*/\1/'
```

```
ItemizedCharges:=4 [Processors] * 5.787e-05 [ChargeRate{Processors}] * 86400 [Duration] + 4
```

Chapter 18. Managing Charge Rates

Charge rates establish how much to charge for usage. A charge rate consists of its name, an optional value and the amount. Charge rates are applied when usage properties matching the charge rate names are found in the usage data. In order for a charge rate of a given name to be applied, a usage record attribute of the same name must exist. For example, a charge rate having the name Processors will be applied if Processors is defined as a Usage Record attribute and the incoming usage data for the charge request contains a property called Processors that matches the value specified in the charge rate.

There are two basic types of charge rates: Name-valued charge rates and Numeric-valued charge rates.

- Name-valued charge rates are used for usage properties that take strings for values (e.g. QualityOfService=premium or Account=chemistry). The charge rate that is applied will be determined by a lookup of the usage property value to see if there is a matching charge rate value. A default rate may be specified by creating a name-valued charge rate with an empty charge rate value. Multiple values may be assigned to the same rate via separate charge rate definitions or by combining the values in a single charge rate value separated by commas.
- Numeric-valued charge rates are used for usage properties that take numbers for values (e.g. Processors=2 or CPUTime=12.67). The charge rate amount that is applied will be multiplied by the usage property value. The charge rate value is commonly left blank to be taken as the default rate for the full range of usage property values. A particular value may also be specified as the charge rate value which means that that rate will only be used if the usage property value exactly matches the charge rate value. A half-bounded expression may be used by specifying a less than or greater than sign with an optional equal sign, followed by the number. For example, the charge rate value <=4 would match a usage property value of x if x <= 4. A charge rate value may also be specified as a range (of the form <number>[-<number>]). For example, the range 1-4 would be match a usage property value of x if 1 <= x <= 4. If you need to be more specific about the boundedness of the ranges, you may replace the dash with a less than sign with an optional equal sign on either side of it to indicate whether the endpoints are included. For example, the range 1<4 would match if 1 < x < 4, 1=<4 would match if 1 <= x < 4, 1<=4 would match if 1 < x <=4 and 1=<=4 would match if 1 <= x <= 4. So you might use ranges like 1=<2, 2=<4, 4=<8, and >=8. Multiple values or value ranges having the same charge rate may be specified in a single expression separated by commas.

A charge rate amount may have an operation modifier which dictates the way the rate is factored into the charge calculation. For example, consumption-based charge rates or usage fees will often be additive in nature while quality-based charge rates may be multiplicative. The additive charge rates may be further distinguished by whether they should be added before or after the multiplicative charge rates are applied. The charge formula can be represented in the following form: (Σ(Pre-Additive Rates)*Π(Multiplicative Rates))+Σ(Post-Additive Rates). Thus, there are three operation modifiers: Pre-Additive, Multiplicative and Post-Additive.

- Pre-additive modifiers are applied to charge rates that should be added together before any charge multipliers are applied. A pre-additive modifier is specified by prepending a plus sign '+' to the charge rate amount. Since pre-additive is the most commonly specified operation modifier, a charge rate amount without an operation modifier will be assumed to be pre-additive by default.

- Multiplicative modifiers are applied to charge rates that should be multiplied together with other multiplicative charge rates and with the sum of the pre-additive charge rates. A multiplicative modifier is specified by prepending an asterisk '*' to the charge rate amount.
- Post-additive modifiers are applied to charge rates that should be added together after any charge multipliers are applied. A post-additive modifier is specified by appending a plus sign '+' to the charge rate amount.

A pre-additive charge rate may have a time-based modifier which dictates that the charge should be multiplied by the amount of time the feature was used. For example, it is common for the processor charge to be multiplied by the the amount of time the processors were used. A time-based modifier is specified by appending a forward slash '/' to the charge rate amount, followed by one of the following time designators: s (per-second), m (per-minute), h (per-hour), d (per-day), W (per-week), M (per-month), Y (per-year). As an example, a per-hour time-based modifier is specified by appending '/h' to the charge rate amount and will cause a charge to be multiplied by the number of hours the feature was used. Technically, a rate with a time-based modifier will be multiplied by the number of seconds the feature was used, then divided by the number of seconds corresponding to the time designator (e.g. 3600), and will ultimately be rounded to the number of decimal places in the currency precision.

A pre-additive charge rate may have a divisor modifier which dictates that the charge should be divided by the specified integer. A divisor modifier is specified by appending a forward slash '/' to the charge rate amount, followed by an integer number. A divisor modifier can be used in lieu of expressing a small decimal fraction charge rate such as when converting a value from MegaBytes to GigaBytes. If a divisor modifier is used in conjunction with a time-based modifier, the divisor modifier must precede the time-based modifier.

A charge rate may have one or more conditions which dictates additional qualifications that must be met in order for the charge rate to be applied. A condition is specified by prepending <propertyName>=<propertyValue> followed by a question mark '?' to the value field of the charge rate. If you want Processors to apply a special charge rate (e.g. .5/s) for user amy, the charge rate value should consist of the string "User=amy?". Additionally, you may combine charge rate conditions with either a pipe symbol '|' for or, or an ampersand symbol '&' for and. For example, User=amy|User=dave? or User=amy&Project=chemistry?. You may not combine ands and ors in the same charge rate value.

Creating Charge Rates

To create a new charge rate, use the command **mam-create-chargerate**:

mam-create-chargerate {[-n] charge_rate_name} [-x charge_rate_value] {-z charge_rate_amount} [-d description] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

> **Important:** If a usage record attribute does not exist for the name of the charge rate you are creating, you must first create the corresponding usage record property. See Customizing the UsageRecord Object.

Example 18-1. Charging for requested memory

For consumable resources, we use a time-based modifier (e.g. '/s') to multiply the memory by the duration the resource was used (in this case, seconds). We also divide the result by 1024 since Moab reports memory in MegaBytes but we want to charge for GigaBytes.

```
$ mam-create-chargerate -n Memory -z 1/1024/s -d "1 credit per requested GigaByte of memory per second"
Successfully created 1 charge rate
```

Example 18-2. Charging for GPUs (and/or MICs)

If you intend to have the accelerator charge multiplied by the amount of time that was used, use the appropriate time modifier. Use the name "GPUs" if charging for GPUs, and the name "MICs" if charging for MICs (of course you may create separate charge rates for each if both are present in your system).

```
$ mam-create-chargerate -n GPUs -z 1/s -d "1 credit per GPU-second"
Successfully created 1 charge rate
```

> **Note:** Remember that it is essential to first add GPUs and/or MICs as a custom usage record property before being able to charge by it. See Customizing the UsageRecord Object for an example of how to do this.

Example 18-3. Charging for cpu time

Since cpu time already incorporates the element of time in its value, we do not need to include a time-based modifier in the charge rate.

```
$ mam-create-chargerate -n CPUTime -z 1 -d "1 credit per utilized cpu-second"
Successfully created 1 charge rate
```

Example 18-4. Charging for blocked processors (jobs only)

It is possible to charge for blocked processors rather than allocated processors. For example, all of the processors in an entire node may be blocked by a job using a node-exclusivity policy (e.g. a node access policy of "single-job") even though a lesser number of processors were actually requested and allocated to the job.

```
$ mam-create-chargerate -n BlockedProcessors -x Type=Job? -z 1/s -d "1 credit per blocked processor second"
Successfully created 1 charge rate
```

Note: Remember that it is essential to first add BlockedProcessors as a custom usage record property before being able to charge by it. See Customizing the UsageRecord Object for an example of how to do this.

Example 18-5. Charging for processor equivalents

Some sites may wish to charge for processor equivalents rather than allocated processors. Processor equivalents scale the allocated processors by the most constrained consumable resource (e.g. memory or cpu).

```
$  mam-create-chargerate -n ProcessorEquivalents -z 1/s -d "1 credit per
processor equivalent per second"
Successfully created 1 charge rate
```

Note: Remember that it is essential to first add ProcessorEquivalents as a custom usage record property before being able to charge by it. See Customizing the UsageRecord Object for an example of how to do this.

Example 18-6. Charging for the unused cycles in reservations

If your resource manager supports it, and if configured to do so, you may charge for the unused cycles in administrative or standing reservations. If using Moab Workload Manager, you must must first enable reservation charging as described in the "Charging for Reserved Resources" topic in the "Reservation Policies" section of the Moab Administrator Guide.

It will also be necessary to add the required reservation statistics to the usage record object (e.g. IdleProcessorSeconds and ReservedProcessorSeconds). See the example entitled "Enabling Reservation Statistics" in the Customizing the Usage Record Object for steps on how to do this.

After adding the necessary usage record attributes, you must create a charge rate that charges for the unused cycles in the reservation. The following charge rate will charge for processor seconds that were not blocked by jobs running within the reservation.

```
$  mam-create-chargerate -n IdleProcessorSeconds -x 'Type=Reservation?' -z 1
-d "1 credit per unused processor second in reservations"
Successfully created 1 charge rate
```

If also charging for jobs, it is recommended that you charge jobs for the blocked processors with a condition of 'Type=Job?' since this is the best counterpart to the IdleProcessorSeconds metric which charges for unblocked processors. See the example "Charging for blocked processors" for how to do this.

Example 18-7. Creating a name-valued pre-additive charge rate

```
$ mam-create-chargerate -n License -x Matlab -z 5
Successfully created 1 charge rate
```

Example 18-8. Creating a numeric-valued multiplicative charge rate

```
$ mam-create-chargerate -n Discount -z *1
Successfully created 1 charge rate
```

Example 18-9. Charging for quality of service

We want to multiply the resource charge by a value that depends on the quality of service applied to the job. Thus we must create a set of name-valued multiplicative charge rates with a default value.

```
$ mam-create-chargerate -n QualityOfService -x Premium -z *2
Successfully created 1 charge rate

$ mam-create-chargerate -n QualityOfService -x BottomFeeder -z *0.5
Successfully created 1 charge rate

$ mam-create-chargerate -n QualityOfService -z *1
Successfully created 1 charge rate
```

Example 18-10. Charging for licenses

```
$ mam-create-chargerate -n Licenses -x matlab -z +20
Successfully created 1 charge rate
```

Example 18-11. Charging for generic resources

```
$ mam-create-chargerate -n Resources -x graphics -z 5
Successfully created 1 charge rate
```

Example 18-12. Charging for job variables

```
$ mam-create-chargerate -n Variables -x foo:bar -z 10
Successfully created 1 charge rate
```

Example 18-13. Creating a numeric-valued post-additive charge rate

```
$ mam-create-chargerate -n Shipping -z 25+
Successfully created 1 charge rate
```

Example 18-14. Creating a name-valued post-additive charge rate

```
$ mam-create-chargerate -n Zone -x Asia -z 200+
Successfully created 1 charge rate
```

Example 18-15. Creating a couple of conditional numeric-valued pre-additive charge rates

```
$ mam-create-chargerate -n Disk -x User=dave? -z 0.2/s
Successfully created 1 charge rate

$ mam-create-chargerate -n Disk -x User=mike? -z 0.5/s
Successfully created 1 charge rate
```

Example 18-16. Creating some numeric-valued pre-additive charge rate ranges and a default

```
$ mam-create-chargerate -n Processors -x 1-4 -z 2/s
Successfully created 1 charge rate

$ mam-create-chargerate -n Processors -x 5-8 -z 1.5/s
Successfully created 1 charge rate

$ mam-create-chargerate -n Processors -z 1/s
Successfully created 1 charge rate
```

Example 18-17. Creating some numeric-valued pre-additive charge rate ranges for floating point values (without time-based modifiers)

```
$ mam-create-chargerate -n Power -x '<2' -z 0.005
Successfully created 1 charge rate

$ mam-create-chargerate -n Power -x '2=<4' -z 0.004
Successfully created 1 charge rate

$ mam-create-chargerate -n Power -x '>=4' -z 0.003
```

```
Successfully created 1 charge rate
```

Example 18-18. Assigning multiple classes to run for free

```
$ mam-create-chargerate -n Class -x dev,test -z *0
Successfully created 1 charge rate
```

Querying Charge Rates

To display charge rate information, use the command **mam-list-chargerates**:

mam-list-chargerates [[-n] charge_rate_name] [-x charge_rate_value] [--full] [--show attribute_name,...] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 18-19. Listing all charge rates

```
$ mam-list-chargerates
Name             Value        Amount   Description
---------------- ------------ -------- -----------
Class            dev,test     *0
CPUTime                       1
Discount                      *1
Disk             User=dave?   0.2/s
Disk             User=mike?   0.5/s
License          Matlab       5/s
Memory                        1/1024/s
Power            <2           0.005
Power            2=<4         0.004
Power            >=4          0.003
Processors                    1/s
Processors       1-4          2/s
Processors       5-8          1.5/s
QualityOfService              *1
QualityOfService BottomFeeder *0.5
QualityOfService Premium      *2
Shipping                      25+
Zone             Asia         200+
```

Modifying Charge Rates

To modify a charge rate, use the command **mam-modify-chargerate**:

mam-modify-chargerate {[-n] charge_rate_name} [-x charge_rate_value] [-z charge_rate_amount] [-d description] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 18-20. Changing a charge rate

```
$ mam-modify-chargerate -n License -x Matlab -z 4/s
Successfully modified 1 charge rate
```

Deleting Charge Rates

To delete a charge rate, use the command **mam-delete-chargerate**:

mam-delete-chargerate {[-n] charge_rate_name} [-x charge_rate_value] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 18-21. Deleting a charge rate

```
$ mam-delete-chargerate -n Memory
Successfully deleted 1 charge rate
```

Chapter 19. Managing Transactions

Moab Accounting Manager logs all modifying transactions in a detailed transaction journal (queries are not recorded). Previous transactions can be queried but not modified or deleted. By default, a standard user may only query transactions performed by them.

Querying Transactions

To display transaction information, use the command **mam-list-transactions**:

mam-list-transactions [[-T] transaction_id] [-R request_id] [-O object] [-A action] [-k primary_key_value] [-U actor] [-f fund_id] [-i allocation_id] [-u user_name] [-a account_name] [-m machine_name] [-j usage_record_id] [-J instance_name] [-s start_time] [-e end_time] [-X, --extension property=value]... [--full] [--show attribute_name,...] [--format csv|raw|standard] [--hours] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 19-1. List all deposits made in 2017

```
$  mam-list-transactions -A Deposit -s 2017-01-01 -e 2018-01-01
```

Example 19-2. List refund totals broken down by fund

```
$  mam-list-transactions -A Refund --show "Sum(Amount),GroupBy(Fund)"
```

Example 19-3. List usage and charge totals broken down by account and user

```
$  mam-list-transactions -A Charge --show
"GroupBy(Account),GroupBy(User),Sum(ProcHours),Sum(Amount)=Charged"
```

Example 19-4. List every transaction performed by amy since the beginning of 2017

```
$  mam-list-transactions -U amy -s 2017-01-01
```

Example 19-5. List all transactions related to job moab.1

```
$  mam-list-transactions -J moab.1
```

Example 19-6. List all transactions affecting charge rates

```
$  mam-list-transactions -O ChargeRate
```

Customizing the Transaction Object

The transaction record as natively defined can be customized with the attributes you want to track in your use cases. It is possible to add additional attributes to the Transaction table so that it will be automatically populated from actions having assignments, conditions, options and data values referring to the attribute.

Transaction discriminators are those properties which are considered primary differentiators between transaction records (besides the metadata differentiators of object, action and instance). Transaction discriminators are used in the dynamic web portal as filters for the listing of transaction records. Any new attributes added to the Transaction object will become transaction discriminators. Removing a discriminator attribute from the transaction object will necessarily remove it as a transaction discriminator as well. It will be necessary to log out and back in after adding or removing a discriminator in order for it to be reflected in the web GUI.

Example 19-7. Adding an Organization field to the Transaction record (which also makes it a discriminator)

```
$  mam-shell Attribute Create Object=Transaction Name=Organization
DataType=String
Successfully created 1 attribute
```

Chapter 20. Managing Events

Moab Accounting Manager has an internal event scheduler that can be configured to execute Moab Accounting Manager actions at a designated time in the future or on a periodic basis. Valid actions on an event include Create, Query, Fire, Modify, Refresh and Delete. Event attributes include Id, FireCommand, ArmTime, FireTime, RearmPeriod, EndTime, Notify, RearmOnFailure, FailureCommand, CatchUp and Description.

There are two server configuration parameters which affect event scheduling: event.scheduler which specifies whether the event scheduler is enabled or not (it is disabled by default) and event.pollinterval which is the period in minutes that the event scheduler uses to fire events. The poll interval must divide evenly into the number of minutes in a day (1440).

> **Important:** You must set event.scheduler = true in mam-server.conf and restart the MAM server in order for events to fire.

The command(s) to be fired by an event are expressed in a serialized form of the request identical to the syntax used in the interactive control program (mam-shell). There are two commands that can be configured in an event: the FireCommand which is the command to be executed when the event is fired, and the FailureCommand which is the command to be executed if the fired command results in an unsuccessful response status. The FireTime is the target time for the event to be triggered by the event scheduler. The actual fire time may be dependent on the state of the server and will be recorded in the CreationTime property of the corresponding "Event Fire" Transaction. An event may also be fired manually with the Event Fire action.

The RearmPeriod is a time period expression specifying when the event will be rearmed. This period expression is of the form: "<period>[[@<instant>][~|^]|!]". The period is expressed as an integer number followed by a designator of minute(s), hour(s), day(s), week(s), month(s) or year(s). For example, the period might be 1 day, 2 hours, or 5 minutes. The optional Instant locks the period to a specific instant within the time period such as 1 day @ hour 12 or 1 month @ day 3. The modifiers indicate whether the time period should be relative to now (!), or relative to the start of this (~) designator (month or minute, etc.), or relative to the start of the first (^) designator (month or minute, etc.). For example, assuming the FireTime was 7:15, if you specified "4 hours !" as the rearm period it would be rearmed at 11:15, if you specified "4 hours ~" as the rearm period it would be rearmed at 11:00, and if you specified "4 hours ^" as the rearm period it would be rearmed at 8:00.

The ArmTime is the time the event was last armed or fired. This field is used as a reference time to be able to derive how long the event has been waiting to happen. This field will be initially set to mark the moment the first FireTime is set and updated thereafter to indicate the last time the event was fired. In the case where an event does not have a FireTime set, this field may be set manually and used in a similar manner. If we consider the time between event firings as "laps", this could be thought of as the Lap Start Time. If the RearmOnFailure boolean is set to False, the event will not be rearmed if the command was unsuccessful. If set to True, the event will be evaluated for rearming even if the command response has a status of Failure. The standard default is False. If the CatchUp boolean is set to True and the server was down during the time this event should have fired, the event scheduler will attempt to make up for the past due events by progressively firing them (rearming based on previous arm time) until catching up to the present. The actions will still show as having occurred in the present rather than in the past. If set to

False, and the server is brought back up after an outage, the event scheduler will still fire immediately for a past due event, but it will only fire once and then rearm relative to the current time.

A Notification method can be specified via the Notify parameter and is of the form: [+-=][<delivery_method>:] [<recipient>][,[+-=][<delivery_method>:][<recipient>]]*. If the term is a -, the notification is sent only on failure. If the term is a +, the notification is sent only on success. Otherwise the notification is always sent. There can be multiple notify expressions separated by a comma. All applicable notifications will be sent. See the chapter on <link linkend="Notifications">Managing Notifications</link> for more information about delivery method and recipient.

Creating Events

To create a new event, use the command **mam-create-event**:

mam-create-event --fire-command fire_command [-s fire_time] [-e end_time] [--rearm-period rearm_period] [--rearm-on-failure True\|(False)] [--failure-commmand failure_command] [--notify notification_url] [--catch-up (True)\|False] [-d description] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 20-1. Creating an automatic allocation renewal event

```
$  mam-create-event --fire-command "Fund Reset" -s "2018-01-01"
--rearm-period "3 months^"
Successfully created 1 event
```

> **Note:** You must set event.scheduler = true in mam-server.conf and restart the MAM server in order for events to fire.

Querying Events

To display event information, use the command **mam-list-events**:

mam-list-events [[-E] event_id] [-s start_time] [-e end_time] [--full] [--show attribute_name,...] [--format csv\|raw\|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 20-2. Listing all events

```
$  mam-list-events
Id FireCommand FireTime   ArmTime             RearmPeriod EndTime Notify RearmOnFailure Fai
-- ----------- ---------- ------------------- ----------- ------- ------ -------------- ---
1  Fund Reset  2018-01-01 2017-11-09 10:31:28 3 months^                  False
```

Modifying Events

To modify an event, use the command **mam-modify-event**:

mam-modify-event {[-E] event_id} [--fire-command fire_command] [-s fire_time] [-e end_time] [--rearm-period rearm_period] [--rearm-on-failure True\(False)] [--failure-commmand failure_command] [--notify notification_url] [--catch-up (True)\False] [-d description] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 20-3. Changing an events's rearm period to be monthly

```
$  mam-modify-event --rearm-period "1 month" 1
Successfully modified 1 event
```

Deleting Events

To delete an event, use the command **mam-delete-event**:

mam-delete-event {[-E] event_id} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 20-4. Deleting an event

```
$  mam-delete-event 1
Successfully deleted 1 event
```

Chapter 21. Managing Notifications

When event commands are executed (asynchronously), the success or failure of the operation is communicated back to the initiator via a notification. When an event is created, you may specify the Notify option which will associate a notification method with the event. Currently there is only one DeliveryMethod implemented which is Store. With the Store delivery method, command response information is stored as instances of the Notification object. These messages can later be retrieved by the initiator via a Notification Query. Payments can also route a notification method down to their associated events via a Notify option.

The notification attributes include Id (autogenerated), Type, Event, Status, Code, Message, Key, Recipient, EndTime and CreationTime. Stored notifications can be queried on any of these conditions. The notification type distinguishes what type of command resulted in the notification (Fire or Failure). The notification key is the value of the primary key of the object instance that the command acted on (e.g. the Payment Id). The recipient could be a user name or any tag that identifies the intended reader for the notification. The Notification Query supports a Delete option, which if set to True, will delete the notifications after they have been queried. Additionally, stored notification have an EndTime after which they are automatically deleted by MAM. The Notification actions include Send, Refresh, Create, Query, Delete and Modify.

There are two server configuration parameters which affect notifications: notification.deliverymethod which dictates which deliverymethod is used by default if unspecified and notification.duration which defines how long notifications stick around if the Store delivery method is used.

Querying Notifications

To display notification information, use the command **mam-list-notifications**:

mam-list-notifications [[-N] notification_id] [-E event_id] [-T notification_type] [-k primary_key_value] [-u recipient] [-x status] [-s start_time] [-e end_time] [--delete] [--full] [--show attribute_name,...] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 21-1. Listing all failure notifications

```
$  mam-list-notifications -x Failure
Id Event Type Status  Code Message
-- ----- ---- ------- ---- ------------------------------------------------------------------
4  20    Fire Failure 782  Payment Begin failed starting payment: Failed creating payment s
```

Deleting Notifications

To delete a notification, use the command **mam-delete-notification**:

mam-delete-notification {[-N] notification_id} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 21-2. Deleting a notification

```
$ mam-delete-notification 4
Successfully deleted 1 notification
```

Example 21-3. Deleting all successful notifications

To delete many notification, query them with the --delete option:

```
$ mam-list-notifications -x Success --delete
Id Event Type Status  Code Message
-- ----- ---- ------- ---- ------------------------------------------------------------
1  11    Fire Success 000  Payment Begin: Successfully charged 10 credits for instance Moab
2  14    Fire Success 000  Payment Begin: Successfully charged 10 credits for instance Moab
3  17    Fire Success 000  Payment Begin: Successfully charged 10 credits for instance Moab
Successfully deleted 3 notifications
```

Chapter 22. Managing Roles

Moab Accounting Manager uses instance-level role based access controls to determine what users can perform what functions. Named roles are created, actions are associated with the roles, and users are assigned to these roles.

The actions for a role consist of a set of tuples of object, action and instance permitted by the role. In other words, each role action defines an object (whether specific or ANY), the action that can be taken on that object (whether specific or ANY) and the instance of the object that action can be taken on (whether specific or ANY).

In the base configuration, there are three default roles: SystemAdmin, Anonymous and OVERRIDE. Other configurations, such as the bank configuration, add additional roles. Roles can be added as desired. The three base roles are required for proper function of Moab Accounting Manager and should not be deleted. By default, the SystemAdmin role can perform any action on any object. This role is usually assigned to the super user. The Anonymous role is intended to define the actions available to your standard unprivileged user. This may include the ability to set your password, query certain public objects and modify objects that belong to you (implemented via the OVERRIDE role). The OVERRIDE role is a special role type that defines those actions that should use special business logic intrinsic to the routine that handles that object and action. For example, in the bank configuration, the OVERRIDE logic for the Account Query routine will only allow the standard user to see information about accounts for which he or she is a member. A given user's privileges will be the superset of the actions of all roles that apply to that user.

The instance indicates which specific instances of the object the action can be performed on. There are several special instance types that can be used in certain situations. The ANY instance is supported by all objects and permits the specified action on all instances of the specified object. The SELF instance applies to the user's own instance if the object is User, or to objects that have a User attribute associated with the user. The MEMBERS instance applies to objects for which the user is a direct member. The ADMIN instance applies to objects for which the user is designated as an administrator. Unless otherwise specified, the instance will default to a value of ANY.

Creating Roles

To create a new role, use the command **mam-create-role**. Users and actions may be associated with the role at creation time. When assigning actions to a role, the object, action and instance must be specified in the form shown. Multiple actions or users may be specified for the role.

mam-create-role {[-r] role_name} [-d description] [-u user_name,...]... [-A object_name->action_name[{instance_name}],...]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 22-1. Creating a Manager role

```
$  mam-create-role -r Manager -d "Manages Roles and Responsibilities"
Successfully created 1 role
```

Querying Roles

To display the role information, use the command **mam-list-roles**:

mam-list-roles [[-r] role_name] [--full] [--show attribute_name,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--version] [--about]

Example 22-2. Listing all roles along with users and descriptions

```
$ mam-list-roles --show=Name,Users,Description
Name         Users   Description
------------ ------- ------------------------------------------------
AccountAdmin         Can update or view an account they are admin for
Anonymous    ANY     Things that can be done by anybody
OVERRIDE     ANY     A custom authorization method will be invoked
Scheduler    root    Scheduler relevant Transactions
SystemAdmin  scottmo Can update or view any object
UserServices         User Services
```

Example 22-3. Listing information about the scheduler role

```
$ mam-list-roles --long Scheduler
Name      Users Actions                   Description
--------- ----- ------------------------- ------------------------------
Scheduler root  UsageRecord->Create(ANY)  Scheduler relevant Transactions
                UsageRecord->Quote(ANY)
                UsageRecord->Reserve(ANY)
                UsageRecord->Charge(ANY)
                Lien->Delete(ANY)
```

Modifying Roles

To modify a role, use the command **mam-modify-role**:

mam-modify-role {[-r] role_name} [-d description] [--add-user(s) user_name,...]... [--add-action(s) object_name->action_name[{instance_name}],...]... [--del-user(s) user_name,...]... [--del-action(s) object_name->action_name[{instance_name}],...]... [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Users may be added to a role or removed from a role. Actions also may be added to a role or removed from a role. When specifying actions, the instance will default to a value of ANY.

Example 22-4. Adding a user to a role

Let's add dave to our new Manager role:

```
$ mam-modify-role --add-user dave -r Manager
```

```
Successfully added 1 user
```

Example 22-5. Associating an action with a role

Allow the Manager to change role responsibilities

```
$  mam-modify-role --add-action "RoleAction->ANY" Manager
Successfully added 1 action
```

Deleting Roles

To delete a role, use the command **mam-delete-role**:

mam-delete-role {[-r] role_name} [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 22-6. Deleting the Manager role

```
$  mam-delete-role Manager
Successfully deleted 1 role and 2 associations
```

Chapter 23. Managing Passwords

Passwords must be established for each user who wishes to use the web-based GUI. Passwords must be at least eight characters and are stored in encrypted form. A mam-set-password command line client exists to aid a user or administrator in setting or changing a password. Other operations (deleting or listing password entries) must be performed using the interactive control program (mam-shell). By default, a standard user may only set or change their own password. A system administrator may set or change any user's password.

> **Important:** Because Moab Accounting Manager caches password information for faster responsiveness, it will be necessary to restart the server after running mam-set-password for the GUI to accept that password change.

Setting Passwords

To set a new password, use the command **mam-set-password**. If the user name is not specified via an option or as the unique argument, then the invoking user will be taken as the user whose password will be set. The invoker will be prompted for the new password.

mam-set-password [[-u] user_name] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Example 23-1. Setting a password

```
$  mam-set-password amy
Enter your new password:
Successfully created 1 password
```

Querying Passwords

To display password information, use the command **mam-shell Password Query**:

mam-shell Password Query [Show:=<*"Field1,Field2,..."*>] [User==<*User Name*>] [ShowUsage:=True]

Example 23-2. List the users who have set passwords

```
$  mam-shell Password Query Show:=User
User
----
amy
mam
```

Deleting Passwords

To delete a password, use the command **mam-shell Password Delete**:

mam-shell Password Delete User==<*User Name*>

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to Moab Accounting Manager objects. Misuse of this command could result in the inadvertent deletion of all passwords.

Example 23-3. Deleting a password

```
$  mam-shell Password Delete User==amy
User Password
---- --------------------------------
amy  HZYzwD20o1XIE/gxRYyFKP2sumkCluHm
Successfully deleted 1 password
```

Chapter 24. Using the MAM Shell (mam-shell)

mam-shell is an interactive control program that can access all of the advanced functionality in Moab Accounting Manager.

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. Inadvertant mistakes could result in modifications that are very difficult to reverse.

Usage

mam-shell commands can be invoked directly from the command line as arguments, or read from stdin (interactively or redirected from a file).

mam-shell [--format csv|raw|standard] [--debug] [--site site_name] [--help] [--man] [--quiet] [--verbose] [--version] [--about] [command]

Example 24-1. Specifying the command as direct arguments

```
$ mam-shell System Query
Name                    Version Description
----------------------- ------- ------------------
Moab Accounting Manager 8.0.0   Commercial Release
```

Example 24-2. Using the interactive prompt

```
$ mam-shell
mam> System Query
Name                    Version Description
----------------------- ------- ------------------
Moab Accounting Manager 8.0.0   Commercial Release

mam> quit
```

Example 24-3. Reading commands from a file

```
$ cat >commands.mam <<EOF
System Query
quit
EOF
```

```
$ mam-shell <commands.mam
Name                    Version Description
----------------------- ------- ------------------
Moab Accounting Manager 8.0.0   Commercial Release
```

Command Syntax

mam-shell commands are of the form:

<Object> [*=<Alias>*] [,*<Object>* [*=<Alias>*]...] *<Action>* [[*<Conjunction>*] [*<Open_Parenthesis>*...] [*<Object>.*] *<Name>* *<Operator>* [*<Subject>.*] *<Value>* [*<Close_Parenthesis>*...] ...]

The basic form of a command is <Object> <Action> [<Name><Operator><Value>]*. When an action is performed on more than one object, such as in a multi-object query, the objects are specified in a comma-separated list. Commands may accept zero or more predicates which may function as fields to return, conditions, update values, processing options, etc. Predicates, in their simplest form, are expressed as Name, Operator, Value tuples. Predicates may be combined via conjunctions with grouping specified with parentheses. When performing multi-object queries, names and values may need to be associated with their respective objects.

Valid conjunctions include:

&&
: and

||
: or

&!
: and not

|!
: or not

Open parentheses may be any number of literal open parentheses '('.

Name is the name of the condition, assignment, or option. When performing a multi-object query, an attribute name may need to be prepended by its associated object separated by a period (<object>.<attribute>). When specifying a partial condition, the name will consist of the attribute followed by the part enclosed in curly braces (<attribute>{<part>}).

Valid operators include:

==
: equals

<
: less than

>
: greater than

<=
: less than or equal to

>=
: greater than or equal to

!=
: not equal to

~
: matches

=
: is assigned

+=
: is incremented by

-=
: is decremented by

:=
: option

:!
: not option

Value is the value of the selection list, condition, assignment, or option. When performing a multi-object query, a value may need to be prepended by its associated object (called the subject) separated by a period.

Close parentheses may be any number of literal closing parentheses ')'.

Valid Objects

To list the objects available for use in mam-shell commands, issue the mam-shell command: Object Query

Example 24-4. Listing all objects

```
mam> Object Query Show:="Sort(Name)"
Name
-------------------
Account
AccountUser
Action
Allocation
Attribute
ChargeRate
Constraint
Fund
FundFund
Lien
LienAllocation
Object
Organization
Password
Quote
QuoteChargeRate
Role
RoleAction
RoleUser
System
Transaction
UsageRecord
User
```

Valid Actions for an Object

To list the actions that can be performed on an object, use the mam-shell command: Action Query

Example 24-5. Listing all actions associated with the Fund object

```
mam> Action Query Object==Fund Show:="Sort(Name)"
Name
--------
Create
Delete
Deposit
Modify
Query
```

```
Transfer
Undelete
Withdraw
```

Valid Predicates for an Object and Action

By appending the option "ShowUsage:=True" to a command, the syntax of the command is returned, expressed in SSSRMAP XML Message Format.

Example 24-6. Show the usage for Allocation Query

```
mam> Allocation Query ShowUsage:=True
<Request action="Query">
    <Object>Allocation<Object>
    [<Get name="Id" [op="Sort|Tros|Count|GroupBy|Max|Min"]></Get>]
    [<Get name="Fund" [op="Sort|Tros|Count|GroupBy|Max|Min"]></Get>]
    [<Get name="StartTime" [op="Sort|Tros|Count|GroupBy|Max|Min"]></Get>]
    [<Get name="EndTime" [op="Sort|Tros|Count|GroupBy|Max|Min"]></Get>]
    [<Get name="Amount" [op="Sort|Tros|Count|GroupBy|Max|Min|Sum|Average"]></Get>]
    [<Get name="CreditLimit" [op="Sort|Tros|Count|GroupBy|Max|Min|Sum|Average"]></Get>]
    [<Get name="InitialDeposit" [op="Sort|Tros|Count|GroupBy|Max|Min|Sum|Average"]></Get>]
    [<Get name="Allocated" [op="Sort|Tros|Count|GroupBy|Max|Min|Sum|Average"]></Get>]
    [<Get name="Active" [op="Sort|Tros|Count|GroupBy"]></Get>]
    [<Get name="Description" [op="Sort|Tros|Count|GroupBy|Max|Min"]></Get>]
    [<Where name="Id" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="<Integer
    [<Where name="Fund" [op="EQ|NE|GT|GE|LT|LE|Match|NotMatch (EQ)"] [conj="And|Or (And)"]
    [<Where name="StartTime" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="<I
    [<Where name="EndTime" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="<Int
    [<Where name="Amount" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="<Inte
    [<Where name="CreditLimit" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="
    [<Where name="InitialDeposit" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [grou
    [<Where name="Allocated" [op="EQ|NE|GT|GE|LT|LE (EQ)"] [conj="And|Or (And)"] [group="<I
    [<Where name="Active" [op="EQ|NE (EQ)"] [conj="And|Or (And)"] [group="<Integer Number>
    [<Where name="Description" [op="EQ|NE|GT|GE|LT|LE|Match|NotMatch (EQ)"] [conj="And|Or (
    [<Option name="Filter">{Filter_Name}={Filter_Value}</Option>]*
    [<Option name="FilterType">ExactMatch|Exclusive|NonExclusive (NonExclusive)</Option>]
    [<Option name="IncludeAncestors">True|False (False)</Option>]
    [<Option name="Time">YYYY-MM-DD[ hh:mm:ss]</Option>]
    [<Option name="Unique">True|False (False)</Option>]
    [<Option name="ChunkSize">{Integer Number}</Option>]
    [<Option name="Limit">{Integer Number}</Option>]
    [<Option name="Offset">{Integer Number}</Option>]
    [<Option name="ShowHidden">True|False (False)</Option>]
    [<Option name="ShowUsage">True|False (False)</Option>]
<Request>
```

Common Options

There are a number of options that may be specified for all commands. These options include:
ShowUsage

ShowUsage

This option may be included with any command to cause the command to return a usage message in SSSRMAP XML Message Format.

Common Actions Available for most Objects

There are a number of actions that are available for most objects. These actions include Query, Create, Modify, Delete and Undelete. Commands involving these actions inherit some common structure unique to the action type.

Query Action

The Query action is used to query objects. It accepts selections that describe the attributes (fields) to return (including aggregation operations on those attributes), conditions that select which objects to return the attributes for, and other options unique to queries.

Selections

Selections use the Show option to specify a list of the attributes to return for the selected object. If selections are not specified, a default set of attributes (defaulting to those not marked as hidden) will be returned.

Name = Show
Op = :=
Value = "selection1,selection2,selection3,..."

Aggregation operators may be applied to attributes by enclosing the target attribute in parenthesis and prepending the name of the desired operator. The aggregation operators that can be applied depend on the datatype of the attribute

Valid selection operators include:

Sort Ascending sort
Tros Descending sort
Count Count
Max Maximum value
Min Minimum value
Average Average value
Sum Sum
GroupBy Group other aggregations by this attribute

Partial values may be requested for complex (multi-valued) attributes. Partial values are specified in the form: <attribute>{<

Additionally, aliases can be applied to selections so that columns can be renamed as desired. Aliases are expressed by adding "=<Alias>" to the target attribute name (and after the trailing parenthesis ofthe aggregation if specified).

Examples:
Allocation Query Show:="GroupBy(Fund),Sum(Amount)=Total"
UsageRecord Query Show:="GroupBy(Account),Sum(Licenses{matlab})=Matlab_Licenses_Used"

Conditions

Conditions are used to select which objects the action is to be performed on.

Name = Name of the attribute to be tested
Op = conditional operator
Value = The object or value against which the attribute is tested

When expressing a condition that is part of a multi-object join, the name may consist of the attribute prepended with the object and a period (<object>.<attribute>).

When expressing a condition for a part of a complex (multi-valued) attribute, the name will consist of the attribute followed by the part in curly braces (<attribute>{<part>}).

Valid condition operators include:

== Equal to
!= Not equal to
< Less than
> Greater than
<= Less than or equal to
>= Greater than or equal to
~ Matches
!~ Does not match

Matching uses the wildcards * and ? (equivalent to SQL % and _ respectively) in a manner similar to file globbing. * matches zero or more unspecified characters and ? matches exactly one unspecified character. For example mscf* matches objects having the specified attributes whose values start with the letters mscf, while mscf? matche jects having the specified attributes whose values start with mscf and have a total of exactly five characters.

Examples:
UsageRecord Query Application~"NWChem*"
UsageRecord Query Metrics{temperature}>100.0

Options

Options indicate processing options that affect the result.

Name = Name of the option
Op = :=
Value = Value of the option

Valid options for query actions include:

ShowHidden:=True|False (False) Includes hidden attributes in the result
Time:="YYYY-MM-DD[hh:mm:ss]" Run the command as if it were the specified time
Unique:=True|False (False) Display only unique results (like DISTINCT in SQL)
ChunkSize={Integer Number}* Number of records to return per page
Limit:={Integer Number}* Limit the results to the number of objects specified
Offset:={Integer Number}* Number of records to skip before starting to return data

> **Note:** * Note that the query may return records in a nondeterministic order (different order in different requests) unless ordering is specified using the Sort operator. It is important to specify the sort order when using the ChunkSize, Limit or Offset options. It is especially important to specify the sort order when using the ChunkSize option, otherwise the same records might be returned in different chunks, while other records may not be returned at all. This is not a bug; it is a consequence of the behavior of the underlying database which does not promise to deliver the results of a query in any particular order unless ORDER BY is used to constrain the order.

Example 24-7. Return the number of inactive liens

```
mam> Lien Query EndTime<now Show:="Count(Id)"
Id
--
8
```

Create Action

The Create action is used to create a new object. It accepts assignments that describe the values of the attributes to be set.

Assignments

Assignments specify values to be assigned to attributes in the new object.

Name = Name of the attribute being assigned a value
Op = = (is assigned)
Value = The new value being assigned to the attribute

Example 24-8. Add a new account member

```
mam> AccountUser Create Account=chemistry Name=scottmo
Account   Name    Active Admin
--------- ------- ------ -----
```

```
chemistry scottmo True   False
Successfully created 1 accountUser
```

Modify Action

The Modify action is used to modify existing objects. It accepts conditions that select which objects will be modified and assignments that describe the values of the attributes to be set.

Assignments

Assignments specify values to be assigned to attributes in the selected objects.

Name = Name of the attribute being assigned a value
Op = assignment operators {=, +=, -=}
Value = The value being assigned to the attribute

Valid assignment operators include:

= is assigned
+= is incremented by
-= is decremented by

Conditions

Conditions are used to select which objects the action is to be performed on.

Name = Name of the attribute to be tested
Op = conditional operator
Value = The object or value against which the attribute is tested

Valid condition operators include:

== Equal to
!= Not equal to
< Less than
> Greater than
<= Less than or equal to
>= Greater than or equal to
~ Matches
!~ Does not match

Matching uses the wildcards * and ? (equivalent to SQL % and _ respectively) in a manner similar to file globbing. * matches zero or more unspecified characters and ? matches exactly one unspecified character. For example mscf* matches objects having the specified attributes whose values start with the letters mscf, while mscf? matches objects having the specified attributes whose values start with mscf and have a total of exactly five characters.

Example 24-9. Change/set scottmo's phone number and email address

```
mam> User Modify Name==scottmo PhoneNumber="(509) 376-2204"
EmailAddress="scottmo@adaptivecomputing.com"
Name    Active CommonName        PhoneNumber    EmailAddress          DefaultAccount Descri
------- ------ ----------------- -------------- --------------------- -------------- ------
scottmo True   Jackson, Scott M. (509) 376-2204 scottmo@adaptivecomputing.com

Successfully modified 1 user
```

Example 24-10. Extend all liens against account chemistry by 10 days

```
mam> Lien Modify EndTime+=864000 Instance=="job.1"
Id Fund Amount Instance   UsageRecord User Account   Machine EndTime             Descriptio
-- ---- ------ ---------- ----------- ---- --------- ------- ------------------- ----------
1  2    57600  PBS.1234.0 1           amy  chemistry colony  2017-04-16 10:47:30
Successfully modified 1 lien
```

Delete Action

The Delete action is used to delete objects. It accepts conditions that select which objects are to be deleted.

Conditions

Conditions are used to select which objects the action is to be performed on.

Name = Name of the attribute to be tested
Op = conditional operator
Value = The object or value against which the attribute is tested

Valid condition operators include:

== Equal to
!= Not equal to
< Less than
> Greater than
<= Less than or equal to
>= Greater than or equal to
~ Matches
!~ Does not match

Matching uses the wildcards * and ? (equivalent to SQL % and _ respectively) in a manner similar to file globbing. * matches zero or more unspecified characters and ? matches exactly one unspecified character. For example mscf* matches objects having the specified attributes whose values start with the letters mscf, while mscf? matche

jects having the specified attributes whose values start with mscf and have a total of exactly five characters.

Example 24-11. Get rid of the pesky Jacksons

```
mam> User Delete CommonName~"Jackson*"
Name    Active CommonName        PhoneNumber    EmailAddress          DefaultAccount Descri
------- ------ ----------------- -------------- --------------------- -------------- ------
scottmo True   Jackson, Scott M. (509) 376-2204 scottmo@adaptivecomputing.com

Successfully deleted 1 user and 1 association
```

Undelete Action

The Undelete action is used to restore deleted objects. It accepts conditions that select which objects are to be undeleted.

Conditions

Conditions are used to select which objects the action is to be performed on.

Name = Name of the attribute to be tested
Op = conditional operator
Value = The object or value against which the attribute is tested

Valid condition operators include:

== Equal to
!= Not equal to
< Less than
> Greater than
<= Less than or equal to
>= Greater than or equal to
~ Matches
!~ Does not match

Matching uses the wildcards * and ? (equivalent to SQL % and _ respectively) in a manner similar to file globbing. * matches zero or more unspecified characters and ? matches exactly one unspecified character. For example mscf* matches objects having the specified attributes whose values start with the letters mscf, while mscf? matche
jects having the specified attributes whose values start with mscf and have a total of exactly five characters.

Example 24-12. Let's resurrect the deleted users that were active

```
mam> User Undelete Active==True
Name    Active CommonName        PhoneNumber    EmailAddress                  DefaultAccoun
------- ------ ----------------- -------------- ----------------------------- -------------
scottmo True   Jackson, Scott M. (509) 376-2204 scottmo@adaptivecomputing.com

Successfully undeleted 1 user and 1 association
```

Multi-Object Queries

mam-shell supports multi-object queries (table joins). Multiple objects are specified via a comma-separated list and attributes need to be prefixed by the associated object.

Example 24-13. Print the sums for active balance and allocated amounts grouped by account

```
mam> Allocation,Constraint Query
Show:="GroupBy(Constraint.Value)=Account,Sum(Allocation.Amount)=Balance,Sum(Allocation.Allo
Constraint.Fund==Allocation.Fund Constraint.Name==Account
Allocation.Active==True
Account   Balance   Allocation
--------- --------- ----------
biology   193651124 360000000
chemistry 296167659 360000000
```

Example 24-14. Show all actions within amy's privileges

```
mam> RoleUser,RoleAction Query
Show:="RoleAction.Object,RoleAction.Name=Action"
RoleUser.Role==RoleAction.Role && ( RoleUser.Name==amy || RoleUser.Name==ANY
) Unique:=True
Object                Action
--------------------- -------
Account               Query
AccountUser           Query
Action                Query
Allocation            Query
Attribute             Query
ChargeRate            Query
Constraint            Query
Fund                  Query
FundFund              Query
Lien                  Query
LienAllocation        Query
```

```
Object                Query
Organization          Query
Password              ANY
Quote                 Query
QuoteChargeRate       Query
Role                  Query
RoleAction            Query
RoleUser              Query
System                Query
Transaction           Query
UsageRecord           Query
User                  Query
```

Note: Although the forgoing was a good example of a join request, it should be understood that it is not a straightforward way to determine the full extent of a user's privileges. Some of the actions may be tied to specific object instances and many of them are associated with an override method which may not actually permit the user access to any instances of the object. Using Show:="RoleUser.Role,RoleUser.Name=User,RoleAction.Object,RoleAction.Name=Action,RoleAction.Instance" may be revealing in this regard. See the chapter on Roles for more information about managing roles.

Chapter 25. Customizing Objects

Moab Accounting Manager provides the ability to dynamically create new objects or customize or delete existing objects through the interactive control program (mam-shell).

Note: The object customizations described in this chapter will be noticeable in subsequent mam-shell queries (and in the web GUI after a fresh login). Client commands may need to be modified to properly interact with changed objects or attributes.

Caution

The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. Inadvertent mistakes could result in modifications that are very difficult to reverse.

Managing Objects

In Moab Accounting Manager, Objects correspond to tables in the repository which have Attributes (such as Name and Color) and Actions (such as Query and Modify). A specific instance of an object is described as an Instance and has Properties (the specific values of the attributes for that object). The instance is uniquely referred to via its primary key(s) (such as its Name or Id).

An object must have a name and may have a description. An object may be set to auto-generate its instances when first seen (see Object Auto-Generation) and/or a default value may be designated for the object (see Object Default Values).

Objects may reference other objects. If a single instance of an object references only a single instance of another object (for example, a usage record may only have one user), then it is sufficient for the first object to have an attribute field for the second object (the UsageRecord object has an attribute called User). However, if there may be a many-to-many relationship between objects (for example, an account may have multiple users and a user may belong to multiple accounts), then it is necessary to maintain a separate object as an association table (e.g. AccountUser). When creating an association object, the object should be given an appropriate name (e.g. AccountUser), it should be marked as an association (Association=True), and an object needs to be designated for the parent (e.g. Account) and the child (e.g. User). The association object itself may have additional attributes that provide qualitative information about the association (i.e. a particular AccountUser association may be active or be an administrator).

Creating a Custom Object

To create a new object, use the command **mam-shell Object Create**. When an object is created, the 5 default actions are automatically created for the object: Create, Delete, Modify, Query and Undelete. A number of default metadata attributes are creates as well: CreationTime, ModificationTime, Deleted, RequestId and TransactionId. These attributes are normally hidden in regular queries.

mam-shell Object Create Name=<*Object Name*>
[AutoGen=True|(False)] [DefaultValue=<*Default Value*>] [Description=<*Description*>] [Association=True|(False)] [Child=<*Child Object*>] [Parent=<*Parent Object*>] [ShowUsage:=True]

Example 25-1. Creating a Node Object

```
$  mam-shell Object Create Name=Node Description="\"Node Information\""
Successfully created 1 object and 5 actions
```

Example 25-2. Add a node name attribute

```
$  mam-shell Attribute Create Object=Node Name=Name DataType=String
PrimaryKey=True
Successfully created 1 attribute
```

Example 25-3. Add a processor count attribute

```
$  mam-shell Attribute Create Object=Node Name=Processors DataType=Integer
Successfully created 1 attribute
```

Querying Objects

To display object information, use the command **mam-shell Object Query**:

mam-shell Object Query [Name==<*Object Name*>] [Show:=*Name*,*AutoGen*,*DefaultValue*,*Description*,*Association*,*Parent*,*Child*] [ShowUsage:=True]

Example 25-4. List Information for the Node Object

```
$  mam-shell Object Query Name==Node
Name Association Parent Child DefaultValue AutoGen Description
---- ----------- ------ ----- ------------ ------- ----------------
Node False                                 False   Node Information
```

Modifying an Object

It is possible to modify an object by using the command **mam-shell Object Modify**:

mam-shell Object Modify Name==<*Object Name*> [AutoGen=True|False] [DefaultValue=<*Default Value*>] [Description=<*Description*>] [Association=True|(False)] [Child=<*Child Object*>] [Parent=<*Parent Object*>] [ShowUsage:=True]

Example 25-5. Changing the Node object's description

```
$  mam-shell Object Modify Name==Node Description="\"Host Information\""
Successfully modified 1 object
```

Deleting an Object

To delete an object, use the command **mam-shell Object Delete**. When an object is deleted, all associated attributes, actions and other associations are automatically deleted as well.

mam-shell Object Delete Name==<*Object Name*> [ShowUsage:=True]

Example 25-6. Deleting the Node Object

```
$  mam-shell Object Delete Name==Node
Successfully deleted 1 object
```

> **Note:** This is a very dangerous operation and could result in the deletion of all object definitions requiring database repair. The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. Be sure to specify conditions for the object you want to delete.

Object Auto-Generation

It is possible to have object instances be automatically generated the first time they are referenced in designated contexts. For example, you might want a user be auto-generated when newly added to an account. You could have an organization auto-generated when specified as the default for a user. You could have a cost-center be auto-generated when referenced in a usage record. To do this, the referenced object must be set to AutoGen=True and the Values property for the attribute that you want to trigger the auto-generation must be set to a string consisting of the '@' sign followed by the object name.

Example 25-7. Auto-generate an account's organization

For example, let's assume that your accounts belong to specific organizations that you may want to run a report against but you don't want to define all of the organizations up front. It would be possible to

automatically generate a new organization instance each time an undefined organization is specified for a account.

```
$  mam-shell Object Modify Name==Organization AutoGen=True
Successfully modified 1 object

$  mam-shell Attribute Modify Object==Account Name==Organization
Values=@Organization
Successfully modified 1 attribute
```

See Usage Record Property Auto-Generation for a discussion of auto-generating objects referenced in usage records.

Global Object-based Defaults

It is possible to set a global default for an object that will be applied to all attributes referencing this object. When a new instance of an object is being created which has an attribute referring to another object via its Values property, if that attribute has not been specified and you want it to default to the global default, you will need to set the DefaultValue attribute for the referenced object to the desired value.

Example 25-8. Setting a system-wide simple default organization called general

```
$  mam-shell Object Modify Name==Organization DefaultValue=general
Successfully modified 1 object
```

Thereafter each (non-association) object which has an attribute with a Values property set to @Organization will default to general if that attribute is not specified. Perhaps we would want the default value to be taken for the organization when a new account is created.

```
$  mam-shell Attribute Modify Object==Account Name==Organization
Values=@Organization
Successfully modified 1 attribute
```

See Local Attribute-based Defaults for more information about setting default values for attributes.

See Usage Record Property Defaults for more information about setting default values for usage record properties.

Managing Attributes

Objects can have any number of fields called Attributes. When an object is first created, a number of attributes are created for the object by default. These are: CreationTime (time the object was first

created), ModificationTime (time the object was last updated), Deleted (whether the object is deleted or not), RequestId (request id that resulted in the last modification of the object), TransactionId (transaction id that resulted in the last modification of the object).

An attribute must have a name and be associated with an object.

An attribute will have a data type which can be one of (AutoGen, Boolean, Currency, Float, Integer, JSON, String, TimeStamp) and defaults to String. A data type of AutoGen means the field will be a primary key of type integer which will assume the next auto-incremented value from the g_key_generator table. TimeStamps are epoch times stored in integer format. Booleans are strings constrained to the values of True or False (or unset). Float is used to store decimal or floating point values. Currency is like Float but may have special business logic for handling currency values. The JSON data type provides support for complex properties and must store a valid JSON value. The current implementation only provides support for simple JSON objects of the form {key:value,...} where key is a double-quoted string and value may be a number or a double-quoted string. One may also use the more nuanced forms (JSON:Integer, JSON:Float, or JSON:String, etc.) to indicate the expectation that the values of the JSON object will be of the designated variety. Using these forms may be useful for clients and web services to render partial queries in the anticipated data type.

An object may have zero or more attributes which are primary keys (PrimaryKey==True), the combination of which are used to uniquely identify an object instance. Moab Accounting Manager will try to ensure that there can only be one object instance with the exact same set of values of its primary keys.

A required attribute (Required==True), must be either specified or be derived via a default value or other dynamic mechanism when the object is created. It can also not be unset.

A fixed attribute (Fixed==True), may not be changed from its initial value.

An attribute may be constrained to certain values via the Values attribute. The values may be constrained to members of a list expressed as a parenthesized comma-delimited list of strings (e.g. Values="(Brazil,China,France,Russia,USA)"). Alternatively, the values may be constrained to be an instance of a particular object type (like a foreign key constraint) by assigning to the Values attribute the name of an object prefixed by the '@' sign (e.g. Values="@Account" -- which would constrain the value of this attribute to be a valid account name). Stronger versions of the @-prefixed object-constrained values may be used in Quote, Reserve and Charge actions to enforce dynamic interactions between usage record properties such as to assign default values if not defined (e.g. Values="@?=Account"), verification values which evoke an error if they differ (e.g. Values="@!=Account"), or designated values which always overwrite the value (e.g. Values="@:=Account"). See Usage Record Property Instantiators for more information.

A default value may be assigned to an attribute via the DefaultValue attribute. When a new instance of an object is created, if a property is not specified for the attribute, the default value will be used.

The Sequence attribute determines which order an object's attributes will be listed in for queries if no selection list is specified in the query. Attributes with smaller sequence numbers will appear before attributes with larger sequence numbers. The Sequence attribute is also used to enforce a proper attribute display ordering in the web GUI.

The Hidden attribute specifies whether an attribute should be shown in a query by default or not. Hidden attributes can be seen in queries by specifying the ShowHidden option with a value of True.

The Description field is a location to describe the meaning of the attribute and is used in the GUI for field descriptions.

Adding an Attribute to an Object

To create a new attribute for an object, use the command **mam-shell Attribute Create**:

mam-shell Attribute Create Object=<*Object Name*> Name=<*Attribute Name*> [DataType=AutoGen|TimeStamp|Boolean|Float|Integer|Currency|(String)] [PrimaryKey=True|(False)] [Required=True|(False)] [Fixed=True|(False)] [Values=<*Foreign Key or List of Values*>] [DefaultValue=<*Default Value*>] [Sequence=<*Integer Number*>] [Hidden=True|(False)] [Description=<*Description*>] [ShowUsage:=True]

Example 25-9. Adding a Country Attribute to User

```
$  mam-shell Attribute Create Object=User Name=Country
Values="\"(Brazil,China,France,Russia,USA)\"" DefaultValue=USA
Successfully created 1 attribute
```

Example 25-10. Tracking Submission Time in Usage records

```
$  mam-shell Attribute Create Object=UsageRecord Name=SubmissionTime
DataType=TimeStamp
Successfully created 1 attribute
```

Querying Attributes

To display attribute information, use the command **mam-shell Attribute Query**:

mam-shell Attribute Query [Object==<*Object Name*>] [Name==<*Attribute Name*>] [Show:=*Object*,*Name*,*DataType*,*PrimaryKey*,*Required*,*Fixed*,*Values*,*DefaultValue*,*Sequence*,*Hidden*:=True] [ShowUsage:=True]

Example 25-11. List the attributes of the Node object

```
$  mam-shell Attribute Query Object==Node
Object Name             DataType  PrimaryKey Required Fixed Values DefaultValue Sequence Hi
------ ---------------- --------- ---------- -------- ----- ------ ------------ -------- --
Node   Processors       Integer   False      False    False                     20       Fa
Node   Name             String    True       True     True                      10       Fa
Node   TransactionId    Integer   False      False    True                      990      Tr
Node   RequestId        Integer   False      False    True                      980      Tr
Node   Deleted          Boolean   False      False    True                      970      Tr
Node   ModificationTime TimeStamp False      False    True                      960      Tr
Node   CreationTime     TimeStamp False      False    True                      950      Tr
```

Modifying an Attribute

To modify an attribute, use the command **mam-shell Attribute Modify**:

mam-shell Attribute Modify Object==<*Object Name*> Name==<*Attribute Name*> [Required=True|(False)] [Fixed=True|(False)] [Values=<*Foreign Key or List of Values*>] [DefaultValue=<*Default Value*>] [Sequence=<*Integer Number*>] [Hidden=<*True|(False)*>] [Description=<*Description*>] [ShowUsage:=True]

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. A mistake with this command could result in the inadvertent modification of all attributes.

Example 25-12. Change User Organization values to not be restricted to the set of organization instances

```
$  mam-shell Attribute Modify Object==User Name==Organization Values=NULL
Successfully modified 1 attribute
```

Removing an Attribute from an Object

To delete an attribute from an object, use the command **mam-shell Attribute Delete**:

mam-shell Attribute Delete Object==<*Object Name*> Name==<*Attribute Name*> [ShowUsage:=True]

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. A mistake with this command could result in the inadvertent deletion of all attributes.

> **Caution**
>
> When using Moab Accounting Manager as an accounting manager, certain objects and attributes are assumed to exist. For example, a call to UsageRecord Charge would fail if you had deleted the Allocation Amount attribute. The Attribute Undelete command might come in useful in such a case.

Example 25-13. Removing the Organization attribute from Account

```
$ mam-shell Attribute Delete Object==Account Name==Organization
Successfully deleted 1 attribute
```

Example 25-14. Perhaps we don't care to track the QualityOfService attribute in a Usage record

```
$ mam-shell Attribute Delete Object==UsageRecord Name==QualityOfService
Successfully deleted 1 attribute
```

Local Attribute-based Defaults

It is possible to set a specific default for an object attribute that will be applied when an instance of that object is created but the attribute is not specified. This type of default is intended for attributes which do not refer to another object or which should vary from the object global default. This default value is assigned to an attribute via the DefaultValue attribute. When a new instance of the associated object is created, if a property is not specified for the attribute, the specified default value will be used. A local attribute default will have precedence over a global object default.

Example 25-15. Setting a default organization just for the account object

```
$ mam-shell Attribute Modify Object==Account Name==Organization
DefaultValue=university
Successfully modified 1 attribute
```

Example 25-16. Setting a default phone for the user object

```
$ mam-shell Attribute Modify Object==User Name==Phone DefaultValue="\"No
Phone\""
Successfully modified 1 attribute
```

See Global Object-based Defaults for more information about setting default values for objects.

See Usage Record Property Defaults for more information about setting default values for usage record properties.

Managing Actions

Moab Accounting Manager defines which actions can be performed by which objects. When an object is first created, five basic actions are created for the object by default. These are: Create, Modify, Query,

Delete and Undelete. Specific code must exist in Moab Accounting Manager modules in order for objects to support additional actions.

An action is uniquely specified by its name and the object it is associated with. An action also has a description and a boolean display attribute which governs whether this action should be displayed in the web GUI or not.

Adding an Action to an Object

To specify that an action is allowed for an object, use the command **mam-shell Action Create**:

mam-shell Action Create Object=<*Object Name*> Name=<*Action Name*> [Display=True|(False)] [Description=<*Description*>] [ShowUsage:=True]

Example 25-17. Adding a Modify Action to Transaction

```
$  mam-shell Action Create Object=Transaction Name=Modify Description=Modify
Successfully created 1 action
```

Querying Actions

To display action information, use the command **mam-shell Action Query**:

mam-shell Action Query [Object==<*Object Name*>] [Name==<*Attribute Name*>] [Show:=*Object*,*Name*,*Display*,*Description*] [ShowUsage:=True]

Example 25-18. List the actions of the Node object

```
$  mam-shell Action Query Object==Node
Object Name     Display Description
------ -------- ------- -----------
Node   Create   False   Create
Node   Delete   False   Delete
Node   Modify   False   Modify
Node   Query    False   Query
Node   Undelete False   Undelete
```

Modifying an Action

To modify an action, use the command **mam-shell Action Modify**:

mam-shell Action Modify Object==<*Object Name*> Name==<*Attribute Name*> [Display=True|(False)] [Description=<*Description*>] [ShowUsage:=True]

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. A mistake with this command could result in the inadvertent modification of all actions.

Example 25-19. Display all Node actions but Undelete in the web GUI

```
$  mam-shell Action Modify Object==Node Name!=Undelete Display=True
Successfully modified 4 actions
```

Removing an Action from an Object

To delete an action from an object, use the command **mam-shell Action Delete**:

mam-shell Action Delete Object==<*Object Name*> Name==<*Attribute Name*> [ShowUsage:=True]

> **Caution**
>
> The mam-shell control program allows you to make powerful and sweeping modifications to many objects with a single command. A mistake with this command could result in the inadvertent deletion of all actions.

> **Caution**
>
> When using Moab Accounting Manager as an accounting manager, certain actions are assumed to exist. Be careful what you delete!

Example 25-20. Do not allow accounts to be deleted

```
$  mam-shell Action Delete Object==Account Name==Delete
Successfully deleted 1 action
```

Examples Creating Custom Objects

Creating a custom object normally involves defining a new object and adding attributes to the object.

Example 25-21. Creating a License object to track license usage and charges.

Invoke the mam-shell control program in interactive mode.

```
$  mam-shell
```

Create the License Object.

```
mam>  Object Create Name=License Description=License
Successfully created 1 object and 5 actions
```

Next we can define its attributes. We'll give each record a unique id (so the record can be more easily modified), a license type that can be one of (Matlab,Mathematica,Compiler,AutoCAD,Oracle), the user who is using it, the start and end time, how many instances of the license were used, and how much was charged.

```
mam>  Attribute Create Object=License Name=Id DataType=AutoGen
PrimaryKey=True Description="Record Id"
Successfully created 1 attribute

mam>  Attribute Create Object=License Name=Type DataType=String Required=True
Values="(Matlab,Mathematica,Compiler,AutoCAD,Oracle)" Fixed=True
Description="License Type"
Successfully created 1 attribute

mam>  Attribute Create Object=License Name=User Required=True Values="@User"
Description="User Name"
Successfully created 1 attribute

mam>  Attribute Create Object=License Name=StartTime DataType=TimeStamp
Description="Start Time"
Successfully created 1 attribute

mam>  Attribute Create Object=License Name=EndTime DataType=TimeStamp
Description="End Time"
Successfully created 1 attribute

mam>  Attribute Create Object=License Name=Count DataType=Integer
Description="Number of Licenses Used"
Successfully created 1 attribute
```

```
mam>  Attribute Create Object=License Name=Charge DataType=Currency
Description="Amount Charged"
Successfully created 1 attribute
```

Finally, since we would like to manage licenses from the web GUI we will set Display=True.

```
mam>  Action Modify Object==License Name!=Undelete Display=True
Successfully modified 4 actions
```

When we are done we can exit the mam-shell prompt.

```
mam>  quit
```

That's about it. Licenses should now be able to be managed via the GUI and mam-shell. The data source will need to use one of the methods of interacting with Moab Accounting Manager (see Interaction Methods) in order to push license record usage info to Moab Accounting Manager.

Apart from being used as an accounting manager, Moab Accounting Manager can be used as a generalized information service. It can be used to manage just about any object-oriented information over the web. For example, Moab Accounting Manager could be used to provide meta-schedulers with machine/user mappings, or node/resource information.

Example 25-22. Using Moab Accounting Manager as a Grid Map File.

Invoke the mam-shell control program in interactive mode.

```
$  mam-shell
```

Create the GridMap Object.

```
mam>  Object Create Name=GridMap Description="Online Grid Map File"
Successfully created 1 object and 5 actions
```

Next, we can define its attributes. Each entry will consist of a userid (which will serve as the primary key) and a required public X.509 certificate.

```
mam>  Attribute Create Object=GridMap Name=User PrimaryKey=True Values=@User
Description="User Name"
Successfully created 1 attribute
```

```
mam>  Attribute Create Object=GridMap Name=Certificate DataType=String
Required=True Description="X.509 Public Key"
Successfully created 1 attribute
```

Exit the mam-shell prompt.

```
mam>  quit
```

From this point, a peer service will need to use one of the methods of interacting with Moab Accounting Manager (see Interaction Methods) in order to query the GridMap information.

Chapter 26. Integration

Integrating with Moab Workload Manager

Moab Workload Manager can be configured to interact with Moab Accounting Manager to track and charge for resources utilized by jobs and reservations. You will need to use Moab HPC Suite -- Enterprise Edition in order to have support for Moab Accounting Manager.

Select an appropriate accounting manager interface type

There are two accounting manager interface types that Moab can use to interact with Moab Accounting Manager: MAM, which makes direct calls to MAM over the SSS wire protocol, and Native, in which customizable scripts are invoked to communicate with Moab Accounting Manager. The MAM accounting manager interface is the default as it is usually faster. The Native accounting manager interface can be used if higher customizability is needed, or if you need to interface with a third party accounting or allocation system. See the section on Accounting Manager Interface Types in the Moab Workload Manager documentation for more information. Choose the accounting manager interface type that is right for your needs and remember it. This information will be used in a later step.

Run Configure --with-am

It may be necessary or advantageous when installing Moab Workload Manager to run configure with certain accounting related options.

Configure Moab to use the Moab Accounting Manager by running ./configure with the applicable options when installing Moab:

- --with-am[=TYPE] -- enable accounting management with the specified accounting manager interface type (mam or native) [mam]
- --with-am-dir=DIR -- uses the specified prefix directory for the accounting manager if installed in a non-default location

The --with-am option specifies the accounting manager interface type that you want to use as either mam, which is the default, or native. Specifying this option will add essential entries into Moab configuration files. Although these entries may be added manually later, this step facilitates configuration by adding parameters appropriate for your selected accounting manager interface type.

Use --with-am-dir to specify the prefix directory for Moab Accounting Manager if it has been installed in a non-default location. This value is used to help the native accounting manager scripts find the Moab Accounting Manager libraries and server connection information.

Example 26-1. Configuring Moab to use the direct accounting manager interface

```
$  ./configure --with-am
```

Edit the Moab Server Configuration File

Add or uncomment the essential AMCFG lines in the moab.cfg file.

Example 26-2. Configuring Moab to use the MAM accounting manager interface

If you are using the direct (MAM) accounting manager interface, at a minimum, you must tell Moab to use AMCFG[] TYPE=MAM. Additionally, if your Moab Accounting Manager server is running on a different host than the Moab Workload Manager server, you must specify the hostname via the AMCFG[] HOST parameter.

```
$  vi /opt/moab/etc/moab.cfg
```

```
AMCFG[mam] TYPE=MAM HOST=localhost
```

Example 26-3. Configuring Moab to use the Native accounting manager interface

If you are using the script (Native) accounting manager interface, at a minimum, you must tell Moab to use AMCFG[] TYPE=NATIVE. Moab Workload Manager will default to using a set of stock scripts to interact with Moab Accounting Manager.

```
$  vi /opt/moab/etc/moab.cfg
```

```
AMCFG[mam] TYPE=NATIVE
```

Edit the Moab Private Configuration File

If you have chosen to use the direct MAM accounting manager interface type, you will need to configure Moab to have Moab Accounting Manager's symmetric key for secure authentication. This step is not necessary when using the Native accounting manager interface type since the secret key can be securely derived from Moab Accounting Manager and used via the connection libraries.

Example 26-4. Configuring Moab to communicate securely with Moab Accounting Manager

Add or uncomment a CLIENTCFG[AM:mam] KEY parameter line in moab-private.cfg. Copy the token.value parameter in /opt/mam/etc/mam-site.conf into the KEY value in /opt/moab/etc/moab-private.cfg.

```
#  vi /opt/moab/etc/moab-private.cfg
```

```
CLIENTCFG[AM:mam] KEY=UiW7EihzKyUyVQg6dKirDhV3
```

Restart Moab Workload Manager

Restart Moab in order for the configuration changes to take effect.

Example 26-5. Restarting Moab

For RedHat-6-based and SUSE-11-based systems:

```
#  service moab restart
```

For RedHat-7-based, SUSE-12-based and Fedora systems:

```
#  systemctl restart moab.service
```

Integrating with Moab Web Services

Moab Web Services can be configured to interact with Moab Accounting Manager in order to be able to perform RESTful web service queries against accounting objects in MAM.

Edit the MWS HPC Configuration File

Uncomment and set the following parameters in /opt/mws/etc/mws.d/mws-config-hpc.groovy:

- mam.secretKey: Set to the value of the token.value parameter in /opt/mam/etc/mam-site.conf
- mam.server: Set to the hostname of the MAM servef
- mam.port: Set to the port of the MAM server (defaults to 7112)

Example 26-6. Configuring MWS to communicate with MAM

```
$  vi /opt/mws/etc/mws.d/mws-config-hpc.groovy
mam.secretKey = "UiW7EihzKyUyVQg6dKirDhV3"
mam.server = "localhost"
mam.port = 7112
```

Restart Moab Web Services

Restart tomcat in order for the MWS configuration changes to take effect.

Example 26-7. Restarting MWS

```
#  service tomcat6 restart
```

Methods of interacting with Moab Accounting Manager

There are essentially five ways of interacting with Moab Accounting Manager. Let's consider a simple usage charge in each of the different ways.

Using the appropriate command-line client

From inside a script, or by invoking a system command, you can use a command line client (one of the "g" commands in the bin directory).

Example 26-8. To issue a charge at the completion of job usage, you could use mam-charge:

```
mam-charge -J Moab.1234 -a chemistry -u amy -m colony -P 2 -t 3600
```

Using the interactive control program

The interactive control program, mam-shell, will issue a charge for a job expressed in xml.

Example 26-9. To issue a charge you must invoke the Charge action on the Job object:

```
mam-shell UsageRecord Charge Data:="<UsageRecord><Instance>Moab.1234</Instance><Account>c
```

Use the Perl API

The Perl API exposes the full functionality of Moab Accounting Manager. The client commands can be examined as sample code. Use perldoc on the modules in lib/MAM for function documentation.

Example 26-10. To make a charge via this interface you might do something like:

```
use MAM;

my $request = new MAM::Request(object => "UsageRecord", action => "Charge");
my $usageRecord = new MAM::Datum("UsageRecord");
$usageRecord->setProperty("Instance", "Moab.1234");
$usageRecord->setProperty("Account", "chemistry");
```

```
$usageRecord->setProperty("User", "amy");
$usageRecord->setProperty("Machine", "colony");
$usageRecord->setProperty("Processors", "2");
$usageRecord->setProperty("Duration", "3600");
$request->addDatum($usageRecord);
$request->setOption("Duration", "3600");
my $response = $request->getResponse();
print $response->getStatus(), ": ", $response->getMessage(), "\n";
```

Communicating via the SSSRMAP Protocol

Finally, it is possible to interact with Moab Accounting Manager by directly using the SSSRMAP Wire Protocol and Message Format over the network (see *SSS Resource Management and Accounting Documentation* (http://www.clusterresources.com/products/gold/docs/)). This will entail building the request body in XML, appending an XML digital signature, combining these in an XML envelope framed in an HTTP POST, sending it to the server, and parsing the similarly formed response. The Moab Workload Manager communicates with Moab Accounting Manager via this method.

Example 26-11. The message might look something like:

POST /SSSRMAP HTTP/1.1
Content-Type: text/xml; charset="utf-8"
Transfer-Encoding: chunked

190
<?xml version="1.0" encoding="UTF-8"?>
<Envelope>
 <Body>
  <Request action="Charge" actor="scottmo">
  <Object>UsageRecord</Object>
   <Data>
    <UsageRecord>
     <Instance>Moab.1234</Instance>
     <Account>chemistry</Account>
     <User>amyh</User>
     <Machine>colony</Machine>
     <Processors>2</Processors>
     <Duration>3600</Duration>
    </UsageRecord>
   </Data>
   <Option name="Duration">3600</Option>
  </Request>
 </Body>
 <Signature>
  <DigestValue>azu4obZswzBt89OgATukBeLyt6Y=</DigestValue>
  <SignatureValue>YXE/C08XX3RX4PMU1bWju+5/E5M=</SignatureValue>

```
    <SecurityToken type="Symmetric"></SecurityToken>
  </Signature>
</Envelope>
0
```

Chapter 27. Configuration Files

Moab Accounting Manager uses four configuration files: one for connection information (mam-site.conf), one for the server (mam-server.conf), one for the clients (mam-client.conf) and one for the graphical user interface (mam-gui.conf). For configuration parameters that have hard-coded defaults, the default value is specified within brackets.

After modifying configuration parameters used by the server (such as those in the site configuration or server configuration files), you must restart the mam-server for the new settings to take effect. Alternatively, for most parameters, you can force the server to reread its configuration by running ‘mam-server --reconfig‘ or by sending the HUP signal to the main server process.

Site Configuration

The site configuration file specifies the connection information for the current site such as the server host name, port, backup server, default security method and the symmetric key. Optionally, it may also have blocks that specify connection information for other sites. This file should be readable only by the accounting admin user.

Example 27-1. The following is an example mam-site.conf file

```
server.host = red-head1
backup.host = red-head2
server.port = 7071
token.type = Symmetric
token.value = pBaIapJqbfLd8NiyzTJefFXW

[white]
server.host = white-head1
server.port = 7071
token.value = Fl7wOkioUpyjdqJ8ckvWK_ta

[blue]
server.host = blue-head1
server.port = 7071
token.value = gVSeQ8Diz5O3pZjO1Y4inGWq
```

The following configuration parameters may be set in the site configuration file (mam-site.conf).

- backup.host -- The hostname of the backup server. Each site can have both a primary server and a hot-standby backup server. They should either point to the same database or separate instances of a replicated database. If backup.host is specified, clients will try communicating with the primary server first, and if the connection fails, they will try communicating with the backup server. Since both the primary and backup servers may run simultaneously, events are disabled for the backup server so they do not conflict with events triggered by the primary server.
- server.host -- The hostname of the primary server

- server.port [7112] -- The port that the server listens on
- token.type [Symmetric] -- Indicates the default security token type to be used in both authentication and encryption. Valid token types include Password and Symmetric. The default is Symmetric.
- token.value -- When using the Symmetric token type, token.value is the secret key. It is a base64-encoded symmetric key used between clients and the server for authentication and encryption.

Server Configuration

The following configuration parameters may be set in the server configuration file (mam-server.conf).

- accounting.mode [strict-allocation] -- The accounting mode can be one of usage-tracking, notional-charging, fast-allocation or strict-allocation. If usage-tracking is specified, charges will simply result in the creation of usage records with no charge value. No charge will be calculated and allocations will not be debited. If notional-charging is specified, a charge will be calculated and recorded with the usage record, but allocations are not debited. If fast-allocation is specified, usage records will be updated with charge amounts and allocations will be debited, but liens will not be used to protect the allocation from simultaneous use. If strict-allocation is specified, usage records will be updated with charge amounts and allocations will be debited, and liens will be used to protect the allocation from simultaneous use.
- allocation.enforcediscrete [true] -- If enabled (the default), new allocations will be prevented from overlapping existing ones. This policy helps to improve clarity when reporting on allocation usage during a particular period.
- charge.itemization [false] -- Enables (true) or disables (false) the storing of itemized charges to the Charge table for charge transactions.
- currency.precision [0] -- Indicates the number of decimal places in the credit currency. For example, if you are will be dealing with an integer billable unit like processor-seconds, use 0 (which is the default). If you will be charging dollars and cents, then use 2. This parameter should be the same in the mam-server.conf and mam-client.conf files.
- database.datasource [DBI:Pg:dbname=mam;host=localhost] -- The Perl DBI data source name for the database you wish to connect to
- database.password -- The password to be used for the database connection (if any)
- database.user -- The username to be used for the database connection (if any)
- event.scheduler [true] -- Specifies whether the event scheduler is enabled (true) or not (false). Be aware that MAM relies on pre-configured events for refreshing stale allocations and notifications, hence disabling the event scheduler will prevent these updates from occurring.
- event.pollinterval [5] -- The period in minutes that the event scheduler uses to check and fire events. The poll interval must divide evenly into the number of minutes in a day (1440).
- log4perl.appender.Log.filename -- Used by log4perl to set the base name of the log file
- log4perl.appender.Log.max -- Used by log4perl to set the number of rolling backup logs

- log4perl.appender.Log.size -- Used by log4perl to set the size the log will grow to before it is rotated
- log4perl.appender.Log.Threshold -- Used by log4perl to set the debug level written to the log. The logging threshold can be one of TRACE, DEBUG, INFO, WARN, ERROR and FATAL
- log4perl.appender.Screen.Threshold -- Used by log4perl to set the debug level written to the screen. The logging threshold can be one of TRACE, DEBUG, INFO, WARN, ERROR and FATAL
- notification.deliverymethod [store] -- Specifies which deliverymethod is used by default if unspecified
- notification.duration [1209600] -- Defines how long in seconds that stored notifications persist before being automatically deleted. The default is two weeks
- response.chunksize [1000] -- Indicates the line length in the data response that will trigger message segmentation (or truncation). A value of 0 (zero) means unlimited, i.e. that the server will not truncate or segment large responses unless overriden by a chunksize specification in a client request. The response chunksize will be taken to be the smaller of the client and server chunksize settings.
- security.authentication [true] -- Indicates whether incoming message authentication is required
- security.encryption [false] -- Indicates whether incoming message encryption is required
- user.firstaccountdefault [true] -- If set to true, the first account that a user is added to will become the default account for that user. This default value is true.

Client Configuration

The following configuration parameters may be set in the client configuration file (mam-client.conf).

- accounting.mode [strict-allocation] -- The accounting mode can be one of usage-tracking, notional-charging, fast-allocation or strict-allocation. The value of this parameter may modify the default fields displayed by certain commands such as mam-list-usagerecords.
- account.show [Name,Active,Users,Organization,Description] -- The default fields shown by mam-list-accounts
- allocation.show [Id,Fund,StartTime,EndTime,InitialDeposit,Allocated,CreditLimit,Remaining,PercentUsed] -- The default fields shown by mam-list-allocations
- balance.show [Id,Name,Balance,Reserved,Effective,CreditLimit,Available] -- The default fields shown by mam-balance
- currency.precision [0] -- Indicates the number of decimal places in the credit currency. For example, if you are will be dealing with integer billable units like processor-seconds, use 0 (which is the default). If you will be charging dollars and cents, then use 2. This parameter should be the same in the mam-server.conf and mam-client.conf files

- event.show [Id,FireCommand,FireTime,ArmTime,RearmPeriod,EndTime,Notify,RearmOnFailure,FailureCommand,CatchUp,Creati -- The default fields shown by mam-list-events
- fund.show [Id,Name,Constraints,Allocated,Balance,DefaultDeposit,Description] -- The default fields shown by mam-list-funds
- log4perl.appender.Log.filename -- Used by log4perl to set the base name of the log file
- log4perl.appender.Log.max -- Used by log4perl to set the number of rolling backup logs
- log4perl.appender.Log.size -- Used by log4perl to set the size the log will grow to before it is rotated
- log4perl.appender.Log.Threshold -- Used by log4perl to set the debug level written to the log. The logging threshold can be one of TRACE, DEBUG, INFO, WARN, ERROR and FATAL
- log4perl.appender.Screen.Threshold -- Used by log4perl to set the debug level written to the screen. The logging threshold can be one of TRACE, DEBUG, INFO, WARN, ERROR and FATAL
- notification.show [Id,Event,Type,Status,Code,Message,Key,Recipient,EndTime,CreationTime] -- The default fields shown by mam-list-notifications
- organization.show [Name,Description] -- The default fields shown by mam-list-organizations
- quote.show [Id,Amount,Pinned,Instance,UsageRecord,StartTime,EndTime,Duration,ChargeRates,Description] -- The default fields shown by mam-list-quotes
- lien.show [Id,Instance,Amount,StartTime,EndTime,UsageRecord,Funds,Description] -- The default fields shown by mam-list-liens
- response.chunking [false] -- Indicates whether large responses should be chunked (segmented). If set to false, large responses will be truncated
- response.chunksize [1000] -- Indicates the line length in the data response that will trigger message segmentation (or truncation). A value of 0 (zero) means unlimited, i.e. that the client will accept the chunksize set by the server. The response chunksize will be taken to be the smaller of the client and server chunksize settings
- security.authentication [true] -- Indicates whether outgoing message are signed
- security.encryption [false] -- Indicates whether outgoing messages are encrypted
- security.promotion [suidperl] -- When using the symmetric key for security authentication or encryption, since the site configuration file is readable only by the accounting admin user, a method must be employed to temporarily elevate privileges in order to encrypt the communication with the symmetric key. One of two security promotion methods may be selected: suidperl or mamauth. Suidperl allows a Perl script to temporarily elevate privileges to the owner of the script if the setuid bit is set on the file. This method is recommended when suidperl can be installed on the system. If you prefer not to use suidperl or if it is not available for your system (such as with Perl 5.12 and higher), you will need to use the mamauth security promotion method. Mamauth uses a setuid binary executable that allows the request body to be passed in as standard input and returns the authenticated digest and signature. Currently, only suidperl can be used for encryption of client communication. The security promotion method should be configured at install time by specifying the --with-promotion configuration parameter and defaults to suidperl when it is available
- statement.show [Account,User,Machine] -- The default discriminator fields in mam-statement

- transaction.show
  [Id,Object,Action,Actor,Name,Child,Instance,Count,Amount,Delta,Balance,User,Account,Machine,Fund,Allocation,Usa
  -- The default fields shown by mam-list-transactions
- usagerecord.show
  [Id,Type,Instance,Charge,Stage,User,Group,Account,Organization,Class,QualityOfService,Machine,Nodes,Processors,M
  -- The default fields shown by mam-list-usagerecords
- user.show [Name,Active,CommonName,PhoneNumber,EmailAddress,DefaultAccount,Description]
  -- The default fields shown by mam-list-users

GUI Configuration

The following configuration parameters may be set in the GUI configuration file (mam-gui.conf).

- currency.enablehours [false] -- If set to true, the graphical user interface will include a ShowHours radio button (defaulting to True) for certain panels (e.g. Fund Deposit, Query, Statement, Transfer, Withdraw) that will allow the currency inputs or outputs to be divided by 3600
- currency.precision [0] -- Indicates the number of decimal places in the credit currency. For example, if you are will be dealing with integer billable units like processor-seconds, use 0 (which is the default). If you will be charging dollars and cents, then use 2. This parameter should be the same in the mam-server.conf and mam-client.conf files
- log4perl.appender.Log.filename -- Used by log4perl to set the base name of the log file
- log4perl.appender.Log.max -- Used by log4perl to set the number of rolling backup logs
- log4perl.appender.Log.size -- Used by log4perl to set the size the log will grow to before it is rotated
- log4perl.appender.Log.Threshold -- Used by log4perl to set the debug level written to the log. The logging threshold can be one of TRACE, DEBUG, INFO, WARN, ERROR and FATAL
- response.chunking [false] -- Indicates whether large responses should be chunked (segmented). If set to false, large responses will be truncated
- response.chunksize [1000] -- Indicates the line length in the data response that will trigger message segmentation (or truncation). A value of 0 (zero) means unlimited, i.e. that the client will accept the chunksize set by the server. The response chunksize will be taken to be the smaller of the client and server chunksize settings
- security.authentication [true] -- Indicates whether outgoing message are signed
- security.encryption [false] -- Indicates whether outgoing messages are encrypted
- security.promotion [suidperl] -- When using the symmetric key for security authentication or encryption, since the site configuration file is readable only by the accounting admin user, a method must be employed to temporarily elevate privileges in order to encrypt the communication with the symmetric key. One of two security promotion methods may be selected: suidperl or mamauth. Suidperl allows a Perl script to temporarily elevate privileges to the owner of the script if the setuid bit is set on the file. This method is recommended when suidperl can be installed on the system. If you

prefer not to use suidperl or if it is not available for your system (such as with Perl 5.12 and higher), you will need to use the mamauth security promotion method. Mamauth uses a setuid binary executable that allows the request body to be passed in as standard input and returns the authenticated digest and signature. Currently, only suidperl can be used for encryption of client communication. The security promotion method should be configured at install time by specifying the --with-promotion configuration parameter and defaults to suidperl when it is available

- statement.discriminators -- The Fund Statement page will group summary entries in the debit detail by these transaction properties.

Chapter 28. Web Services

Moab Accounting Manager Web Services (MAMWS) provides a REST-like interface permitting access to the full Moab Accounting Manager API. MAMWS communicates with Moab Accounting Manager using the same wire protocol, message format and Perl API as the MAM command-line clients and GUI interfaces. MAMWS runs under mod_perl from an apache httpd server.

This chapter describes the Web Services API and then enumerates specific examples of resources using that API. See the Installation documentation for information on installing and configuring MAM Web Services.

API

MAMWS provides a web interface using REST (Representation State Transfer) concepts to create, query, modify and delete objects in Moab Accounting Manager. MAMWS also supports additional actions as well as alternative syntax options for interacting with the web service. This section describes the format of the request and response syntax, as well as authentication and error code details.

URLs

A MAMWS URL is composed of a resource URI and optional query string. A resource URL is composed of the prefix and a resource. The prefix is the composed of the protocol (normally https), the MAM Web Services httpd server hostname or ip address, the location (/mamws) and an optional api version. MAMWS resources correspond with MAM objects and instances of those objects. Thus, a MAMWS resource is composed of a MAM object optionally followed by one or more primary keys.

```
<mamws_url> ::= <mamws_uri>[<query_string>]
<mamws_uri> ::= <mamws_prefix><mamws_resource>
<mamws_prefix> ::= <protocol>://<mamws_server>/mamws[/<version>]
<mamws_resource> ::= <mam_object>[/<primary_key>...]
<query_string> := ?<parameter>[&<parameter>...]
```

An expanded URL is of the form:

```
<protocol>://<mamws_server>/mamws[/<version>]/<object>[/<primary_key>...][?<parameter>[
```

The resource's object is specified in kebab-case and is normally pluralized. For example, /usage-records represents the UsageRecord object in MAM, while /usage-records/1 represents the instance of the UsageRecord object having the value 1 as the primary key. HTTP parameters and data are used as syntactical parameters and options for the API queries.

HTTP Methods

MAMWS supports the use of REST concepts utilizing HTTP (Hypertext Transfer Protocol) methods operating on endpoint URLs that describe resources. The HTTP methods used in MAMWS are comprised of the following:

Table 28-1. HTTP Methods

Method	Path Info	Description
GET	/<object>	Query for a list of resources
GET	/<object>/<primary_key>...	Query a single resource
POST	/<object>	Create a resource (primary key(s) not included in path)
PUT	/<object>/<primary_key>...	Create a resource (primary key(s) included in path)
PATCH	/<object>/<primary_key>...	Modify a resource
DELETE	/<object>/<primary_key>...	Delete a resource
POST	/<object>?action=<action>	Other actions

JSON Data Format

When HTTP data is included in the HTTP request or response, it is encoded in JSON object format. Input data for a POST or PATCH must be in JSON format with the top-level data type being a JSON object. The Content-Type header should be set to 'application/json'. Output data is always in JSON format and always consists of a JSON object with two or more key/value pairs. The output is "pretty-printed" by default.

Example 28-1. Sample request data

```
POST /users
{
   "active" : true,
   "common-name" : "Amy Miller",
   "default-account" : "chemistry",
   "email-address" : "amy@hpc.com",
   "name" : "amy",
   "phone-number" : "(801) 555-1437"
}
```

Example 28-2. Sample response data

```
GET /users/amy

{
   "code" : "000",
   "count" : 1,
```

```
   "data" : [
      {
         "active" : true,
         "common-name" : "Amy Miller",
         "default-account" : "chemistry",
         "description" : null,
         "email-address" : "amy@hpc.com",
         "name" : "amy",
         "phone-number" : "(801) 555-1437"
      }
   ],
   "status" : "Success"
}
```

API Version

The current latest version is version 1 (v1). This is currently the only existing version. However, it is practically inevitable that there will eventually be compatibility-breaking changes to the api and so we immediately introduce the support for versioning.

The version
is optional and when used is appended to the prefix (effectively prepended to the resource) in the URL (i.e. https://<mamws_server>/mamws[/<version>]/<object>[/<primary_key>...][?<parameter>[&<parameter>...]]).

If you omit the version in the URL, the web services client will use the current latest version. The request will fail if an invalid version is specified.

Example 28-3. Specifying the API version in a query

```
GET https://localhost/mamws/v1/users
```

Request Format

A MAMWS Request includes the object (or instance which consists of the object and primary keys) and the action (whether explicit or implied), and may provisionally include selections, assignments, conditions, options, data and meta-options.

Object

The request object is specified in the URL path info. Some actions or methods additionally require or allow primary keys to be specified as additional path elements in the URL to specify the object instance.

Example 28-4. Specifying the User object in a query (querying all users)

```
GET /users
```

Example 28-5. Specifying the instance of the User object having the primary key 'amy' (querying just the user amy)

```
GET /users/amy
```

Example 28-6. Listing valid objects

```
GET /objects?fields=name
```

Example 28-7. Listing primary keys for the usage record object in sequential order

```
GET /attributes?filter=object=UsageRecord,primary-key=True&fields=name&sort-by-sequence
```

> **Note:** The values of the filters (object and primary-key) must be specified in UpperCamel case since the web service interface does not translate the case for values.

Example 28-8. Listing all attributes for the usage record object

```
GET /attributes?filter=object=UsageRecord&fields=sort(name)
```

Action

The request action may be specified via the action parameter or it may be implied by the HTTP method. When not specified via parameter, the action will be implied from the HTTP method. The GET method implies the Query action, the PUT and POST methods imply the Create action, the PATCH method implies the Modify action and the DELETE method implies the Delete action. The POST method will permit any supported action to be explicitly specified via the action parameter. All other methods are restricted to their default action.

Example 28-9. The Delete action is implied by the DELETE method

```
DELETE /users/amy
```

Example 28-10. The Refund action is specified explicitly via the action parameter

```
POST /usage-records/1?action=refund&id=1
```

Example 28-11. Listing all actions available to the usage record object

```
GET /actions?filter=object=UsageRecord&fields=sort(name)
```

Other Request Components

Other components of the request, including selections, assignments, conditions, options, and data, may be specified via parameters in the query string or via JSON JSON data.

Table 28-2. Parameter Types Allowed by Actions

Parameter Type	Description	Examples
Selections	Designate which properties of an object are returned in a query	fields=name,active

Assignments	Specify new field values when creating and modifying objects	update=active=true
		active@=true
		{
		"active" : true
		}

Conditions	Specify which objects to query, update or delete	filter=active=true active==true
Options	Specify additional business-logic parameters	options=show-hidden=true show-hidden:=true show-hidden=true

Data	Although not a parameter type, JSON data may be used with some actions as assignment properties or as input data	{
		"processors" : 2,
		"account" : "chemistry"
		}

Meta-Options	Options used to by the web services client and not forwarded in the MAM request	pretty=false !pretty

Refer to the following chart for the request components supported by different actions.

Table 28-3. Parameter Types Allowed by Action

Actions	Request Components Supported
Query	[Selections], [Conditions], [Options], [Meta-Options]

Actions	Request Components Supported
Create	Assignments*, [Options], [Meta-Options]
Modify	Assignments*, [Conditions], [Options], [Meta-Options]
Delete, Undelete	[Conditions], [Options], [Meta-Options]
All other actions	[Options], [Data], [Meta-Options]

Note: Use of square brackets means that the use of this request component is optional.

Selections

Selections designate the fields that are to be returned in a query. Besides simple field selection, selection criteria may also include sorting, extraction of partial values from complex data types, aliases, and aggregation (sum, average, min, max, etc).

Selections are expressed as a comma-separated list of desired object properties as the value of the fields parameter in the following form:

fields=[<aggregation_function>(]<name>[{<part>}])[=<alias>],...

Table 28-4. Selection Parameter Components

Selection Parameter Component	Description	Examples
aggregation_function	Designates sorting or an aggregation function to apply to the field • sum (sum of values) • average (average of values) • min (minimum value) • max (maximum value) • count (count of values) • group-by (group-by field) • sort (descending sort) • tros (increasing sort)	fields=sum(amount),group-by(account) fields=sort(name)
name	Name of the field or object property to display or use in an aggregation	fields=name,email-address
part	Name of the part to extract from the complex object property	fields=resources{telescope}
alias	Designates what to call the returned property or aggregation value	fields=name=user fields=sum(amount)=total

Note: Aliases for the fields parameter include select, show and get

Assignments

Assignments designate the new values in the creation or modification of objects. Besides simple assignment, assignments may alternatively increment or decrement the value.

Assignments may be expressed in one of three different ways: via the update parameter, directly with the property name as the parameter name with an assignment operator, or as JSON data.

When using the update parameter, assignments are expressed as a comma-separated list of update expressions in the following form:

update=<name><op><value>,...

Table 28-5. Assignment Parameter Components

Assignment Parameter Component	Description	Examples
name	Name of the object property to set or update	update=name=amy,active=true
op	Designates whether the specified value should be assigned to the property, or used to increment or decrement it • = (assignment) • += (increment) • -= (decrement)	update=duration+=3600
value	Designates the value to assign as the new value of the property or the amount to increment or decrement it. Use null to unset the object property.	update=email-address=null

Note: Aliases for the update parameter include assign and set.

As a second alternative, assignments may be expressed directly with the property name as the parameter name with an assignment operator in the form:

<name><op><value>

Table 28-6. Assignment Operator Components

Assignment Operator Component	Description	Examples

Assignment Operator Component	Description	Examples
name	Name of the object property to set or update	active@=true
op	Designates whether the specified value should be assigned to the property, or used to increment or decrement it • @= (assignment) • += (increment) • -= (decrement)	duration+=3600
value	Designates the value to assign as the new value of the property or the amount to increment or decrement it. Use null to unset the object property.	email-address@=null

And as a third alternative, the properties to be assigned may be expressed as a JSON object in the HTTP request data in the form:

```
{
  <name> : <value>,...
}
```

Table 28-7. Assignment Data Components

Assignment Data Component	Description	Examples

name	Name of the object property to set or update	{
		"name" : "amy",
		"active" : true
		}

value	Designates the value to assign as the new value of the property or the amount to increment or decrement it. Use null to unset the object property.	{
		"email-address" : null
		}

Note: This form cannot be used to increment or decrement the object property.

Conditions

Conditions allow filtering of the objects to be queried, updated or acted upon. Besides simple equality

conditions, condition criteria may include filtering on part names of a complex value, comparisons (greater-than, not equal, etc), pattern matching, conjunctions (and, or) and grouping.

Conditions may be expressed in one of two different ways: via the filter parameter or directly with the property name as the parameter name with a condition operator.

When using the filter parameter, conditions are expressed as a list of filter expressions (delimited with the respective conjunction symbol) in the following form:

filter=[<pre-group>]<name>[{<part>}]<op><value>[<post-group>]<conjunction>...

Table 28-8. Condition Parameter Components

Condition Parameter Component	Description	Examples
pre-group	Zero or more open parentheses used for grouping of anded and ored conditions	filter=(instance~j1|charge>10),id<5&fields=id
name	Name of the object property used in determining which objects to include in the query or update	filter=active=true
part	Designates to only include objects having an individual named part with the specified value	filter=resources{telescope}==2
op	Comparison or matching operator employed to determine whether objects having the specified name are included in the operation • == or = (equality) • > (greater than) • >= (greater than or equal to) • < (less than) • <= (less than or equal to) • != (not equal to) • ~ (matches) • !~ (does not match) The following wildcards are supported with matching operators • ? (matches any one character) • * (matches zero or more of any characters)	filter=processors>=4 filter=account~chem*
value	Value of the specified object property. Use null to include objects whose specified property is unset.	filter=email-address==null

Condition Parameter Component	Description	Examples
pre-group	Zero or more open parentheses used for grouping of anded and ored conditions	filter=instance~j1l(charge>10,id<5)&fields=id
conjunction	Symbol used to connect condition groups indicating whether the current and preceding condition group should be anded or ored • , (and) • | (or)	filter=user==amy,account==chemistry filter=account==chemistry|account==biology

Note: Aliases for the filter parameter include query and where.

As a second alternative, conditions may be expressed directly with the property name as the parameter name with a condition operator in the form:

<name><op><value>

Table 28-9. Condition Operator Components

Condition Operator Component	Description	Examples
name	Name of the object property used to filter objects included in the operation	active==true
op	Comparison or matching operator employed to determine whether objects having the specified name are included in the operation • == (equality) • > (greater than) • >= (greater than or equal to) • < (less than) • <= (less than or equal to) • != (not equal to) • ~ (matches) • !~ (does not match) The following wildcards are supported with matching operators • ? (matches any one character) • * (matches zero or more of any characters)	processors>=4 account~chem*

Condition Operator Component	Description	Examples
value	Value of the specified object property. Use null to specify objects whose named property is unset.	email-address==null

Note: This form cannot be used to specify parts, conjunctions or grouping.

Options

Options specify additional business-logic options the may affect the behavior of the request or resulting response.

Options may be expressed in one of three different ways: via the options parameter, directly with the option name as the parameter name with an option operator, or using the meta-option operator (=) where there is no similarly-named meta-option.

When using the option parameter, options are expressed as a list of comma-delimited option expressions in the following form:

options=<name>=<value>,...

Table 28-10. Option Parameter Components

Option Parameter Component	Description	Examples
name	Name of the option	options=job-id=2,amount=1.5
value	Value of the option.	options=show-hidden=true

As a second alternative, options may be expressed directly with the option name as the parameter name with an option operator in the form:

<name><op><value>

Table 28-11. Option Operator Components

Option Operator Component	Description	Examples
name	Name of the option	active:=true
op	Option operator • := (assertion)	filter-type:=NonExclusive

Option Operator Component	Description	Examples
value	Value of the option. As a shorthand notation for a boolean value of true, the operator and the value can be omitted. As a shorthand notation for a boolean value of false, the name can be preceded by an exclamation point (!) and the operator and value omitted.	active !active

As a third alternative, parameters of the form <name>=<value> that are not interpreted as meta-options will be taken as request options. However, when using this form, care must be taken to avoid conflict with the meta-options.

For example, filter=User=amy should not be used to express the Filter request option with the value User=amy since this expression would be interpreted as specifying the filter meta-option for the User condition with value amy. In this case, you would need to use either the constraint-filter meta-option (constraint-filter=User=amy), the options meta-option (options=filter=User=amy) or the option operator (filter:=User=amy).

Data

Some actions require input data with the request (e.g. Charge, Reserve and Quote require a usage record as input data). Other actions, such as Create and Modify, allow the newly created or updated fields to be passed in via the data as an alternate form of expressing the assignment fields.

In all cases, data is expressed as a JSON object in the following form:

```
{
  <name> : <value>,...
}
```

Table 28-12. Data Components

Request Data Component	Description	Examples

name	Name of the object property	{
		"name" : "amy",
		"active" : true
		}

Request Data Component	Description	Examples

value	Value of the object property. In some cases, the value itself can be a simple JSON object (e.g. complex usage record fields)	{
		"class" : null,
		"amount" : 12.5,
		"resources" : {
		"telescope" : 2
		}
		}

Meta-Options

Meta-options are HTTP parameters used by the web services client and not forwarded in the request to Moab Accounting Manager. Meta-options include fields, update, filter, options and their respective aliases. Additional supported meta-options are listed in the following table:

Table 28-13. Supplemental Meta-Options*

Meta-Option Name	Function	Example
force	In some situations, asserting the force parameter may allow an action to do something potentially dangerous or bend RESTful rules, such as allowing the PATCH or DELETE methods to operate on multiple instances.	force=true force
pretty	Pretty-printing is enabled by default. To disable it, deassert the pretty parameter.	pretty=false !pretty
suppress-nulls	When rendering the response data in JSON, by default, null-valued fields are explicitly shown as having the null value. Asserting the suppress-nulls parameter will avoid printing fields having a null value.	suppress-nulls=true suppress-nulls

* Meta-options also include fields, update, filter, options and their respective aliases described in previous sections.

Response Format

A MAMWS Response has an HTTP status code and HTTP data. The HTTP data is in the form of a JSON object with key value pairs that includes a MAM status and code (different from the HTTP status code), and may optionally include a message, a count, and JSON data. The MAM response is expressed

in the HTTP response data as a JSON object of the following form:

```
{
  "code" : <code>,
  ["count" : <count>,]
  ["data" : <data>,]
  ["message" : <message>,]
  "status" : <status>
}
```

Table 28-14. Response Data Components

Response Data Component	Description	Examples
code	MAM SSSRMAP* Status Code	"code" : "740"
count	Usually number of objects returned or affected, but sometimes is used to return other key values such as amount charged	"count" : 24

data	Response data as a JSON object	"data" : [
		{
		"name" : "amy",
		"active" : true
		},
		{
		"name" : "bob",
		"active" : false
		}
		]

message	Response message	"message" : "Successfully modified 2 users"
status	Status • Success • Warning • Failure	"status" : "Failure"

* SSSRMAP stands for Scalable Systems Software Resource Manager and Accounting Protocol.

HTTP Codes

Table 28-15. HTTP Status Codes

HTTP Status Code	Description	When Used

HTTP Status Code	Description	When Used
200	OK	Successful response received from MAM server
400	Bad Request	Invalid request on the client side or any business-logic or miscellaneous problem that the server could not successfully fulfill.
401	Unauthorized	User did not successfully authenticate
403	Forbidden	User is not authorized to perform the request
404	Not Found	The specified resource does not exist
405	Method Not Allowed	The HTTP method is not used in the API

Status Codes

MAMWS uses 3 digit SSSRMAP status codes in the JSON response object. See the documents "Scalable Systems Software Resource Management and Accounting Protocol (SSSRMAP) Message Format" (found at http://www.adaptivecomputing.com/products/open-source/gold/gold-product-documentation/) for the list and meanings of these codes.

Authentication

MAMWS uses HTTP Basic Authentication for all REST API requests. The required username and password is forwarded to the MAM server for authentication and authorization. Thus, each user that wants to be able to use MAM Web Services must first set a password in MAM (e.g with the mam-set-password client command).

The username and password in the Basic Authentication header are encoded but not encrypted. Therefore, it is strongly recommended that MAMWS be run under an httpd server with SSL enabled.

Actions

This section is included to aid in mapping from MAM actions to HTTP methods and URIs in MAM Web Services.

Query

To query an object, use the GET method. In MAM, there is no fundamental difference between querying a single instance of an object or multiple instances of the object. Querying a single object simply includes a query filter using the object's primary keys. With REST, these are differentiated by the presence of additional path info nodes in the request URI.

Table 28-16. HTTP Methods for the Query Action

HTTP Method	MAMWS Resource	Description	Example
GET	/<object>	Query multiple objects	GET /users
GET	/<object>{/<primary_key_value>}	Query a single object	GET /users/amy

In a MAMWS query response, the selected object properties are returned in the JSON data field as an array of objects. This is true both when querying in the single object form or in the multiple object form, and is done this way so that a client can use the same parsing routine for both cases.

Create

There are two primary methods that can be used to create resources (objects) in MAMWS: using POST and using PUT. When using POST, the resource URI should not include the primary keys with the object in the path info. Alternatively, when using PUT, the resource URI must include the primary keys with the object in the path info. Thus, PUT may only be used when you know the primary keys that will uniquely define the object instance being created. The POST method will be considered the primary method since it is considered more straightforward to put all of the new object properties in a single location (the request data).

Table 28-17. HTTP Methods for the Create Action

HTTP Method	MAMWS Resource	Description	Example

POST	/<object>	Create an object (primary key(s) not included in path)	PUT /users
			{
			"name" : "amy",
			"active" : true
			}

PUT	/<object>{/<primary_key_value>}	Create an object (primary key(s) included in path)	PUT /users/amy
			{
			"active" : true
			}

Modify

To modify an object, use the PATCH method.

Table 28-18. HTTP Method for the Modify Action

HTTP Method	MAMWS Resource	Description	Example

PATCH	/<object>{/<primary_key>}	Modify an object	PATCH /users/amy
			{
			"active" : false
			}

Delete

To delete an object, use the DELETE method.

Table 28-19. HTTP Method for the Delete Action

HTTP Method	MAMWS Resource	Description	Example
DELETE	/<object>{/<primary_key>}	Delete an object	DELETE /users/amy

Other actions

All other actions are implemented via the POST method and using the action parameter.

Table 28-20. HTTP Method for Other Actions

HTTP Method	MAMWS Resource	Description	Example
POST	/<object>	Perform an action against an object	Post /users?action=undelete&filter=name=amy

Resources

This section enumerates the standard MAMWS resources and the actions (e.g. HTTP methods) supported by them.

MAMWS resources map directly to MAM objects. With the aim to help distinguish the primary accounting-related resources from the resources used as part of the internal base information service, we have here divided the resources into two broad subsections, accounting resources and framework resources, although the distinction is purely conceptual and there is no functional difference in the interface.

Accounting Resources

Accounts

Supported Actions

Action	HTTP Method	Resource
Query accounts	GET	/accounts[/<name>]
Create an account	POST	/accounts
Modify an account	PATCH	/accounts/<name>
Delete an account	DELETE	/accounts/<name>
Query account users	GET	/account-users[/<account>[/<user>]]
Add a user to an account	POST	/account-users
Modify an account user	PATCH	/account-users/<account>/<user>
Remove a user from an account	DELETE	/account-users/<account>/<user>

Query Accounts

Synopsis:

GET /accounts[/<name>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
constraint-filter	Applies meta-filters to the query (user: include only accounts having the specified user)	GET /accounts?constraint-filter=user=amy
fields	Designates the properties to be returned in the query	GET /accounts?fields=name
filter	Filters the objects to be returned in the query	GET /accounts?filter=organization=sciences
limit	Limits the results to the number of objects specified	GET /accounts?limit=100
offset	Number of objects to skip before starting to return data	GET /accounts?offset=100

Parameter	Description	Example
show-hidden	Includes hidden attributes in the result	GET /accounts?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /accounts?fields=organization&unique=true

Example 28-12. Sample Request

```
GET /accounts/amy
```

Example 28-13. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "description" : "Chemistry Department",
         "name" : "chemistry",
         "organization" : "sciences"
      }
   ],
   "status" : "Success"
}
```

Create an Account

Synopsis:

POST /accounts[?<parameter>]
{
<name> : <value>,...
}

Parameters:

Parameter	Description	Example

create-fund	Overrides the fund auto-generation setting	POST /accounts?create-fund=true
		{
		"name" : "chemistry"
		}

Example 28-14. Sample Request

```
POST /accounts
{
   "description" : "Chemistry Department",
   "name" : "chemistry",
   "organization" : "sciences"
}
```

Example 28-15. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "description" : "Chemistry Department",
         "name" : "chemistry",
         "organization" : "sciences"
      }
   ],
   "message" : "Successfully created 1 account",
   "status" : "Success"
}
```

Modify an Account

Synopsis:

PATCH /accounts/<name>
{
<name> : <value>,...
}

Example 28-16. Sample Request

```
PATCH /accounts/chemistry
{
   "active" : false
}
```

Example 28-17. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : false,
         "description" : "Chemistry Department",
         "name" : "chemistry",
         "organization" : "sciences"
      }
   ],
   "message" : "Successfully modified 1 account",
   "status" : "Success"
}
```

Delete an Account

Synopsis:

DELETE /accounts/<name>

Example 28-18. Sample Request

```
DELETE /accounts/chemistry
```

Example 28-19. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "description" : "Chemistry Department",
         "name" : "chemistry",
         "organization" : "sciences"
      }
   ],
   "message" : "Successfully deleted 1 account",
   "status" : "Success"
}
```

Query Account Users

Synopsis:

GET /account-users[/<account>[/<user>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /account-users/chemistry?fields=name
filter	Filters the objects to be returned in the query	GET /account-users?filter=name=amy
limit	Limits the results to the number of objects specified	GET /account-users?limit=100
offset	Number of objects to skip before starting to return data	GET /account-users?offset=100
show-hidden	Includes hidden attributes in the result	GET /account-users?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /account-users?fields=name&unique=true

Example 28-20. Sample Request

```
GET /account-users/chemistry?fields=name
```

Example 28-21. Sample Response

```
{
   "code" : "000",
   "count" : 2,
   "data" : [
      {
         "name" : "amy"
      },
      {
         "name" : "dave"
      }
   ],
   "status" : "Success"
}
```

Add a User to an Account

Synopsis:

POST /account-users
{
<name> : <value>,...
}

Example 28-22. Sample Request

```
POST /account-users
{
   "account" : "chemistry",
   "active" : true,
   "admin" : true,
   "name" : "amy"
}
```

Example 28-23. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "active" : true,
         "admin" : true,
         "name" : "amy"
      }
   ],
   "message" : "Successfully created 1 account user",
   "status" : "Success"
}
```

Modify an Account User

Synopsis:

PATCH /account-users/<account>/<user>
{
<name> : <value>,...
}

Example 28-24. Sample Request

```
PATCH /account-users/chemistry/amy
{
   "active" : false
}
```

Example 28-25. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "active" : false,
         "admin" : true,
         "name" : "amy"
      }
   ],
   "message" : "Successfully modified 1 account user",
   "status" : "Success"
}
```

Remove a User from an Account

Synopsis:

DELETE /account-users/<account>/<user>

Example 28-26. Sample Request

```
DELETE /accounts-users/chemistry/amy
```

Example 28-27. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "active" : false,
         "admin" : true,
         "name" : "amy"
       }
   ],
```

```
   "message" : "Successfully deleted 1 account user",
   "status" : "Success"
}
```

Allocations

Supported Actions

Action	HTTP Method	Resource
Query allocations	GET	/allocations[/<id>]
Modify an allocation	PATCH	/allocations/<id>
Delete an allocation	DELETE	/allocations/<id>

Query Allocations

Synopsis:

GET /allocations[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
constraint-filter	Displays allocations whose fund constraints comply with the specified filters	GET /allocations?constraint-filter=user=amy
fields	Designates the properties to be returned in the query	GET /allocations?fields=id,amount
filter	Filters the objects to be returned in the query	GET /allocations?filter=active=true
filter-type	Designates the constraint filter type	GET /allocations?constraint-filter=user=amy&filter-type=ExactMatch
limit	Limits the results to the number of objects specified	GET /allocations?limit=100
offset	Number of objects to skip before starting to return data	GET /allocations?offset=100
show-hidden	Includes hidden attributes in the result	GET /allocations?show-hidden=true

Parameter	Description	Example
unique	Displays only unique results (like DISTINCT in SQL)	GET /allocations?fields=fund&unique=true

Example 28-28. Sample Request

```
GET /allocations/2
```

Example 28-29. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "allocated" : 3000,
         "amount" : 3000,
         "credit-limit" : 0,
         "description" : null,
         "end-time" : "Infinity",
         "fund" : 2,
         "id" : 2,
         "initial-deposit" : 3000,
         "start-time" : "2016-06-15 18:29:44"
      }
   ],
   "status" : "Success"
}
```

Modify an Allocation

Synopsis:

PATCH /allocations/<id>
{
<name> : <value>,...
}

Example 28-30. Sample Request

```
PATCH /allocations/2
{
   "credit-limit" : 1000
}
```

Example 28-31. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "allocated" : 3000,
         "amount" : 3000,
         "credit-limit" : 1000,
         "description" : null,
         "end-time" : "Infinity",
         "fund" : 2,
         "id" : 2,
         "initial-deposit" : 3000,
         "start-time" : "2016-06-15 18:29:44"
      }
   ],
   "message" : "Successfully modified 1 allocation",
   "status" : "Success"
}
```

Delete an Allocation

Synopsis:

DELETE /allocations/<id>

Example 28-32. Sample Request

```
DELETE /allocations/2
```

Example 28-33. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "allocated" : 3000,
         "amount" : 3000,
         "credit-limit" : 0,
         "description" : null,
         "end-time" : "Infinity",
         "fund" : 2,
         "id" : 2,
         "initial-deposit" : 3000,
```

```
        "start-time" : "2016-06-15 18:29:44"
      }
   ],
   "message" : "Successfully deleted 1 allocation",
   "status" : "Success"
}
```

Charges

Supported Actions

Action	HTTP Method	Resource
Query itemized charges	GET	/charges

Query Itemized Charges

Synopsis:

GET /charges[?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /charges?fields=sum(amount)
filter	Filters the objects to be returned in the query	GET /charges?filter=usage-record=1
limit	Limits the results to the number of objects specified	GET /charges?limit=100
offset	Number of objects to skip before starting to return data	GET /charges?offset=100
show-hidden	Includes hidden attributes in the result	GET /charges?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /charges?fields=name&unique=true

Example 28-34. Sample Request

```
GET /charges?filter=usage-record=1
```

Example 28-35. Sample Response


```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 1,
         "description" : null,
         "details" : "12 [Processors] * 0.000277777777777778 [ChargeRate{Processors}] * 300
         "duration" : 300,
         "instance" : "24809",
         "name" : "Processors",
         "rate" : "1/h",
         "scaling-factor" : 1,
         "usage-record" : 1,
         "value" : "12"
      }
   ],
   "status" : "Success"
}
```


Charge Rates

Supported Actions

Action	HTTP Method	Resource
Query charge rates	GET	/charge-rates[/<name>[/<value>]]
Create a charge rate	POST	/charge-rates
Modify a charge rate	PATCH	/charge-rates/<name>/<value>
Delete a charge rate	DELETE	/charge-rates/<name>/<value>

Query Charge Rates

Synopsis:

GET /charge-rates[/<name>[/<value>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /charge-rates?fields=name

Parameter	Description	Example
filter	Filters the objects to be returned in the query	GET /charge-rates?filter=name=Processors
limit	Limits the results to the number of objects specified	GET /charge-rates?limit=100
offset	Number of objects to skip before starting to return data	GET /charge-rates?offset=100
show-hidden	Includes hidden attributes in the result	GET /charge-rates?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /charge-rates?fields=name&unique=true

Example 28-36. Sample Request

```
GET /charge-rates
```

Example 28-37. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : "1/h",
         "description" : "1 credit per processor-hour",
         "name" : "Processors",
         "value" : null
      }
   ],
   "status" : "Success"
}
```

Create a Charge Rate

Synopsis:

POST /charge-rates

{

<name> : <value>,...

}

Example 28-38. Sample Request

```
POST /charge-rates
{
   "amount" : "1/h",
   "description" : "1 credit per processor-hour",
   "name" : "Processors"
}
```

Example 28-39. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : "1/h",
         "description" : "1 credit per processor-hour",
         "name" : "Processors",
         "value" : null
      }
   ],
   "message" : "Successfully created 1 charge rate",
   "status" : "Success"
}
```

Modify a Charge Rate

Synopsis:

PATCH /charge-rates/<name>/<value>
{
<name> : <value>,...
}

Example 28-40. Sample Request

```
PATCH /charge-rates/Processors/null
{
   "amount" : "2/h"
}
```

Example 28-41. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : "2/h",
         "description" : "1 credit per processor-hour",
         "name" : "Processors",
         "value" : null
      }
   ],
   "message" : "Successfully modified 1 charge rate",
   "status" : "Success"
}
```

Delete a Charge Rate

Synopsis:

DELETE /charge-rates/<name>/<value>

Example 28-42. Sample Request

```
DELETE /charge-rates/Processors/null
```

Example 28-43. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : "1/h",
         "description" : "1 credit per processor-hour",
         "name" : "Processors",
         "value" : null
      }
   ],
   "message" : "Successfully deleted 1 charge rate",
   "status" : "Success"
}
```

Funds

Supported Actions

Action	HTTP Method	Resource
Query funds	GET	/funds[/<id>]
Create a fund	POST	/funds
Modify a fund	PATCH	/funds/<id>
Delete a fund	DELETE	/funds/<id>
Query fund constraints	GET	/constraints[/<id>]
Add a fund constraint	POST	/constraints
Remove a fund constraint	DELETE	/constraints/<id>
Deposit into a fund	POST	/funds?action=deposit
Withdraw from a fund	POST	/funds?action=withdraw
Transfer between funds	POST	/funds?action=transfer
Reset a fund	POST	/funds?action=reset

Query Funds

Synopsis:

GET /funds[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
constraint-filter	Displays funds whose constraints do not conflict with the specified filters	GET /funds?constraint-filter=user=amy
fields	Designates the properties to be returned in the query	GET /funds?fields=id,name
filter	Filters the objects to be returned in the query	GET /funds?filter=priority>0
filter-type	Designates the constraint filter type	GET /funds?constraint-filter=user=amy&filter-type=ExactMatch
limit	Limits the results to the number of objects specified	GET /funds?limit=100
offset	Number of objects to skip before starting to return data	GET /funds?offset=100
show-hidden	Includes hidden attributes in the result	GET /funds?show-hidden=true

Parameter	Description	Example
unique	Displays only unique results (like DISTINCT in SQL)	GET /funds?fields=priority&unique=true

Example 28-44. Sample Request

```
GET /funds/2
```

Example 28-45. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "default-deposit" : -1,
         "description" : null,
         "id" : 2,
         "name" : "chemistry",
         "priority" : 0
      }
   ],
   "status" : "Success"
}
```

Create a Fund

Synopsis:

POST /funds[?<parameter>]
{
<name> : <value>,...
}

Parameters:

Parameter	Description	Example
constraint	Specifies a constraint for the fund	POST /funds?constraint=account=chemistry

Example 28-46. Sample Request

```
POST /funds?constraint=account=chemistry
{
```

```
   "default-deposit" : 5000
}
```

Example 28-47. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "default-deposit" : 5000,
         "description" : null,
         "id" : 2,
         "name" : "chemistry",
         "priority" : 0
      }
   ],
   "message" : "Successfully created 1 fund with id 2 and 1 constraint",
   "status" : "Success"
}
```

Modify a Fund

Synopsis:

PATCH /funds/<id>
{
<name> : <value>,...
}

Example 28-48. Sample Request

```
PATCH /funds/2
{
   "default-deposit" : -1
}
```

Example 28-49. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "default-deposit" : -1,
         "description" : null,
```

```
         "id" : 2,
         "name" : "chemistry",
         "priority" : 0
      }
   ],
   "message" : "Successfully modified 1 fund",
   "status" : "Success"
}
```

Delete a Fund

Synopsis:

DELETE /funds/<id>

Example 28-50. Sample Request

```
DELETE /funds/2
```

Example 28-51. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "default-deposit" : -1,
         "description" : null,
         "id" : 2,
         "name" : "chemistry",
         "priority" : 0
      }
   ],
   "message" : "Successfully deleted 1 fund",
   "status" : "Success"
}
```

Query Fund Constraints

Synopsis:

GET /constraints[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /constraints?fields=fund,name,value
filter	Filters the objects to be returned in the query	GET /constraints?filter=name=Account,value=chemistry
limit	Limits the results to the number of objects specified	GET /constraints?limit=100
offset	Number of objects to skip before starting to return data	GET /constraints?offset=100
show-hidden	Includes hidden attributes in the result	GET /constraints?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /constraints?fields=name&unique=true

Example 28-52. Sample Request

```
GET /constraints?filter=fund=2
```

Example 28-53. Sample Response

```
{
   "code" : "000",
   "count" : 2,
   "data" : [
      {
         "fund" : 2,
         "id" : 2,
         "name" : "Account",
         "value" : "chemistry"
      }
   ],
   "status" : "Success"
}
```

Add a Fund Constraint

Synopsis:

POST /constraints

{

<name> : <value>,...

}

Example 28-54. Sample Request

```
POST /constraints
{
   "fund" : 2,
   "name" : "Account",
   "value" : "chemistry"
}
```

Example 28-55. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "fund" : 2,
         "id" : 2,
         "name" : "Account",
         "value" : "chemistry"
      }
   ],
   "message" : "Successfully created 1 constraint",
   "status" : "Success"
}
```

Remove a Fund Constraint

Synopsis:

DELETE /constraints/<id>

Example 28-56. Sample Request

```
DELETE /constraints/2
```

Example 28-57. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "fund" : 2,
         "id" : 2,
         "name" : "Account",
         "value" : "chemistry"
```

```
      }
   ],
   "message" : "Successfully deleted 1 constraint",
   "status" : "Success"
}
```

Deposit into a Fund

Synopsis:

POST /funds?action=deposit[&<parameter>...]

Parameters:

Parameter	Description	Example
allocation	Specifies that the deposit should go into the specified allocation	POST /funds?action=deposit&allocation=2&amount=1000
amount	Amount to deposit	POST /funds?action=deposit&id=2&amount=1000
constraint-filter	Restricts the fund to one whose constraints do not conflict with the specified filters	POST /funds?action=deposit&constraint-filter=account=chemistry&amount=1000
credit-limit	Credit limit for the new allocation	POST /funds?action=deposit&id=2&credit-limit=1000
filter-type	Designates the constraint filter type	POST /funds?action=deposit&constraint-filter=account=chemistry&filter-type=ExactMatch&amount=1000
id	Id of the fund into which the deposit will be made	POST /funds?action=deposit&id=2&amount=1000
reset	Ends the current allocation and creates a new allocation with the deposit	POST /funds?action=deposit&id=2&amount=1000&reset=

Example 28-58. Sample Request

```
POST /funds?action=deposit&id=2&amount=1000
```

Example 28-59. Sample Response

```
{
   "code" : "000",
   "count" : 1000,
   "message" : "Successfully deposited 1000.00 credits into fund 2",
   "status" : "Success"
}
```

Withdraw from a Fund

Synopsis:

POST /funds?action=withdraw[&<parameter>...]

Parameters:

Parameter	Description	Example
allocation	The credits will be withdrawn from the specified allocation only	POST /funds?action=withdraw&allocation=2&amount=10
amount	Amount to withdraw	POST /funds?action=withdraw&id=2&amount=1000
constraint-filter	Restricts the fund to one whose constraints do not conflict with the specified filters	POST /funds?action=withdraw&constraint-filter=account=chemistry&amount=1000
filter-type	Designates the constraint filter type	POST /funds?action=withdraw&constraint-filter=account=chemistry&filter-type=ExactMatch&amount=1000
id	Id of the fund from which the withdrawal will be made	POST /funds?action=withdraw&id=2&amount=1000

Example 28-60. Sample Request

```
POST /funds?action=withdraw&id=2&amount=1000
```

Example 28-61. Sample Response

```
{
   "code" : "000",
```

```
   "count" : 1000,
   "message" : "Successfully withdrew 1000.00 credits from fund 2",
   "status" : "Success"
}
```

Transfer between Funds

Synopsis:

POST /funds?action=transfer[&<parameter>...]

Parameters:

Parameter	Description	Example
amount	Amount to transfer	POST /funds?action=transfer&from-id=2&to-id=3&amount=1000
from-allocation	The credits will be transferred from the specified allocation only	POST /funds?action=transfer&from-allocation=2&to-id=3&amount=1000
from-id	Fund to be debited	POST /funds?action=transfer&from-id=2&to-id=3&amount=1000
to-allocation	The credits will be transferred to the specified allocation only	POST /funds?action=transfer&from-id=2&to-allocation=3&amount=1000
to-id	Fund to be credited	POST /funds?action=transfer&from-id=2&to-id=3&amount=1000

Example 28-62. Sample Request

```
POST /funds?action=transfer&from-id=2&to-id=1&amount=1000
```

Example 28-63. Sample Response

```
{
   "code" : "000",
```

```
   "count" : 1000,
   "message" : "Successfully transferred 1000.00 credits from fund 2 to fund 1",
   "status" : "Success"
}
```

Reset a Fund

Synopsis:

POST /funds?action=reset[&<parameter>...]

Parameters:

Parameter	Description	Example
constraint-filter	Restricts the fund to one whose constraints do not conflict with the specified filters	POST /funds?action=reset&constraint-filter=account=chemistry
filter-type	Designates the constraint filter type	POST /funds?action=reset&constraint-filter=account=chemistry&filter-type=ExactMatch
id	Id of the fund to reset	POST /funds?action=reset&id=2

Example 28-64. Sample Request

```
POST /funds?action=reset&id=1
```

Example 28-65. Sample Response

```
{
   "code" : "000",
   "count" : 5000,
   "message" : "Successfully deposited 5000.00 credits into fund 1\nSuccessfully stopped 1
   "status" : "Success"
}
```

Liens

Supported Actions

Action	HTTP Method	Resource
Query liens	GET	/liens[/<id>]
Modify a lien	PATCH	/liens/<id>
Delete a lien	DELETE	/liens/<id>
Query lien allocations	GET	/lien-allocations[/<lien>[/<allocation>]]

Query Liens

Synopsis:

GET /liens[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
active	Displays only unexpired liens	GET /liens?active=true
constraint-filter	Displays leins whose constraints comply with the specified filters	GET /liens?constraint-filter=user=amy
fields	Designates the properties to be returned in the query	GET /liens?fields=id,amount
filter	Filters the objects to be returned in the query	GET /liens?filter=usage-record=1
filter-type	Designates the constraint filter type	GET /liens?constraint-filter=user=amy&filter-type=ImpingesUpon
limit	Limits the results to the number of objects specified	GET /liens?limit=100
offset	Number of objects to skip before starting to return data	GET /liens?offset=100
show-hidden	Includes hidden attributes in the result	GET /liens?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /liens?fields=usage-record&unique=true

Example 28-66. Sample Request

```
GET /liens?active=true
```

Example 28-67. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : null,
         "duration" : 600,
         "end-time" : "2016-06-15 18:39:47",
         "id" : 1,
         "instance" : "24809",
         "start-time" : "2016-06-15 18:29:47",
         "usage-record" : 1
      }
   ],
   "status" : "Success"
}
```

Modify a Lien

Synopsis:

PATCH /liens/<id>

{

<name> : <value>,...

}

Example 28-68. Sample Request

```
PATCH /liens/1
{
   "end-time" : "2016-06-16"
}
```

Example 28-69. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : null,
         "duration" : 600,
         "end-time" : "2016-06-16",
         "id" : 1,
         "instance" : "24809",
         "start-time" : "2016-06-15 18:29:47",
```

```
         "usage-record" : 1
      }
   ],
   "message" : "Successfully modified 1 lien",
   "status" : "Success"
}
```

Delete a Lien

Synopsis:

DELETE /liens/<id>

Example 28-70. Sample Request

```
DELETE /liens/2
```

Example 28-71. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : null,
         "duration" : 600,
         "end-time" : "2016-06-15 18:39:47",
         "id" : 1,
         "instance" : "24809",
         "start-time" : "2016-06-15 18:29:47",
         "usage-record" : 1
      }
   ],
   "message" : "Successfully deleted 1 lien",
   "status" : "Success"
}
```

Query Lien Allocations

Synopsis:

GET /lien-allocations[/<lien>[/<allocation>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /lien-allocations/chemistry?fields=sum(amount),group-by(fund)
filter	Filters the objects to be returned in the query	GET /lien-allocations?filter=fund=4
limit	Limits the results to the number of objects specified	GET /lien-allocations?limit=100
offset	Number of objects to skip before starting to return data	GET /lien-allocations?offset=100
show-hidden	Includes hidden attributes in the result	GET /lien-allocations?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /lien-allocations?fields=fund&unique=true

Example 28-72. Sample Request

```
GET /lien-allocations?fields=sum(amount),group-by(fund)
```

Example 28-73. Sample Response

```
{
   "code" : "000",
   "count" : 2,
   "data" : [
      {
         "amount" : 2,
         "fund" : 2
      },
      {
         "amount" : 10.56,
         "fund" : 4
      }
   ],
   "status" : "Success"
}
```

Organizations

Supported Actions

Action	HTTP Method	Resource
Query organizations	GET	/organizations[/<name>]
Create an organization	POST	/organizations
Modify an organization	PATCH	/organizations/<name>
Delete an organization	DELETE	/organizations/<name>

Query Organizations

Synopsis:

GET /organizations[/<name>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /organizations?fields=name
filter	Filters the objects to be returned in the query	GET /organizations?filter=name~sci*
limit	Limits the results to the number of objects specified	GET /organizations?limit=100
offset	Number of objects to skip before starting to return data	GET /organizations?offset=100
show-hidden	Includes hidden attributes in the result	GET /organizations?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /organizations?fields=name&unique=true

Example 28-74. Sample Request

```
GET /organizations/sciences
```

Example 28-75. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
```

```
      {
         "description" : "Sciences College",
         "name" : "sciences"
      }
   ],
   "status" : "Success"
}
```

Create an Organization

Synopsis:

POST /organizations

{

<name> : <value>,...

}

Example 28-76. Sample Request

```
POST /organizations
{
   "description" : "Sciences College",
   "name" : "sciences"
}
```

Example 28-77. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Sciences College",
         "name" : "sciences"
      }
   ],
   "message" : "Successfully created 1 organization",
   "status" : "Success"
}
```

Modify an Organization

Synopsis:

PATCH /organizations/<name>

{
  <name> : <value>,...
}

Example 28-78. Sample Request

```
PATCH /organizations/sciences
{
   "description" : "Sciences Department"
}
```

Example 28-79. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Sciences Department",
         "name" : "sciences"
      }
   ],
   "message" : "Successfully modified 1 organization",
   "status" : "Success"
}
```

Delete an Organization

Synopsis:

DELETE /organizations/<name>

Example 28-80. Sample Request

```
DELETE /organizations/sciences
```

Example 28-81. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Sciences College",
         "name" : "sciences"
      }
```

```
   ],
   "message" : "Successfully deleted 1 organization",
   "status" : "Success"
}
```

Quotes

Supported Actions

Action	HTTP Method	Resource
Query quotes	GET	/quotes[/<id>]
Modify a quote	PATCH	/quotes/<id>
Delete a quote	DELETE	/quotes/<id>
Query quote charge rates	GET	/quote-charge-rates[/<quote>[/<name>[/<value>]]]

Query Quotes

Synopsis:

GET /quotes[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
active	Displays only unexpired quotes	GET /quotes?active=true
constraint-filter	Displays quotes whose constraints comply with the specified filters	GET /quotes?constraint-filter=user=amy
fields	Designates the properties to be returned in the query	GET /quotes?fields=id,amount
filter	Filters the objects to be returned in the query	GET /quotes?filter=usage-record=1
limit	Limits the results to the number of objects specified	GET /quotes?limit=100
offset	Number of objects to skip before starting to return data	GET /quotes?offset=100
show-hidden	Includes hidden attributes in the result	GET /quote?show-hidden=true

Parameter	Description	Example
unique	Displays only unique results (like DISTINCT in SQL)	GET /quotes?fields=usage-record&unique=true

Example 28-82. Sample Request

```
GET /quotes?active=true
```

Example 28-83. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 0.56,
         "description" : null,
         "duration" : 1000,
         "end-time" : "2016-08-23 18:16:18",
         "id" : 1,
         "instance" : "j1",
        "pinned" : true,
        "start-time" : "2016-08-23 17:59:38",
         "usage-record" : 12
      }
   ],
   "status" : "Success"
}
```

Modify a Quote

Synopsis:

PATCH /quotes/<id>
{
<name> : <value>,...
}

Example 28-84. Sample Request

```
PATCH /quotes/1
{
   "end-time" : "2016-08-24"
}
```

Example 28-85. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 0.56,
         "description" : null,
         "duration" : 1000,
         "end-time" : "2016-08-24",
         "id" : 1,
         "instance" : "j1",
        "pinned" : true,
        "start-time" : "2016-08-23 17:59:38",
         "usage-record" : 12
      }
   ],
   "message" : "Successfully modified 1 quote",
   "status" : "Success"
}
```

Delet an Quote

Synopsis:

DELETE /quotes/<id>

Example 28-86. Sample Request

```
DELETE /quotes/1
```

Example 28-87. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 0.56,
         "description" : null,
         "duration" : 1000,
         "end-time" : "2016-08-23 18:16:18",
         "id" : 1,
         "instance" : "j1",
         "pinned" : true,
         "start-time" : "2016-08-23 17:59:38",
         "usage-record" : 12
      }
```

```
   ],
   "message" : "Successfully deleted 1 quote",
   "status" : "Success"
}
```

Query Quote Charge Rates

Synopsis:

GET /quote-charge-rates[/<quote>[/<name>[/<value>]]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /quote-charge-rates?fields=name
filter	Filters the objects to be returned in the query	GET /quote-charge-rates?filter=name=Processors
limit	Limits the results to the number of objects specified	GET /quote-charge-rates?limit=100
offset	Number of objects to skip before starting to return data	GET /quote-charge-rates?offset=100
show-hidden	Includes hidden attributes in the result	GET /quote-charge-rates?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /quote-charge-rates?fields=name&unique=true

Example 28-88. Sample Request

```
GET /quote-charge-rates/1
```

Example 28-89. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : "1/h",
         "name" : "Processors",
         "quote" : 1,
         "value" : null
```

```
      }
   ],
   "status" : "Success"
}
```

Transactions

Supported Actions

Action	HTTP Method	Resource
Query transactions	GET	/transactions[/<id>]

Query Transactions

Synopsis:

GET /transactions[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /transactions?filter=action=Charge&fields=sum(amount)
filter	Filters the objects to be returned in the query	GET /transactions?filter=action=Charge
limit	Limits the results to the number of objects specified	GET /transactions?limit=100
offset	Number of objects to skip before starting to return data	GET /transactions?offset=100
show-hidden	Includes hidden attributes in the result	GET /transactions?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /transactions?fields=account&unique=true

Example 28-90. Sample Request

```
GET /transactions?filter=usage-record=1,action=Charge
```

Example 28-91. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "action" : "Charge",
         "actor" : "root",
         "allocation" : 2,
         "amount" : 1,
         "balance" : 2999,
         "child" : "24809",
         "count" : 1,
         "delta" : -1,
         "description" : null,
         "duration" : 300,
         "fund" : 2,
         "id" : 334,
         "instance" : "24809",
         "key" : "1",
         "machine" : "colony",
         "object" : "UsageRecord",
         "remaining" : 2999,
         "usage-record" : 1,
         "user" : "amy"
      }
   ],
   "status" : "Success"
}
```

Usage Records

Supported Actions

Action	HTTP Method	Resource
Query usage records	GET	/usage-records[/<id>]
Create a usage record	POST	/usage-records
Modify a usage record	PATCH	/usage-records/<id>
Delete a usage record	DELETE	/usage-records/<id>
Quote for usage	POST	/usage-records?action=quote
Reserve for usage	POST	/usage-records?action=reserve
Charge for usage	POST	/usage-records?action=charge
Refund usage	POST	/usage-records?action=refund

Query Usage Records

Synopsis:

GET /usage-records[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /usage-records?fields=account,charge
filter	Filters the objects to be returned in the query	GET /usage-records?filter=instance=24809
limit	Limits the results to the number of objects specified	GET /usage-records?limit=100
offset	Number of objects to skip before starting to return data	GET /usage-records?offset=100
show-hidden	Includes hidden attributes in the result	GET /usage-records?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /usage-records?fields=account&unique=true

Example 28-92. Sample Request

```
GET /usage-records?filter=instance=24809
```

Example 28-93. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "c-p-u-time" : 1800,
         "charge" : 0,
         "class" : "batch",
         "description" : null,
         "duration" : 300,
         "end-time" : "2016-06-15 18:34:47",
         "exit-code" : null,
         "group" : "research",
         "id" : 1,
         "instance" : "24809",
```

```
         "licenses" : null,
         "machine" : "colony",
         "memory" : null,
         "metrics" : null,
         "nodes" : 1,
         "organization" : "sciences",
         "processors" : 12,
         "quality-of-service" : "normal",
         "requested-duration" : 600,
         "resources" : "{\"gres\":1,\"color\":2}",
         "stage" : null,
         "start-time" : "2016-06-15 18:29:47",
         "submit-time" : null,
         "type" : "Job",
         "user" : "amy",
         "variables" : null
      }
   ],
   "status" : "Success"
}
```

Create a Usage Record

Synopsis:

POST /usage-records

{

<name> : <value>,...

}

Example 28-94. Sample Request

```
POST /usage-records
{
   "account" : "chemistry",
   "c-p-u-time" : 1800,
   "class" : "batch",
   "duration" : 300,
   "end-time" : "2016-06-15 18:34:47",
   "group" : "research",
   "instance" : "24809",
   "machine" : "colony",
   "nodes" : 1,
   "organization" : "sciences",
   "processors" : 12,
   "quality-of-service" : "normal",
   "requested-duration" : 600,
   "resources" : "{\"gres\":1,\"color\":2}",
   "start-time" : "2016-06-15 18:29:47",
   "type" : "Job",
```

```
   "user" : "amy",
}
```

Example 28-95. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "c-p-u-time" : 1800,
         "charge" : 0,
         "class" : "batch",
         "description" : null,
         "duration" : 300,
         "end-time" : "2016-06-15 18:34:47",
         "exit-code" : null,
         "group" : "research",
         "id" : 1,
         "instance" : "24809",
         "licenses" : null,
         "machine" : "colony",
         "memory" : null,
         "metrics" : null,
         "nodes" : 1,
         "organization" : "sciences",
         "processors" : 12,
         "quality-of-service" : "normal",
         "requested-duration" : 600,
         "resources" : "{\"gres\":1,\"color\":2}",
         "stage" : null,
         "start-time" : "2016-06-15 18:29:47",
         "submit-time" : null,
         "type" : "Job",
         "user" : "amy",
         "variables" : null
      }
   ],
   "message" : "Successfully created 1 usage-record",
   "status" : "Success"
}
```

Modify a Usage Record

Synopsis:

PATCH /usage-records/<id>

{

  <name> : <value>,...
}

Example 28-96. Sample Request

```
PATCH /usage-records/1
{
   "group" : "staff"
}
```

Example 28-97. Sample Response


```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "c-p-u-time" : 1800,
         "charge" : 0,
         "class" : "batch",
         "description" : null,
         "duration" : 300,
         "end-time" : "2016-06-15 18:34:47",
         "exit-code" : null,
         "group" : "staff",
         "id" : 1,
         "instance" : "24809",
         "licenses" : null,
         "machine" : "colony",
         "memory" : null,
         "metrics" : null,
         "nodes" : 1,
         "organization" : "sciences",
         "processors" : 12,
         "quality-of-service" : "normal",
         "requested-duration" : 600,
         "resources" : "{\"gres\":1,\"color\":2}",
         "stage" : null,
         "start-time" : "2016-06-15 18:29:47",
         "submit-time" : null,
         "type" : "Job",
         "user" : "amy",
         "variables" : null
      }
   ],
   "message" : "Successfully modified 1 usage record",
   "status" : "Success"
}
```

Delete a Usage Record

Synopsis:

DELETE /usage-records/<id>

Example 28-98. Sample Request

```
DELETE /usage-rcocrds/1
```

Example 28-99. Sample Response


```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "account" : "chemistry",
         "c-p-u-time" : 1800,
         "charge" : 0,
         "class" : "batch",
         "description" : null,
         "duration" : 300,
         "end-time" : "2016-06-15 18:34:47",
         "exit-code" : null,
         "group" : "research",
         "id" : 1,
         "instance" : "24809",
         "licenses" : null,
         "machine" : "colony",
         "memory" : null,
         "metrics" : null,
         "nodes" : 1,
         "organization" : "sciences",
         "processors" : 12,
         "quality-of-service" : "normal",
         "requested-duration" : 600,
         "resources" : "{\"gres\":1,\"color\":2}",
         "stage" : null,
         "start-time" : "2016-06-15 18:29:47",
         "submit-time" : null,
         "type" : "Job",
         "user" : "amy",
         "variables" : null
      }
   ],
   "message" : "Successfully deleted 1 usage record",
   "status" : "Success"
}
```

Quote for Usage

Synopsis:

POST /usage-records?action=quote[&<parameter>...]

```
{
  <name> : <value>,...
}
```

Parameters:

Parameter	Description	Example

charge	Specifies the quote amount if calculated externally	POST /usage-records?action=quote&charge=1
		{
		"instance" : "j1"
		}

cost-only	Returns the cost, ignoring all balance and validity checks	POST /usage-records?action=quote&cost-only=true
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

duration	Incremental duration for the quote in seconds	POST /usage-records?action=quote&duration=3600
		{
		"processors" : 1
		}

end-time	End time for the quote	POST /usage-records?action=quote&start-time=2016-08-23&end-time=2016-08-24
		{
		"processors" : 1
		}

grace-duration	Grace period in seconds	POST /usage-records?action=quote&id=1&duration=3600&grace
		duration=3600
		{
		"processors" : 1
		}

Parameter	Description	Example

id	Usage record for the quote (if usage record already created)	POST /usage-records?action=quote&id=1
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

itemize	Returns the composite charge information in the response data	POST /usage-records?action=quote&itemize=true
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

quote	Quote template used to override standard charge rates	POST /usage-records?action=quote&quote=1
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

rate	Uses the specified charge rate in the quote	POST /usage-records?action=quote&rate=Processors=2/h
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

start-time	Start time for the quote	POST /usage-records?action=quote&start-time=2016-08-23&duration=3600
		{
		"processors" : 1
		}

Example 28-100. Sample Request

```
POST /usage-records?action=quote
{
   "account" : "chemistry",
   "class" : "batch",
   "group" : "research",
   "machine" : "colony",
   "nodes" : 1,
   "processors" : 12,
   "quality-of-service" : "normal",
   "requested-duration" : 600,
   "user" : "amy"
}
```

Example 28-101. Sample Response

```
{
   "code" : "000",
   "count" : 2,
   "data" : [
      {
         "amount" : 2
      }
   ],
   "message" : "Successfully quoted 2.00 credits",
   "status" : "Success"
}
```

Reserve for Usage

Synopsis:

POST /usage-records?action=reserve[&<parameter>...]

```
{
  <name> : <value>,...
}
```

Parameters:

Parameter	Description	Example

charge	Specifies the lien amount if calculated externally	POST /usage-records?action=reserve&charge=1
		{
		"instance" : "j1"
		}

duration	Incremental duration for the lien in seconds	POST /usage-records?action=reserve&duration=3600
		{
		"processors" : 1
		}

end-time	End time for the lien	POST /usage-records?action=reserve&start-time=2016-08-23&end-time=2016-08-24
		{
		"processors" : 1
		}

Parameter	Description	Example

grace-duration	Grace period in seconds	POST /usage-records?action=reserve&id=1&duration=3600&gra
		duration=3600
		{
		"processors" : 1
		}

id	Usage record for the lien (if already created)	POST /usage-records?action=reserve&id=1
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

itemize	Returns the composite charge information in the response data	POST /usage-records?action=reserve&itemize=true
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

modify	Augments existing liens instead of creating new ones	POST /usage-records?action=reserve&modify=true
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

rate	Uses the specified charge rate in the lien	POST /usage-records?action=reserve&rate=Processors=2/h
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

replace	Similarly named liens will be deleted before this lien is created	POST /usage-records?action=reserve&replace=true
		{
		"processors" : 1,
		"requested-duration" : 3600
		}

start-time	Start time for the lien	POST /usage-records?action=reserve&start-time=2016-08-23&duration=3600
		{
		"processors" : 1
		}

Example 28-102. Sample Request

```
POST /usage-records?action=reserve
{
   "account" : "chemistry",
   "class" : "batch",
   "group" : "research",
   "instance" : "j1",
   "machine" : "colony",
   "nodes" : 1,
   "processors" : 12,
   "quality-of-service" : "normal",
   "requested-duration" : 600,
   "user" : "amy"
}
```

Example 28-103. Sample Response

```
{
   "code" : "000",
   "count" : 2,
   "data" : [
      {
         "amount" : 2,
         "instance" : "j1",
         "lien" : 17,
         "usage-record" : 14
      }
   ],
   "message" : "Successfully reserved 2.00 credits with lien id 17 for instance j1 and crea
   "status" : "Success"
}
```

Charge for Usage

Synopsis:

POST /usage-records?action=charge[&<parameter>...]

{

<name> : <value>,...

}

Parameters:

Parameter	Description	Example

Parameter	Description	Example

charge	Specifies the charge amount if calculated externally	POST /usage-records?action=charge&charge=1
		{
		"instance" : "j1"
		}

duration	Incremental duration for the lien in seconds	POST /usage-records?action=charge&duration=3600
		{
		"processors" : 1
		}

end-time	End time for the charge	POST /usage-records?action=charge&start-time=2016-08-23&end-time=2016-08-24
		{
		"processors" : 1
		}

fund	fund to charge	POST /usage-records?action=charge&fund=2
		{
		"processors" : 1,
		"duration" : 3600
		}

id	Usage record for the charge (if already created)	POST /usage-records?action=charge&id=1
		{
		"processors" : 1,
		"duration" : 3600
		}

incremental	Any associated liens will be debited instead of removed	POST /usage-records?action=charge&incremental=true
		{
		"processors" : 1,
		"duration" : 3600
		}

itemize	Returns the composite charge information in the response data	POST /usage-records?action=charge&itemize=true
		{
		"processors" : 1,
		"duration" : 3600
		}

Parameter	Description	Example

rate	Uses the specified charge rate in the charge	POST /usage-records?action=charge&rate=Processors=2/h
		{
		"processors" : 1,
		"duration" : 3600
		}

start-time	Start time for the charge	POST /usage-records?action=charge&start-time=2016-08-23&duration=3600
		{
		"processors" : 1
		}

Example 28-104. Sample Request

```
POST /usage-records?action=charge
{
   "account" : "chemistry",
   "class" : "batch",
   "c-p-u-time" : 1800,
   "duration" : 300,
   "end-time" : "2016-06-15 18:34:47",
   "group" : "research",
   "instance" : "j1",
   "machine" : "colony",
   "nodes" : 1,
   "processors" : 12,
   "quality-of-service" : "normal",
   "start-time" : "2016-06-15 18:29:47",
   "user" : "amy"
}
```

Example 28-105. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 1,
         "instance" : "j1",
         "usage-record" : 15
      }
   ],
   "message" : "Successfully charged 1.00 credits for instance j1 and created usage record
   "status" : "Success"
}
```

Refund Usage

Synopsis:

POST /usage-records?action=refund[&<parameter>...]

Parameters:

Parameter	Description	Example
allocation	Allocation to be credited	POST /usage-records?action=refund&id=1&allocation=2
amount	Amount to refund	POST /usage-records?action=refund&id=1&amount=0.5
id	Usage record to be refunded	POST /usage-records?action=refund&id=1
instance	Instance to be refunded	POST /usage-records?action=refund&instance=j1

Example 28-106. Sample Request

```
POST /usage-records?action=refund&instance=j1
```

Example 28-107. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "amount" : 1,
         "id" : "1",
         "instance" : "j1"
      }
   ],
   "message" : "Successfully refunded 1.00 credits to usage record 1 for instance j1",
   "status" : "Success"
}
```

Users

Supported Actions

Action	HTTP Method	Resource
Query users	GET	/users[/<name>]
Create a user	POST	/users
Modify a user	PATCH	/users/<name>
Delete a user	DELETE	/users/<name>

Query Users

Synopsis:

GET /users[/<name>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
constraint-filter	Applies meta-filters to the query (account: include only users associated with the specified account)	GET /users?constraint-filter=account=chemistry
fields	Designates the properties to be returned in the query	GET /users?fields=name,email-address
filter	Filters the objects to be returned in the query	GET /users?filter=active=true
limit	Limits the results to the number of objects specified	GET /users?limit=100
offset	Number of objects to skip before starting to return data	GET /users?offset=100
show-hidden	Includes hidden attributes in the result	GET /users?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /users?fields=default-account&unique=true

Example 28-108. Sample Request

```
GET /users/amy
```

Example 28-109. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "common-name" : "Amy Miller",
         "default-account" : "chemistry",
         "description" : null,
         "email-address" : "amy@hpc.com",
         "name" : "amy",
         "phone-number" : "(801) 555-1437"
      }
   ],
   "status" : "Success"
}
```

Create a User

Synopsis:

POST /users

{

<name> : <value>,...

}

Example 28-110. Sample Request

```
POST /users
{
   "common-name" : "Amy Miller",
   "default-account" : "chemistry",
   "email-address" : "amy@hpc.com",
   "name" : "amy",
   "phone-number" : "(801) 555-1437"
}
```

Example 28-111. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "common-name" : "Amy Miller",
```

```
         "default-account" : "chemistry",
         "description" : null,
         "email-address" : "amy@hpc.com",
         "name" : "amy",
         "phone-number" : "(801) 555-1437"
      }
   ],
   "message" : "Successfully created 1 user",
   "status" : "Success"
}
```

Modify a User

Synopsis:

PATCH /users/<name>
{
<name> : <value>,...
}

Example 28-112. Sample Request

```
PATCH /users/amy
{
   "email-address" : "amy@htc.org"
}
```

Example 28-113. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "common-name" : "Amy Miller",
         "default-account" : "chemistry",
         "description" : null,
         "email-address" : "amy@htc.org",
         "name" : "amy",
         "phone-number" : "(801) 555-1437"
      }
   ],
   "message" : "Successfully modified 1 user",
   "status" : "Success"
}
```

Delete a User

Synopsis:

DELETE /users/<name>

Example 28-114. Sample Request

```
DELETE /users/amy
```

Example 28-115. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "active" : true,
         "common-name" : "Amy Miller",
         "default-account" : "chemistry",
         "description" : null,
         "email-address" : "amy@hpc.com",
         "name" : "amy",
         "phone-number" : "(801) 555-1437"
      }
   ],
   "message" : "Successfully deleted 1 user",
   "status" : "Success"
}
```

Framework Resources

Actions

Supported Actions

Action	HTTP Method	Resource
Query actions	GET	/actions[/<object>[/<name>]]
Create an action	POST	/actions
Modify an action	PATCH	/actions/<object>/<name>
Delete an action	DELETE	/actions/<object>/<name>

Query Actions

Synopsis:

GET /actions[/<object>[/<name>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /actions/UsageRecord?fields=name
filter	Filters the objects to be returned in the query	GET /actions?filter=name=Refund
limit	Limits the results to the number of objects specified	GET /actions?limit=100
offset	Number of objects to skip before starting to return data	GET /actions?offset=100
show-hidden	Includes hidden attributes in the result	GET /actions?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /actions?fields=object&unique=true

Example 28-116. Sample Request

```
GET /actions/UsageRecord/Charge
```

Example 28-117. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Charge for Usage",
         "display" : false,
         "name" : "Charge",
         "object" : "UsageRecord"
      }
   ],
   "status" : "Success"
}
```

Create an Action

Synopsis:

POST /actions
{
<name> : <value>,...
}

Example 28-118. Sample Request

```
POST /actions
{
   "description" : "Modify",
   "name" : "Modify",
   "object" : "UsageRecord"
}
```

Example 28-119. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Modify",
         "display" : false,
         "name" : "Modify",
         "object" : "Transaction"
      }
   ],
   "message" : "Successfully created 1 action",
   "status" : "Success"
}
```

Modify an Action

Synopsis:

PATCH /actions/<object>/<name>
{
<name> : <value>,...
}

Example 28-120. Sample Request

```
PATCH /actions/Transaction/Modify
{
   "display" : true
}
```

Example 28-121. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Modify",
         "display" : true,
         "name" : "Modify",
         "object" : "Transaction"
      }
   ],
   "message" : "Successfully modified 1 action",
   "status" : "Success"
}
```

Delete an Action

Synopsis:

DELETE /actions/<object>/<name>

Example 28-122. Sample Request

```
DELETE /actions/Transaction/Modify
```

Example 28-123. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Modify",
         "display" : false,
         "name" : "Modify",
         "object" : "Transaction"
      }
   ],
```

```
   "message" : "Successfully deleted 1 action",
   "status" : "Success"
}
```

Attributes

Supported Actions

Action	HTTP Method	Resource
Query attributes	GET	/attributes[/<object>[/<name>]]
Create an attribute	POST	/attributes
Modify an attribute	PATCH	/attributes/<object>/<name>
Delete an attribute	DELETE	/attributes/<object>/<name>

Query Attributes

Synopsis:

GET /attributes[/<object>[/<name>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /attributes/UsageRecord?fields=name
filter	Filters the objects to be returned in the query	GET /attributes/ChargeRate?filter=primary-key=True
limit	Limits the results to the number of objects specified	GET /attributes?limit=100
offset	Number of objects to skip before starting to return data	GET /attributes?offset=100
show-hidden	Includes hidden attributes in the result	GET /attributes?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /attributes?fields=name&unique=true

Example 28-124. Sample Request

```
GET /attributes/Account/Organization
```

Example 28-125. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "data-type" : "String",
         "default-value" : null,
         "description" : "Organization",
         "fixed" : false,
         "hidden" : false,
         "name" : "Organization",
         "object" : "Account",
         "primary-key" : false,
         "required" : false,
         "sequence" : 30,
         "values" : "@!=Organization"
      }
   ],
   "status" : "Success"
}
```

Create an Attribute

Synopsis:

POST /attributes
{
<name> : <value>,...
}

Example 28-126. Sample Request

```
POST /attributes
{
   "data-type" : "String",
   "description" : "Organization",
   "name" : "Organization",
   "object" : "Account",
   "values" : "@!=Organization"
}
```

Example 28-127. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "data-type" : "String",
         "default-value" : null,
         "description" : "Organization",
         "fixed" : false,
         "hidden" : false,
         "name" : "Organization",
         "object" : "Account",
         "primary-key" : false,
         "required" : false,
         "sequence" : 30,
         "values" : "@!=Organization"
      }
   ],
   "message" : "Successfully created 1 attribute",
   "status" : "Success"
}
```

Modify an Attribute

Synopsis:

PATCH /attributes/<object>/<name>

```
{
  <name> : <value>,...
}
```

Example 28-128. Sample Request

```
PATCH /attributes/Account/Organization
{
   "default-value" : "university"
}
```

Example 28-129. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "data-type" : "String",
```

```
         "default-value" : "university",
         "description" : "Organization",
         "fixed" : false,
         "hidden" : false,
         "name" : "Organization",
         "object" : "Account",
         "primary-key" : false,
         "required" : false,
         "sequence" : 30,
         "values" : "@!=Organization"
      }
   ],
   "message" : "Successfully modified 1 attribute",
   "status" : "Success"
}
```

Delete an Attribute

Synopsis:

DELETE /attributes/<object>/<name>

Example 28-130. Sample Request

```
DELETE /attributes/Account/Organization
```

Example 28-131. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "data-type" : "String",
         "default-value" : null,
         "description" : "Organization",
         "fixed" : false,
         "hidden" : false,
         "name" : "Organization",
         "object" : "Account",
         "primary-key" : false,
         "required" : false,
         "sequence" : 30,
         "values" : "@!=Organization"
      }
   ],
   "message" : "Successfully deleted 1 attribute",
   "status" : "Success"
```

```
}
```

Events

Supported Actions

Action	HTTP Method	Resource
Query events	GET	/events[/<id>]
Create an event	POST	/events
Modify an event	PATCH	/events/<id>
Delete an event	DELETE	/events/<id>

Query Events

Synopsis:

GET /events[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /events?fields=id,description
filter	Filters the objects to be returned in the query	GET /events?filter=fire-time>now
limit	Limits the results to the number of objects specified	GET /events?limit=100
offset	Number of objects to skip before starting to return data	GET /events?offset=100
show-hidden	Includes hidden attributes in the result	GET /events?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /events?fields=rearm-period&unique=true

Example 28-132. Sample Request

```
GET /events/1
```

Example 28-133. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "arm-time" : "2016-05-31 16:29:05",
         "catch-up" : false,
         "description" : "Delete Stale Notifications",
         "end-time" : null,
         "failure-command" : null,
         "fire-command" : "Notification Refresh",
         "fire-time" : "2016-05-31 16:29:05",
         "id" : 1,
         "notify" : "Store:",
         "rearm-on-failure" : true,
         "rearm-period" : "1 day @ hour 2"
      }
   ],
   "status" : "Success"
}
```

Create an Event

Synopsis:

POST /events

{

<name> : <value>,...

}

Example 28-134. Sample Request

```
POST /events
{
   "catch-up" : false,
   "description" : "Delete Stale Notifications",
   "fire-command" : "Notification Refresh",
   "fire-time" : "Now",
   "rearm-on-failure" : true,
   "rearm-period" : "1 day @ hour 2"
}
```

Example 28-135. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "arm-time" : "2016-05-31 16:29:05",
         "catch-up" : false,
         "description" : "Delete Stale Notifications",
         "end-time" : null,
         "failure-command" : null,
         "fire-command" : "Notification Refresh",
         "fire-time" : "2016-05-31 16:29:05",
         "id" : 1,
         "notify" : "Store:",
         "rearm-on-failure" : true,
         "rearm-period" : "1 day @ hour 2"
      }
   ],
   "message" : "Successfully created 1 event",
   "status" : "Success"
}
```

Modify an Event

Synopsis:

PATCH /events/<id>
{
<name> : <value>,...
}

Example 28-136. Sample Request

```
PATCH /events/1
{
   "rearm-period" : "12 hours^"
}
```

Example 28-137. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "arm-time" : "2016-05-31 16:29:05",
```

```
            "catch-up" : false,
            "description" : "Delete Stale Notifications",
            "end-time" : null,
            "failure-command" : null,
            "fire-command" : "Notification Refresh",
            "fire-time" : "2016-05-31 16:29:05",
            "id" : 1,
            "notify" : "Store:",
            "rearm-on-failure" : true,
            "rearm-period" : "12 hours^"
         }
      ],
      "message" : "Successfully modified 1event",
      "status" : "Success"
}
```


Delete an Event

Synopsis:

DELETE /events/<id>

Example 28-138. Sample Request

```
DELETE /events/1
```

Example 28-139. Sample Response


```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "arm-time" : "2016-05-31 16:29:05",
         "catch-up" : false,
         "description" : "Delete Stale Notifications",
         "end-time" : null,
         "failure-command" : null,
         "fire-command" : "Notification Refresh",
         "fire-time" : "2016-05-31 16:29:05",
         "id" : 1,
         "notify" : "Store:",
         "rearm-on-failure" : true,
         "rearm-period" : "1 day @ hour 2"
      }
   ],
   "message" : "Successfully deleted 1 event",
   "status" : "Success"
```

```
}
```

Notifications

Supported Actions

Action	HTTP Method	Resource
Query notifications	GET	/notifications[/<id>]
Delete a notification	DELETE	/notifications/<id>

Query Notifications

Synopsis:

GET /notifications[/<id>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /notifications?fields=message
filter	Filters the objects to be returned in the query	GET /notifications?filter=status=Failure
limit	Limits the results to the number of objects specified	GET /notifications?limit=100
offset	Number of objects to skip before starting to return data	GET /notifications?offset=100
show-hidden	Includes hidden attributes in the result	GET /notifications?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /notifications?fields=type&unique=true

Example 28-140. Sample Request

```
GET /notifications/1
```

Example 28-141. Sample Response

```
{
```

```
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "code" : "000",
         "end-time" : "2016-09-23 13:55:00",
         "event" : 1,
         "id" : 1,
         "key" : null,
         "message" : "No stale events were located for deletion",
         "recipient" : null,
         "status" : "Success",
         "type" : "Fire"
      }
   ],
   "status" : "Success"
}
```


Delete a Notification

Synopsis:

DELETE /notifications/<id>

Example 28-142. Sample Request

```
DELETE /notifications/1
```

Example 28-143. Sample Response


```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "code" : "000",
         "end-time" : "2016-09-23 13:55:00",
         "event" : 1,
         "id" : 1,
         "key" : null,
         "message" : "No stale events were located for deletion",
         "recipient" : null,
         "status" : "Success",
         "type" : "Fire"
      }
   ],
   "message" : "Successfully deleted 1 notification",
   "status" : "Success"
```

```
}
```

Objects

Supported Actions

Action	HTTP Method	Resource
Query objects	GET	/objects[/<name>]
Create an object	POST	/objects
Modify an object	PATCH	/objects/<name>
Delete an object	DELETE	/objects/<name>

Query Objects

Synopsis:

GET /objects[/<name>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /objects?fields=name
filter	Filters the objects to be returned in the query	GET /objects?filter=association=True
limit	Limits the results to the number of objects specified	GET /objects?limit=100
offset	Number of objects to skip before starting to return data	GET /objects?offset=100
show-hidden	Includes hidden attributes in the result	GET /objects?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /objects?fields=child&unique=true

Example 28-144. Sample Request

```
GET /objects/Organization
```

Example 28-145. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "association" : false,
         "auto-gen" : true,
         "child" : null,
         "default-value" : null,
         "description" : "Virtual Organization",
         "name" : "Organization",
         "parent" : null"
      }
   ],
   "status" : "Success"
}
```

Create an Object

Synopsis:

POST /objects

{

<name> : <value>,...

}

Example 28-146. Sample Request

```
POST /objects
{
   "auto-gen" : true,
   "description" : "Virtual Organization",
   "name" : "Organization",
}
```

Example 28-147. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "association" : false,
         "auto-gen" : true,
         "child" : null,
         "default-value" : null,
```

```
         "description" : "Virtual Organization",
         "name" : "Organization",
         "parent" : null"
      }
   ],
   "message" : "Successfully created 1 object",
   "status" : "Success"
}
```

Modify an Object

Synopsis:

PATCH /objects/<name>
{
<name> : <value>,...
}

Example 28-148. Sample Request

```
PATCH /objects/Organization
{
   "auto-gen" : false
}
```

Example 28-149. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "association" : false,
         "auto-gen" : false,
         "child" : null,
         "default-value" : null,
         "description" : "Virtual Organization",
         "name" : "Organization",
         "parent" : null"
      }
   ],
   "message" : "Successfully modified 1 object",
   "status" : "Success"
}
```

Delete an Object

Synopsis:

DELETE /objects/<name>

Example 28-150. Sample Request

```
DELETE /objects/Organization
```

Example 28-151. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "association" : false,
         "auto-gen" : true,
         "child" : null,
         "default-value" : null,
         "description" : "Virtual Organization",
         "name" : "Organization",
         "parent" : null"
      }
   ],
   "message" : "Successfully deleted 1 object",
   "status" : "Success"
}
```

Passwords

Supported Actions

Action	HTTP Method	Resource
Query passwords	GET	/passwords[/<user>]
Create a password	POST	/passwords
Modify a password	PATCH	/passwords/<user>
Delete a password	DELETE	/passwords/<user>

Query Passwords

Synopsis:

GET /passwords[/<user>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET passwords?fields=user
filter	Filters the objects to be returned in the query	GET /passwords?filter=user~a*
limit	Limits the results to the number of objects specified	GET /passwords?limit=100
offset	Number of objects to skip before starting to return data	GET /passwords?offset=100
show-hidden	Includes hidden attributes in the result	GET /passwords?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /passwords?fields=user&unique=true

Example 28-152. Sample Request

```
GET /passwords/amy
```

Example 28-153. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "password" : "LWL9zkOOyv1ekGCRfFuuMeOHp4EtRdjX",
         "user" : "amy"
      }
   ],
   "status" : "Success"
}
```

Create a Password

Synopsis:

POST /passwods
{
<name> : <value>,...
}

Example 28-154. Sample Request

```
POST /passwords
{
   "password" : "changeme!",
   "user" : "amy"
}
```

Example 28-155. Sample Response

```
{
   "code" : "080",
   "count" : 1,
   "data" : [
      {
         "password" : "LWL9zkOOyv1ekGCRfFuuMeOHp4EtRdjX",
         "user" : "amy"
      }
   ],
   "message" : "Successfully created 1 password",
   "status" : "Success"
}
```

Modify a Password

Synopsis:

PATCH /passwords/<name>
{
<name> : <value>,...
}

Example 28-156. Sample Request

```
PATCH /passwords/amy
{
   "password" : "changeme2"
}
```

Example 28-157. Sample Response

```
{
   "code" : "080",
   "count" : 1,
   "data" : [
      {
         "password" : "TDB5dM5sKdpti8N730cMWxoJx6XUksq1",
         "user" : "amy"
      }
   ],
   "message" : "Successfully modified 1 password",
   "status" : "Success"
}
```

Delete a Password

Synopsis:

DELETE /passwords/<name>

Example 28-158. Sample Request

```
DELETE /passwords/amy
```

Example 28-159. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "password" : "LWL9zkOOyv1ekGCRfFuuMeOHp4EtRdjX",
         "user" : "amy"
      }
   ],
   "message" : "Successfully deleted 1 password",
   "status" : "Success"
}
```

Roles

Supported Actions

Action	HTTP Method	Resource
Query roles	GET	/roles[/<name>]
Create a role	POST	/roles
Modify a role	PATCH	/roles/<name>
Delete a role	DELETE	/roles/<name>
Query role actions	GET	/role-actions[/<role>[/<object>[/<name>]]]
Add an action to a role	POST	/role-actions
Remove an action from a role	DELETE	/role-actions/<role>/<object>/<name>
Query role users	GET	/role-users[/<role>[/<user>]]
Add a user to a role	POST	/role-users
Remove a user from a role	DELETE	/role-users/<role>/<user>

Query Roles

Synopsis:

GET /roles[/<name>][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /roles?fields=name
filter	Filters the objects to be returned in the query	GET /roles?filter=name~Account*
limit	Limits the results to the number of objects specified	GET /roles?limit=100
offset	Number of objects to skip before starting to return data	GET /roles?offset=100
show-hidden	Includes hidden attributes in the result	GET /roles?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /roles?fields=name&unique=true

Example 28-160. Sample Request

```
GET /roles/UserServices
```

Example 28-161. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "User Services",
         "name" : "UserServices"
      }
   ],
   "status" : "Success"
}
```

Create a Role

Synopsis:

POST /roles
{
<name> : <value>,...
}

Example 28-162. Sample Request

```
POST /roles
{
   "description" : "User Services",
   "name" : "UserServices"
}
```

Example 28-163. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "User Services",
         "name" : "UserServices"
      }
   ],
   "message" : "Successfully created 1 role",
   "status" : "Success"
}
```

Modify a Role

Synopsis:

PATCH /roles/<name>
{
<name> : <value>,...
}

Example 28-164. Sample Request

```
PATCH /roles/UserServices
{
   "description" : "Help Desk"
}
```

Example 28-165. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Help Desk",
         "name" : "UserServices"
      }
   ],
   "message" : "Successfully modified 1 role",
   "status" : "Success"
}
```

Delete a Role

Synopsis:

DELETE /roles/<name>

Example 28-166. Sample Request

```
DELETE /roles/UserServices
```

Example 28-167. Sample Response

```
{
   "code" : "000",
   "count" : 1,
```

```
   "data" : [
      {
         "description" : "User Services",
         "name" : "UserServices"
      }
   ],
   "message" : "Successfully deleted 1 role",
   "status" : "Success"
}
```

Query Role Actions

Synopsis:

GET /role-actions[/<role>[/<object>[/<name>]]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /role-actions/UserServices?fields=object,name,instance
filter	Filters the objects to be returned in the query	GET /role-actions?filter=object=UsageRecord
limit	Limits the results to the number of objects specified	GET /role-actions?limit=100
offset	Number of objects to skip before starting to return data	GET /role-actions?offset=100
show-hidden	Includes hidden attributes in the result	GET /role-actions?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /role-actions?fields=object&unique=true

Example 28-168. Sample Request

```
GET /role-actions/UserServices/UsageRecord
```

Example 28-169. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
```

```
      {
         "instance" : "ANY",
         "name" : "Refund",
         "object" : "UsageRecord",
         "role" : "UserServices"
      }
   ],
   "status" : "Success"
}
```

Add an Action to a Role

Synopsis:

POST /role-actions
{
<name> : <value>,...
}

Example 28-170. Sample Request

```
POST /role-actions
{
   "name" : "Refund",
   "object" : "UsageRecord",
   "role" : "UserServices"
}
```

Example 28-171. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "instance" : "ANY",
         "name" : "Refund",
         "object" : "UsageRecord",
         "role" : "UserServices"
      }
   ],
   "message" : "Successfully created 1 role action",
   "status" : "Success"
}
```

Remove an Action from a Role

Synopsis:

DELETE /role-actions/<role>/<object>/<name>

Example 28-172. Sample Request

```
DELETE /role-actions/UserServices/UsageRecord/Refund
```

Example 28-173. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "instance" : "ANY",
         "name" : "Refund",
         "object" : "UsageRecord",
         "role" : "UserServices"
       }
   ],
   "message" : "Successfully deleted 1 role action",
   "status" : "Success"
}
```

Query Role Users

Synopsis:

GET /role-users[/<role>[/<user>]][?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /role-users/UserServices?fields=name
filter	Filters the objects to be returned in the query	GET /role-users?filter=name=amy
limit	Limits the results to the number of objects specified	GET /role-users?limit=100
offset	Number of objects to skip before starting to return data	GET /role-users?offset=100

Parameter	Description	Example
show-hidden	Includes hidden attributes in the result	GET /role-users?show-hidden=true
unique	Displays only unique results (like DISTINCT in SQL)	GET /role-users?fields=name&unique=true

Example 28-174. Sample Request

```
GET /role-users/UserServices/amy
```

Example 28-175. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "name" : "amy",
         "role" : "UserServices"
      }
   ],
   "status" : "Success"
}
```

Add a User to a Role

Synopsis:

POST /role-users
{
<name> : <value>,...
}

Example 28-176. Sample Request

```
POST /role-users
{
   "name" : "amy",
   "role" : "UserServices"
}
```

Example 28-177. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "name" : "amy",
         "role" : "UserServices"
      }
   ],
   "message" : "Successfully created 1 role user",
   "status" : "Success"
}
```

Remove a User from a Role

Synopsis:

DELETE /role-users/<role>/<user>

Example 28-178. Sample Request

```
DELETE /role-users/UserServices/amy
```

Example 28-179. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "name" : "amy",
         "role" : "UserServices"
       }
   ],
   "message" : "Successfully deleted 1 role user",
   "status" : "Success"
}
```

System

Supported Actions

Action	HTTP Method	Resource
Query system properties	GET	/system

Query the System

Synopsis:

GET /system[?<parameter>[&<parameter>...]]

Parameters:

Parameter	Description	Example
fields	Designates the properties to be returned in the query	GET /system?fields=version
show-hidden	Includes hidden attributes in the result	GET /system?show-hidden=true

Example 28-180. Sample Request

```
GET /system
```

Example 28-181. Sample Response

```
{
   "code" : "000",
   "count" : 1,
   "data" : [
      {
         "description" : "Commercial Release",
         "name" : "Moab Accounting Manager",
         "version" : "9.1"
      }
   ],
   "status" : "Success"
}
```

Appendix A. Commands Reference

Moab Accounting Manager provides a server daemon and client commands for use by administrators and end-users.

Common Command Options

Most Moab Accounting Manager commands support the following common options.

Table A-1. Common Command Options

Option	Description
--help	brief command option summary
--format *output_format*	data output format. Valid values include standard, raw and csv. • csv -- Fields are delimited by commas (CSV = comma-separated values). Fields containing commas are double-quoted • raw -- Fields are delimited by the pipe character ('|') • standard (default) -- Fields are aligned to fixed-width columns whose widths are dynamically calculated based on the widest value in a column (including the header)
--man	full command documentation
--site *site_name*	obtain response from specified site
--version	display product version

List of Commands

Table A-2. List of Commands

Command	Description
mam-balance	display balance information
mam-charge	create a usage charge
mam-create-account	create a new account
mam-create-chargerate	create a new charge rate
mam-create-event	create a new event

Command	Description
mam-create-fund	create a new fund
mam-create-lien	create a lien
mam-create-organization	create a new organization
mam-create-quote	create a quote template
mam-create-role	create a new role
mam-create-usagerecord	create a new usage record
mam-create-user	create a new user
mam-delete-account	delete an account
mam-delete-allocation	delete an allocation or purge stale allocations
mam-delete-chargerate	delete a charge rate
mam-delete-event	delete an event
mam-delete-fund	delete a fund
mam-delete-lien	delete a lien
mam-delete-notification	delete a stored notification
mam-delete-organization	delete an organization
mam-delete-quote	delete a quote
mam-delete-role	delete a role
mam-delete-usagerecord	delete a usage record
mam-delete-user	delete a user
mam-deposit	issue a deposit
mam-list-accounts	query accounts
mam-list-allocations	query allocations
mam-list-chargerates	query charge rates
mam-list-events	query events
mam-list-funds	query funds
mam-list-itemizedcharges	query charges
mam-list-liens	query liens
mam-list-notifications	query stored notifications
mam-list-organizations	query organizations
mam-list-quotes	query quotes
mam-list-roles	query roles
mam-list-transactions	query transactions
mam-list-usagerecords	query usage records
mam-list-users	query users
mam-modify-account	modify an account
mam-modify-allocation	modify an allocation
mam-modify-chargerate	modify a charge rate
mam-modify-event	modify an event

Command	Description
mam-modify-fund	modify a fund
mam-modify-lien	modify a lien
mam-modify-organization	modify an organization
mam-modify-quote	modify a quote
mam-modify-role	modify a role
mam-modify-usagerecord	modify a usage record
mam-modify-user	modify a user
mam-quote	quote for usage
mam-refund	issue a usage refund
mam-reserve	reserve for usage
mam-set-password	set a user passwd
mam-statement	display fund statement
mam-transfer	issue a transfer
mam-withdraw	issue a withdrawal
mam-server	Moab Accounting Manager server
mam-shell	low-level interactive shell for Moab Accounting Manager
mam-read-configuration	read and print configuration parameters
mybalance	display personal balance information

mam-balance

Synopsis

mam-balance [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] [--ignore-ancestors] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-balance is used to display balance information for funds having active allocations.

Options

-a *account_name*

displays balance available to the specified account

-c *class_name*

displays balance available to the specified class

--filter *filter_name*=*filter_value*

displays balance for funds whose constraints do not conflict with the specified filters. For example mam-balance -f User=amy will display the balance usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

displays balance available to the specified group

--ignore-ancestors

do not include hierarchical ancestor funds in the result

-m *machine_name*

displays balance available to the specified machine

-o *organization_name*

displays balance available to the specified organization

-u *user_name*

displays balance available to the specified user

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Name, Constraints, Balance, Reserved, Effective, CreditLimit, Available, Capacity, Allocated, Used,

PercentRemaining, PercentUsed, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Allocated

adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.

Available

total amount currently available for charging (Balance - Reserved + CreditLimit)

Balance

sum of active allocation amounts remaining within this fund. It does not take into account current liens.

Capacity

total expendable amount (Allocated + CreditLimit)

Constraints

contraints on fund usage

CreationTime

time this fund was created

CreditLimit

sum of active credit limits within this fund

Deleted

boolean indicating whether this fund is deleted or not

Description

fund description

Effective

effective allocation total not blocked by liens (Balance - Reserved)

Id

fund id

ModificationTime

time this fund was last modified

Name

fund name

PercentRemaining

percentage of allocation remaining (Balance * 100 / Capacity)

PercentUsed

percentage of allocation used (Used * 100 / Capacity)

RequestId

id of the last modifying request

Reserved

sum of active lien amounts against this fund

TransactionId

id of the last modifying transaction

Used

total amount used from this allocation (Allocated - Balance)

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-charge

Synopsis

mam-charge {-J *instance_name*} [[-j] *usage_record_id*] [-n *designated_name*] [-q *quote_id*] [-l *lien_id*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [-N *nodes*] [-P *processors*] [-C *cpu_time*] [-M *memory*] [-D *disk*] [-E *energy*] [-F "{\"*feature_name*\":*feature_count*,...}"] [-R "{\"*resource_name*\":*resource_count*,...}"] [-L "{\"*license_name*\":*license_count*,...}"] [-Z "{\"*metric_name*\":*metric_amount*,...}"] [-V "{\"*variable_name*\":\"*variable_value*\",...}"] [-W *requested_duration*] [-t *actual_duration*] [-s *start_time*] [-e *end_time*] [-x *exit_code*] [--stage *lifecycle_stage*] [-d *description*] [-X, --extension *property*=*value*]... [-zt *charge_duration*] [-zs *charge_start_time*] [-z *charge_amount*] [-f *fund_id*] [--incremental] [--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...]... [--hours] [--itemize] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-charge is used to charge for resource usage.

Options

-a *account_name*

account name

-c *class_name*

class of queue used

-C *cpu_time*

CPU time used. *cpu_time* may be an expression of the form: [*cumulative_cpu_time*][(*incremental_cpu_time*)]. If both *incremental_cpu_time* and *cumulative_cpu_time* are specified, *incremental_cpu_time* will be used for the charge while the *cumulative_cpu_time* value will be recorded as the cumulative value used in the usage record. If only *incremental_cpu_time* is specified, this value will be used for the charge only and no cpu time value will be recorded in the usage record. If only *cumulative_cpu_time* is specified, this value will be used both in the charge and recorded in the usage record.

-d *description*

description of the usage

-D *disk*

amount of disk used

-e *end_time*

end time for the usage in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-E *energy*

amount of energy used

-f *fund_id*

fund_id to charge

-F "{\"*feature_name*\":*feature_count*,...}"

allocated node features. Features represent counts of the node features allocated to the job.

-g *group_name*

group name

--incremental

any associated liens will be debited instead of removed

[-j] *usage_record_id*

usage record id for the charge (if already created with mam-create-usagerecord, mam-quote, mam-reserve or a previous mam-charge). This is used to charge an existing usage record if the instance name (e.g. job id) is ambiguous or if a usage has already been debited and you want to charge an additional amount to the same usage record.

-J *instance_name*

instance name (e.g. job id) for the charge (if known). This can sometimes be non-unique (such as when a resource manager recycles job ids), and does not always unambiguously identify a usage record to charge. In such cases, look up and specify the usage record id for the charge.

-l *lien_id*

lien id (used to match up the right usage record id and remove the correct lien if ambiguous)

-L "{\"*license_name*\":*license_count*,...}"

licenses used. Licenses represent software licenses that are used in integer units.

-m *machine_name*

name of the cluster

-M *memory*

amount of memory used

-n *designated_name*

user-specified job name

-N *nodes*

number of nodes used

-o *organization_name*

organization name

-P *processors*

number of processors used

-q *quote_id*

quote used to determine charge rates

-Q *quality_of_service*

quality of service used

-R "{\"*resource_name*\":*resource_count*,...}"

consumable resources allocated. Resources represent consumable resources that may be allocated in integer amounts.

--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...

uses the specified charge rates in the charge. The specified rates override the general rates or rates guaranteed through a quote. Multiple charge rates may be passed to the --rate option in a comma-delimited list. Alternatively, multiple --rate options may be specified.

-s *start_time*

start time for the usage in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-t *actual_duration*

total actual duration in seconds

-T *usage_record_type*

specifies the usage record type (Job, Reservation, etc.)

-u *user_name*

user name

-V "{\"*variable_name*\":\"*variable_value*\",...}"

job variables. Variables represent arbitrary string variables passed into the job.

-W *requested_duration*

total estimated wallclock duration in seconds

-x *exit_code*

exit code

-X, --extension *property*=*value*

extension property. Any number of extra usage properties may be specified with the charge. When expressing accumulating properties, *value* may be an expression of the form: [*cumulative_value*][(*incremental_value*)]. If both *incremental_value* and *cumulative_value* are specified, *incremental_value* will be used for the charge while the *cumulative_value* value will be recorded as the cumulative value used in the usage record. If only *incremental_value* is specified, this value will be used for the charge only and no cumulative value will be recorded in the usage record. If only *cumulative_value* is specified, this value will be used both in the charge and recorded in the usage record.

-z *charge_amount*

specifies the charge amount if calculated externally

-zs *charge_start_time*

start time for the charge in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. This is only needed for incremental charges when the start of the charge interval differs from the original start time and is used to determine the appropriate allocation to charge. Defaults to now - charge duration if unable to derive by other means.

-zt *charge_duration*

incremental duration for the charge in seconds. This is only needed for incremental charges when the incremental duration differs from the total actual duration and is used to compute the incremental charge amount.

-Z "{\"*metric_name*\":*metric_amount*,...}"

generic metrics. Metrics represent floating point metrics of the job or average metric values across the nodes in the job.

--hours

display time-based credits in hours. For cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--itemize

returns the composite charge information in the response data. This must be used in conjunction with the --verbose flag to display the data.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-account

Synopsis

mam-create-account {[-a] *account_name*} [-A | -I] [-o *organization_name*] [-d *description*] [-X, --extension *property*=*value*]... [-u [^|!][+|-]*user_name*,...]... [--create-fund True|False] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-account is used to create a new account. Users may be associated with the account. If auto-generation is turned o
ject or the --create-fund flag is asserted, a fund will automatically be created for the account.

Options

-a *account_name*

name of the new account

-A

makes the account active

--create-fund True|False

This option is used to override the fund auto-generation setting. Seting this option to True will create a default fund for this account. Setting this option to False will inhibit the creation of a default fund for this account.

-d *description*

account description

-I

makes the account inactive

-o *organization_name*

name of the organization to which the account belongs

-u [^|!][+|-]*user_name*[,[^|!][+|-]*user_name*...]

defines user members of the account. The optional caret or exclamation symbol indicates whether the user should be created as an admin (^) or not (!) for the account. The optional plus or minus sign can preceed each member to indicate whether the member should be created in the active (+) or inactive (-) state. By default, a user will be created in the active state but not an admin. Multiple users may be passed to the -u option in a comma-delimited list or multiple -u options may be specified.

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-chargerate

Synopsis

mam-create-chargerate {[-n] *charge_rate_name*} [-x *charge_rate_value*] {-z *charge_rate_amount*} [-d *description*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-chargerate is used to create a new charge rate.

Options

-d *description*

charge rate description

[-n] *charge_rate_name*

The name of the usage record property that is being charged for, e.g. Processors, QualityOfService, ...

-x *charge_rate_value*

charge rate value. For name-valued charge rates this is the usage property value corresponding to the rate. For numeric-valued charge rates this is the range of values corresponding to the rate. A blank value will function as a default charge rate. See the Charge Rates chapter in the Administrator Guide for more details.

-z *charge_rate_amount*

rate for the charge. This is an integer or decimal number and may include operators that indicate how the charge is applies as well as divisors and time-based units. See the Charge Rates chapter in the Administrator Guide for more details.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-event

Synopsis

mam-create-event --fire-command *fire_command* [-s *fire_time*] [-e *end_time*] [--rearm-period *rearm_period*] [--rearm-on-failure True|(False)] [--failure-commmand *failure_command*] [--notify *notification_url*] [--catch-up (True)|False] [-d *description*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-event is used to create a new event.

Options

--catch-up *boolean*

If the CatchUp boolean is set to True and the server was down during the time this event should have fired, the event scheduler will attempt to make up for the past due events by progressively firing them (rearming based on previous arm time) until catching up to the present. The actions will still show as having occurred in the present rather than in the past. If set to False, and the server is brought back up after an outage, the event scheduler will still fire immediately for a past due event, but it will only fire once and then rearm relative to the current time. The default is True.

-d *description*

event description

-e *end_time*

time this event becomes inactive in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--failure-command *mam-shell_command*

the command to be executed if the fired command results in an unsuccessful response status. This command is expressed in a serialized form of the request identical to the syntax used in the interactive control program (mam-shell). The option argument will need to be appropriately quoted and/or escaped in order to avoid misinterpretation or alteration by the shell.

--fire-command *mam_shell_command*

command to be executed. This command is expressed in a serialized form of the request identical to the syntax used in the interactive control program (mam-shell). The option argument will need to be appropriately quoted and/or escaped in order to avoid misinterpretation or alteration by the shell.

--notify [+-=][*delivery_method*:][*recipient*]

A Notification method logs the result of the fired command. If the term is a -, the notification is sent only on failure. If the term is a +, the notification is sent only on success. Otherwise the notification is always sent. See the chapter on Managing notifications in the Moab Accounting Manager Administrator Guide for more information about delivery method and recipient. The default is to log all event statuses to the Notification table.

--rearm-on-failure *boolean*

If the RearmOnFailure boolean is set to False, the event will not be rearmed if the command was unsuccessful. If set to True, the event will be evaluated for rearming even if the command response has a status of Failure. The standard default is False.

--rearm-period *period*[[@*instant*][~|^]|!]

The RearmPeriod is a time period expression specifying when the event will be rearmed. This period expression is of the form: "*period*[[@*instant*][~|^]|!]". The period is expressed as an integer number followed by a designator of minute(s), hour(s), day(s), week(s), month(s) or year(s). For example, the period might be 1 day, 2 hours, or 5 minutes. The optional Instant locks the period to a specific instant within the time period such as 1 day @ hour 12 or 1 month @ day 3. The modifiers indicate whether the time period should be relative to now (!), or relative to the start of this (~) designator (month or minute, etc.), or relative to the start of the first (^) designator (month or minute, etc.). For example, assuming the FireTime was 7:15, if you specified 4 hours ! as the rearm period it would be rearmed at 11:15, if you specified 4 hours ~ as the rearm period it would be rearmed at 11:00, and if you specified 4 hours ^ as the rearm period it would be rearmed at 8:00.

-s *fire_time*

the target time for the event to be triggered by the event scheduler. The actual fire time may be dependent on the state of the server and will be recorded in the CreationTime property of the corresponding "Event Fire" Transaction. An event may also be fired manually with the mam-shell Event Fire action.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-fund

Synopsis

mam-create-fund [-n *fund_name*] [--priority *fund_priority*] [--default-deposit *deposit_amount*] [-d *description*] [-X, --extension *property*=*value*]... [-u *user_name*,...]... [-g *group_name*,...]... [-a *account_name*,...]... [-o *organization_name*,...]... [-c *class_name*,...]... [-m *machine_name*,...]... [--constraint *constraint_name*=[!]*constraint_value*,...]... [--parent *parent_fund_id*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-fund is used to create new funds. A new id is automatically generated for the fund. It essentially creates a new container into which time-bounded credits valid toward a specific set of constraints can be later credited and debited.

Options

-a *account_name*[,*account_name*...]

account required by the fund. Multiple accounts may be passed to the -a option in a comma-delimited list or multiple -a options may be specified.

-c *class_name*[,*class_name*...]

class or queue required by the fund. Multiple classes may be passed to the -c option in a comma-delimited list or multiple -c options may be specified.

--constraint *constraint_name*=*constraint_value*[,*constraint_name*=*constraint_value*...]

specifies a constraint for the fund. The constraint value may be a perl5 regular expression. An exclamation point may be prepended to the constraint value to express a negation of the constraint. Multiple constraint options may be specified. For example, --constraint User=amy --constraint Machine=colony will make the credits in this fund valid only for the user amy on the machine colony. Multiple constraints may be passed to the --constraint option in a comma-delimited list or multiple --constraint options may be specified.

-d *description*

fund description

--default-deposit *deposit_amount*

used as the default amount for any deposit that is made to this fund that does not specify a deposit amount. A zero value will result in the creation of an allocation with a zero balance (or add nothing if an allocation already exists and a reset is not being requested). A negative value can be used to stipulate that the allocations in the fund should be ended if the fund is reset. An empty value (”) or NULL can be used to stipulate that no change will be made to the allocations if the fund is reset.

-g *group_name*[,*group_name*...]

group required by the fund. Multiple groups may be passed to the -g option in a comma-delimited list or multiple -g options may be specified.

-m *machine_name*[,*machine_name*...]

machine (i.e. cluster) required by the fund. Multiple machines may be passed to the -m option in a comma-delimited list or multiple -m options may be specified.

-n *fund_name*

fund name

-o *organization_name*[,*organization_name*...]

organization required by the fund. Multiple organizations may be passed to the -o option in a comma-delimited list or multiple -o options may be specified.

--parent *parent_fund_id*

associates the newly created fund as a child of the specified parent fund

--priority *fund_priority*

fund priority

-u *user_name*[,*user_name*...]

user required by the fund. Multiple users may be passed to the -u option in a comma-delimited list or multiple -u options may be specified.

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-lien

Synopsis

mam-create-lien [-J *instance_name*] [-s *start_time*] {-e *end_time* | -t *lien_duration*} [-d *description*] [-X, --extension *property*=*value*]... {-A *allocation_id*<-*fund_id*=*sublien_amount*,...}... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-lien is used to create a lien against specified allocations. A lien object and its allocation associations will be created. Unlike **mam-reserve**, no calculated lien amount will be returned nor will a usage record be created with the lien. Warning -- this command bypasses the normal mechanisms that prevent more liens from being placed against an allocation than it can support.

Options

-A *allocation_id*<-*fund_id*=*sublien_amount*[,*allocation_id*<-*fund_id*=*sublien_amount*...]

creates subliens against the specified allocations. At least one allocation expression must be specified with the lien. Multiple allocation expressions may be passed to the -A option in a comma-delimited list. Alternatively, multiple -A options may be specified.

-d *description*

description of the lien

-e *end_time*

expiration time for the lien in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The end time defaults to now + duration if not specified.

[-J] *instance_name*

instance name (e.g. job id) for the lien

-s *start_time*

beginning time for the lien in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The start time defaults to now if not specified.

-t *lien_duration*

amount of time in seconds it is anticipated the item will be used. The duration defaults to end time minus start time if not specified.

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-organization

Synopsis

mam-create-organization {[-o] *organization_name*} [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-organization is used to create a new organization.

Options

-d *description*

organization description

-o *organization_name*

name of the new organization

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-quote

Synopsis

mam-create-quote [[--pin] [-J *instance_name*] | --nopin] [-s *start_time*] {-e *end_time* | -t *quote_duration*} [-d *description*] [-X, --extension *property*=*value*]... {--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...}... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-quote is used to create a chargeable quote template. A quote object and its associated charge rates will be created. The override charge rates specified in the command will be used by instances referencing the quote. Unlike **mam-quote**, no calculated quote amount will be returned nor will a usage record be created with the quote.

Options

-d *description*

description of the quote

-e *end_time*

expiration time for the quote template in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The rates associated with this quote may not be claimed after this time. If end time is not specified but duration is specified, the end time will be calculated as start time + duration. If both are specified but are inconsistent, the duration will be ignored.

-J *instance_name*

instance name (e.g. job id) for the quote. An instance name may not be specified if the quote is unpinned.

--nopin

indicates that the quote is not to be pinned to a specific instance. An unpinned quote may be used by any instance while the quote is active.

--pin

indicates that the quote is to be pinned to a specific instance. This is the default. If the instance is not specified when the quote is created, the first instance to claim it will become the pinned instance. Once a quote is pinned to a particular instance, no other instances may use the quote.

--rate
charge_rate_name[{*charge_rate_value*}]=*charge_rate_amount*[,*charge_rate_name*[{*charge_rate_value*}]=

associates the specified charge rates with the quote. At least one charge rate expression must be specified with the quote. Multiple charge rate expressions may be passed to the --rate option in a comma-delimited list. Alternatively, multiple --rate options may be specified.

-s *start_time*

beginning time for the quote template in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The rates associated with this quote may not be claimed before this time. The start time defaults to now if not specified.

-t *quote_duration*

amount of time in seconds the rates in the quote template may be used. The duration is used to calculate an end time (start time + duration) as an alternative to specifying the end time.

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-role

Synopsis

mam-create-role {[-r] *role_name*} [-d *description*] [-u *user_name*,...]... [-A *object_name*->*action_name*[{*instance_name*}],...]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-role is used to create a new role. Users and actions may be associated with the role at creation time.

Options

-A *object_name->action_name*[{*instance_name*}][,*object_name->action_name*[{*instance_name*}]...]

adds actions to the role. The object, action and instance must be specified in the form shown. Unless specified, the instance will default to a value of ANY. Multiple actions may be passed to the -A option in a comma-delimited list. Alternatively, multiple -A options may be specified

-d *description*

role description

[-r] *role_name*

name of the new role

-u *user_name*[,*user_name*...]

adds users to the role. Multiple users may be passed to the -u option in a comma-delimited list. Alternatively, multiple -u options may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-usagerecord

Synopsis

mam-create-usagerecord {-J *instance_name*} [-n *designated_name*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [-N *nodes*] [-P *processors*] [-C *cpu_time*] [-M *memory*] [-D *disk*] [-E *energy*] [-F "{\"*feature_name*\":*feature_count*,...}"] [-R "{\"*resource_name*\":*resource_count*,...}"] [-L "{\"*license_name*\":*license_count*,...}"] [-Z "{\"*metric_name*\":*metric_amount*,...}"] [-V "{\"*variable_name*\":\"*variable_value*\",...}"] [-W *requested_duration*] [-t *actual_duration*] [-s *start_time*] [-e *end_time*] [-x *exit_code*] [--stage *lifecycle_stage*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-usagerecord is used to create a new usage record

Options

-a *account_name*

account name

-c *class_name*

class or queue used

-C *cpu_time*

CPU time used

-d *description*

description of the usage

-D *disk*

amount of disk used

-e *end_time*

date and time the usage ended in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-E *energy*

amount of energy used

-F "{\"*feature_name*\":*feature_count*,...}"

allocated node features. Features represent counts of the node features allocated to the job.

-g *group_name*

group name

-J *instance_name*

instance name (e.g. job id) for the new usage record

-L "{\"*license_name*\":*license_count*,...}"

licenses used. Licenses represent software licenses that are used in integer units.

-m *machine_name*

name of the cluster

-M *memory*

amount of memory used

-n *designated_name*

user-specified job name

-N *nodes*

number of nodes used

-o *organization_name*

organization name

-P *processors*

number of processors used

-Q *quality_of_service*

quality of service used

-R "{\"*resource_name*\":*resource_count*,...}"

consumable resources allocated. Resources represent consumable resources that may be allocated in integer amounts.

-s *start_time*

date and time the usage started in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-t *actual_duration*

total actual duration in seconds

-T *usage_record_type*

specifies the usage record type (Job, Reservation, Service, VPC, etc.)

-u *user_name*

user name

-V "{\"*variable_name*\":\"*variable_value*\",...}"

job variables. Variables represent arbitrary string variables passed into the job.

-W *requested_duration*

total estimated wallclock duration in seconds

-x *exit_code*

exit code

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

-Z "{\"*metric_name*\":*metric_amount*,...}"

generic metrics. Metrics represent floating point metrics of the job or average metric values across the nodes in the job.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-create-user

Synopsis

mam-create-user {[-u] *user_name*} [-A | -I] [-n *common_name*] [--phone *phone_number*] [--email *email_address*] [-a *default_account*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-create-user is used to create a new user.

Options

-a *default_account*

specifies the account which will be charged when no account is specified

-A

makes the user active

-d *description*

user description

--email *email_address*

email address

-I

makes the user inactive

-n *common_name*

common name for the user

--phone *phone_number*

phone number

[-u] *user_name*

userid (name) for the new user

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-account

Synopsis

mam-delete-account {[-a] *account_name*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-account deletes an account.

Options

[-a] *account_name*

name of the account to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-allocation

Synopsis

mam-delete-allocation {-I | {[-i] *allocation_id*}} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-allocation deletes an allocation or purges stale allocations.

Options

[-i] *allocation_id*

deletes the specified allocation

-I

purges (deletes) stale (inactive) allocations

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-chargerate

Synopsis

mam-delete-chargerate {[-n] *charge_rate_name*} [-x *charge_rate_value*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-chargerate deletes a charge rate.

Options

[-n] *charge_rate_name*

name of the charge rate to delete.

-x *charge_rate_value*

charge rate value to delete. If a value is not specified, only a charge rate with an empty value will be deleted.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-event

Synopsis

mam-delete-event {[-E] *event_id*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-event deletes an event.

Options

[-E] *event_id*

id of event to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-fund

Synopsis

mam-delete-fund {[-f] *fund_id*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-fund deletes a fund.

Options

[-f] *fund_id*

deletes the specified fund

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-lien

Synopsis

mam-delete-lien {-I | {-J *instance_name*} | {[-l] *lien_id*}} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-lien deletes a lien or purges stale liens.

Options

-I

purges (deletes) stale (expired) liens

-J *instance_name*

deletes liens with the specified instance name (or job id)

[-l] *lien_id*

deletes specified lien

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-notification

Synopsis

mam-delete-notification {[-N] *notification_id*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-notification deletes a stored notification.

Options

[-N] *notification_id*

id of notification to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-organization

Synopsis

mam-delete-organization {[-o] *organization_name*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-organization deletes an organization.

Options

[-o] *organization_name*

name of organization to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-quote

Synopsis

mam-delete-quote {-I | {[-q] *quote_id*}} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-quote deletes a quote or purges expired quotes.

Options

[-q] *quote_id*

deletes the specified quote

-l

purges (deletes) all expired quotes

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-role

Synopsis

mam-delete-role {[-r] *role_name*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-role deletes a role.

Options

[-r] *role_name*

name of the role to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-usagerecord

Synopsis

mam-delete-usagerecord {[-j] *usage_record_id* | -J *instance_name*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-usagerecord deletes a usage record.

Options

[-j] *usage_record_id*

specifies the usage record id to delete. Instance names can be non-unique (i.e. resource managers often recycle job ids). This option allows you to specify a unique usage record using the unique identifier.

-J *instance_name*

specifies the instance name (e.g. job id) to delete. Since instance names are assigned by externally and may be non-unique (such as job ids assigned by a resource manager), all usage records with the specified instance name will be deleted.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-delete-user

Synopsis

mam-delete-user {[-u] *user_name*} [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-delete-user deletes a user.

Options

[-u] *user_name*

name of user to delete

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-deposit

Synopsis

mam-deposit [-f *fund_id*] [-i *allocation_id*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] [[-z] *deposit_amount*] [-L *credit_limit*] [-s *start_time*] [-e *end_time*] [--reset] [-d *description*] [--create-fund True|False] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-deposit is used to make time-bounded deposits into funds. After applying all filter options, if there is exactly one debitable fund for the specified criteria, a deposit will be made into that fund. If multiple funds match the specified criteria, a list of matching funds will be displayed and you will be prompted to respecify the deposit against one of the enumerated funds. If no funds match your criteria, if auto-generation is turned on for the fund object or the --create-fund flag is asserted, a fund will be created and a deposit made into it, otherwise the deposit will fail (the fund will need to be created with **mam-create-fund**). The --reset option can be used to end the current allocation and create a new allocation with the deposit. If an amount is not specified for the deposit, the fund's default deposit amount will be used. A zero default deposit amount will result in the creation of an allocation with a zero balance (or add nothing if an allocation already exists and a reset is not being requested). A negative default deposit amount can be used to stipulate that the allocations in the fund should be ended if the fund is reset. An empty default deposit amount stipulates that no change will be made to the allocations if the fund is reset.

Options

-a *account_name*

The fund for the deposit should be restricted to one usable by the specified account

-c *class_name*

The fund for the deposit should be restricted to one usable by the specified class

--create-fund True|False

This option is used to override the fund auto-generation setting. Seting this option to True will create a default fund for this deposit. Setting this option to False will inhibit the creation of a default fund for this deposit.

-d *description*

reason for the deposit. The annotation applies to the transaction description (seen via mam-list-transactions), not the allocation description.

-e *end_time*

end time for the allocation to be credited in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The end time will default to Infinity.

-f *fund_id*

id of the fund into which the deposit will be made

--filter *filter_name*=*filter_value*

restricts the fund to one whose constraints do not conflict with the specified filters. For example mam-modify-fund --filter User=amy will restrict the fund to one usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

The fund for the deposit should be restricted to one usable by the specified group

-i *allocation_id*

specifies that the deposit should go into the specified allocation. This option is incompatible with the -L option.

-L *credit_limit*

if a credit limit is specified, a new allocation will be created with the specified credit limit. This option is incompatible with the -i option.

-m *machine_name*

The fund for the deposit should be restricted to one usable by the specified machine

-o *organization_name*

The fund for the deposit should be restricted to one usable by the specified organization

--reset

Ends the current allocation and creates a new allocation with the deposit.

-s *start_time*

start time for the allocation to be credited in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. The start time will default to -Infinity.

-u *user_name*

The fund for the deposit should be restricted to one usable by the specified user

[-z] *deposit_amount*

amount to deposit

--hours

treat currency as specified in hours. In systems where the currency is measured in resource-seconds (like processor-seconds), this option allows the amount and credit limit to be specified in resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-list-accounts

Synopsis

mam-list-accounts [[-a] *account_pattern*] [-A | -I] [-o *organization_name*] [-X, --extension *property*=*value*]... [-u *user_name*] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-accounts is used to display account information.

The fields which are displayed by default by this command can be customized by setting the account.show configuration parameter in mam-client.conf.

Options

[-a] *account_pattern*

displays only accounts matching the pattern. If no pattern is specified then all accounts are displayed. The following wildcards are supported:

*

matches any number of characters

?

matches a single character

-A

displays only active accounts

-I

displays only inactive accounts

-o *organization_name*

displays only accounts belonging to the specified organization.

-u *user_name*

displays only accounts having the specified user as a member.

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Name, Active, Users, Organization, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Active

boolean indicating whether this account is active or not

CreationTime

time this account was created

Deleted

boolean indicating whether this account is deleted or not

Description

account description

ModificationTime

time this account was last modified

Name

account name

Organization

organization to which the account belongs

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

Users

list of users belonging to the account. A caret prefixing a user name indicates that the user is an account admin. A minus sign prefixing a user name indicates that the user is an inactive member of the account.

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-allocations

Synopsis

mam-list-allocations [[-i] *allocation_id*] [-f *fund_id*] [-A | -I | {[-s *start_time*] [-e *end_time*]}] [-X, --extension *property*=*value*]... [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] [--include-ancestors] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-allocations is used to display allocation information.

The fields which are displayed by default by this command can be customized by setting the allocation.show configuration parameter in mam-client.conf.

Options

-a *account_name*

displays only allocations usable by the specified account

-A

displays only active allocations

-c *class_name*

displays only allocations usable by the specified class

-e *end_time*

displays only allocations that start before the specified end time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-f *fund_id*

displays only the allocations associated with the specified fund

--filter *filter_name*=*filter_value*

displays allocations whose fund constraints comply with the specified filters. For example mam-list-funds --filter User=amy will display funds usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

displays only allocations usable by the specified group

[-i] *allocation_id*

displays only the allocation with the given id

--include-ancestors

includes ancestors of the selected allocations

-I

displays only inactive allocations

-m *machine_name*

displays only allocations usable by the specified machine

-o *organization_name*

displays only allocations usable by the specified organization

-s *start_time*

displays only allocations that end after the specified start time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-u *user_name*

displays only allocations usable by the specified user

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Fund, FundName, Active, StartTime, EndTime, InitialDeposit, Adjustments, Allocated, CreditLimit,

Capacity, Used, PercentUsed, Remaining, PercentRemaining, Balance, Reserved, Effective, Available, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator are specified, fields without an explicit operator are assumed to have the GroupBy operator.

Active

boolean indicating whether this allocation is active or not

Adjustments

total of subsequent adjustments to the initial deposit via deposits, withdrawals or transfers (Allocated - InitialDeposit)

Allocated

adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.

Available

amount currently available for charging. If the allocation is active, this is Remaining - Reserved + CreditLimit. If the allocation is inactive, this is zero.

Balance

active allocation balance. If the allocation is active, this is the remaining allocation amount (Remaining). If the allocation is inactive, this is zero.

Capacity

total expendable amount (Allocated + CreditLimit)

CreationTime

time this allocation was created

CreditLimit

credit limit. Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)

Deleted

boolean indicating whether this allocation is deleted or not

Description

allocation description

Effective

effective balance not blocked by liens. If the allocation is active, this is Remaining - Reserved. If the allocation is inactive, this is zero.

EndTime

time this allocation becomes inactive

Fund

fund id

FundName

fund name

Id

allocation id

InitialDeposit

amount of the first deposit into this allocation

ModificationTime

time this allocation was last modified

PercentRemaining

percentage of allocation remaining (Remaining * 100 / Capacity)

PercentUsed

percentage of allocation used (Used * 100 / Capacity)

Remaining

remaining allocation amount

RequestId

id of the last modifying request

Reserved

sum of active lien amounts against this allocation

StartTime

time this allocation becomes active

TransactionId

id of the last modifying transaction

Used

amount used from this allocation (Allocated - Remaining)

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-chargerates

Synopsis

mam-list-chargerates [[-n] *charge_rate_name*] [-x *charge_rate_value*] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-chargerates is used to display charge rate information.

Options

[-n] *charge_rate_name*

displays only charge rates of the specified name

-x *charge_rate_value*

displays only charge rates having the specified value

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Name, Value, Amount, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Amount

charge rate amount. The amount is an integer or decimal number and may include operators indicating how to apply the charge rate as well as divisors and time-based units. See the Charge Rates chapter in the Administrator Guide for more details.

CreationTime

time this charge rate was created

Deleted

boolean indicating whether this charge rate is deleted or not

Description

charge rate description

ModificationTime

time this charge rate was last modified

Name

charge rate name (e.g. Processors, License, etc.)

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

Value

charge rate value. For name-valued charge rates this is the usage property value corresponding to the rate. For numeric-valued charge rates this is the range of values corresponding to the rate. A blank value will function as a default charge rate. See the Charge Rates chapter in the Administrator Guide for more details.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-events

Synopsis

mam-list-events [[-E] *event_id*] [-s *start_time*] [-e *end_time*] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-events is used to display event information.

The fields which are displayed by default by this command can be customized by setting the event.show configuration parameter in mam-client.conf.

Options

-e *end_time*

displays events with a prospective fire time occuring before the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

[-E] *event_id*

displays only the event with the specified id

-s *start_time*

displays events with a prospective fire time occuring after the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, FireCommand, FireTime, ArmTime, RearmPeriod, EndTime, Notify, RearmOnFailure, FailureCommand, CatchUp, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

ArmTime

time the event was last armed or fired. This field is used as a reference time to be able to derive how long the event has been waiting to happen. This field will be initially set to mark the moment the first FireTime is set and updated thereafter to indicate the last time the event was fired. In the case where an event does not have a FireTime set, this field may be set manually and used in a similar manner. If we consider the time between event firings as "laps", this could be thought of as the Lap Start Time.

CatchUp

if set to True and MAM was down during the time this event should have fired, MAM will attempt to make up for the past due events by progressively firing them (rearming based on previous arm time) until catching up to the present. The actions will still show as having occurred in the present rather than in the past. If set to False, and MAM is brought back up after an outage, MAM will still fire immediately for a past due event, but it will only fire once and then rearm relative to the current time.

CreationTime

time this event was created

Deleted

boolean indicating whether this event is deleted or not

Description

event description

EndTime

time after which an event having a rearm period will be deleted

FailureCommand

serialized mam-shell request string to be executed if the fired command results in an unsuccessful response status. They syntax is the same as used to invoke commands within the mam-shell prompt.

FireCommand

serialized mam-shell request string to be executed when the event is fired. They syntax is the same as used to invoke commands within the mam-shell prompt.

FireTime

target time for the event to be triggered. The actual fire time may be dependent on the state of the server and will be recorded in the CreationTime property of the corresponding "Event Fire" Transaction.

Id

event id

ModificationTime

time this event was last modified

Notify

Expression specifying where to send a notification of the response for the fire command and the failure command. The notification expression is of the form: [+-=][*delivery_method*:][*recipient*][,[+-=][*delivery_method*:][*recipient*]]* (e.g. -store:amy). If the term is a -, the notification is sent only on failure. If the term is a +, the notification is sent only on success. Otherwise the notification is always sent. There can be multiple notify expressions separated by a comma.

RearmPeriod

time period expression specifying when the event will be rearmed. This period expression is of the form: "*period*[[@*instant*][~|^]|!]" where period may be something like 1 day, 2 hours, or 5 minutes. Instant locks the period to a specific instant within the time period such as 1 day @ hour 12 or 1 month @ day 3. The modifiers indicate whether the time period should be

relative to now (!), or relative to the start of this (~) designator (month or minute, etc.), or relative to the start of the first (^) designator (month or minute, etc.).

RearmOnFailure

if set to False, the event will not be rearmed if the command was unsuccessful. if set to True, the event will be evaluated for rearming even if the command response has a status of Failure. The standard default value for this boolean is False.

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-funds

Synopsis

mam-list-funds [[-f] *fund_id*] [-A | -I] [-n *fund_name*] [-X, --extension *property*=*value*]... [-u

user_name] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-funds is used to display fund information.

The fields which are displayed by default by this command can be customized by setting the fund.show configuration parameter in mam-client.conf.

Options

-a *account_name*

displays only funds valid toward the specified account

-A

displays funds with active allocations

-c *class_name*

displays only funds usable by the specified class

[-f] *fund_id*

displays only funds with the specified id

--filter *filter_name*=*filter_value*

displays funds whose constraints do not conflict with the specified filters. For example mam-list-funds -f User=amy will display funds usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

displays funds usable by the specified group

-I

displays funds with inactive allocations

-m *machine_name*

displays only funds valid toward the specified machine

-n *fund_name*

displays only funds with the specified name

-o *organization_name*

displays only funds valid toward the specified organization

-u *user_name*

displays only funds valid toward the specified user

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Name, Balance, Priority, DefaultDeposit, CreditLimit, InitialDeposit, Allocated, Constraints, Allocations, Parent, Children, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Allocated

adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.

Allocations

lists the active allocations in this fund in the format *id*:*amount*:*start_time*:*end_time*

Balance

sum of active allocation amounts within this fund

Children

lists the children funds in the format id[(*deposit_share*)][^] where the carat symbol (^) is displayed if Overflow is True

Constraints

contraints on fund usage

CreationTime

time this fund was created

CreditLimit

sum of active credit limits within this fund

DefaultDeposit

used as the default amount for any deposit that is made to this fund that does not specify a deposit amount. A zero value will result in the creation of an allocation with a zero balance (or add nothing if an allocation already exists and a reset is not being requested). A negative value can be used to stipulate that the allocations in the fund should be ended if the fund is reset. An empty value is used to stipulate that no change will be made to the allocations if the fund is reset.

Deleted

boolean indicating whether this fund is deleted or not

Description

fund description

Id

fund id

InitialDeposit

initial deposit for current allocation

ModificationTime

time this fund was last modified

Name

fund name

Parent

displays the parent fund in the format id[(*deposit_share*)][^] where the carat symbol (^) is displayed if Overflow is True

Priority

fund priority

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-itemizedcharges

Synopsis

mam-list-itemizedcharges [-j *usage_record_id*] [-J *instance_name*] [-n *usage_property_name*] [-s *start_time*] [-e *end_time*] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-itemizedcharges is used to display charge details.

Options

-e *end_time*

displays charges occuring before the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-j *usage_record_id*

displays only charges associated with the specified usage record

-J *instance_name*

displays only charges against the specified instance (such as a job id)

-n *usage_record_property_name*

displays only charges against the specified usage property

-s *start_time*

displays charges occuring after the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: UsageRecord, Instance, Name, Value, Duration, Rate, ScalingFactor, Amount, Details, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Amount

amount charged

CreationTime

time this charge was created

Deleted

boolean indicating whether this charge is deleted or not

Description

charge description

Details

details of the formula used in calculating the charge

Duration

amount of time the item was used in seconds

Instance

instance name (such as a job id) for the charge

ModificationTime

time this charge was last modified

Name

usage record property name (also charge rate name)

Rate

base charge rate

RequestId

id of the last modifying request

ScalingFactor

product of all applicable multipliers (discounts and premiums) applied to the base rate

TransactionId

id of the last modifying transaction

UsageRecord

usage record id

Value

usage record property value

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-liens

Synopsis

mam-list-liens [[-l] *lien_id*] [-A| -I] [-J *instance_pattern*] [-X, --extension *property*=*value*]... [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type AttributedTo|ImpingesUpon] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-liens is used to display lien information.

The fields which are displayed by default by this command can be customized by setting the lien.show configuration parameter in mam-client.conf.

Options

-a *account_name*

display only liens against the given account

-A

displays only unexpired liens

-c *class_name*

displays only liens against the specified class

--filter *filter_name*=*filter_value*

displays liens whose constraints do not conflict with the specified filters. For example mam-list-liens --filter User=amy will display liens for the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type AttributedTo|ImpingesUpon

selects the filtering type. If the AttributedTo filter type is used, the query will return all liens associated with usage records satisfying the filters. If the ImpingesUpon filter type is used, the query will return all liens affecting the balances of funds satisfying the filters. The default filter type is AttributedTo.

-g *group_name*

display only liens against the given group

-I

displays only expired liens

-J *instance_pattern*

displays only liens with instance names (or job ids) matching the pattern.

[-l] *lien_id*

displays only the specified lien

-m *machine_name*

display only liens against the given machine

-o *organization_name*

display only liens against the given organization

-u *user_name*

display only liens against the given user

--site *site_name*

obtain response from specified site

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Amount, Instance, UsageRecord, StartTime, EndTime, Duration, Funds, Allocations, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId. Additionally, unambiguous usage record properties may also be specified for display (User, Group, Account, Organization, Class, QualityOfService, Machine, Nodes, Processors, Memory, etc).

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Funds

list of funds that the lien has holds against

Allocations

list of allocations that the lien has holds against in the format allocation_id<-fund_id=reserved_amount

Amount

reserved amount

CreationTime

time this lien was created

Deleted

boolean indicating whether this lien is deleted or not

Description

lien description

Duration

expected duration of the reserved usage in seconds

EndTime

time the lien becomes inactive

Id

lien id

Instance

the lien is against the specified instance (i.e. job id)

ModificationTime

time this lien was last modified

RequestId

id of the last modifying request

StartTime

time the lien becomes active

TransactionId

id of the last modifying transaction

UsageRecord

the id of the usage record associated with the lien and containing the usage properties

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-notifications

Synopsis

mam-list-notifications [[-N] *notification_id*] [-E *event_id*] [-T *notification_type*] [-k *primary_key_value*] [-u *recipient*] [-x *status*] [-s *start_time*] [-e *end_time*] [--delete] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-notifications is used to display stored notification information.

The fields which are displayed by default by this command can be customized by setting the notification.show configuration parameter in mam-client.conf.

Options

--delete

delete a notification after it has been queried

-e *end_time*

displays notifications sent before the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-E *event_id*

displays only the notifications associated with the specified event id

-k *primary_key_value*

displays only the notifications associated with the specified primary key value. This value is the value of the primary key of the object instance that the command acted on.

[-N] *notification_id*

displays only the notifications with the specified id

-s *start_time*

displays notifications sent after the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-x *status*

displays notifications having the specified status (e.g. Success, Failure, etc.)

-T *notification_type*

displays notifications of the specified type (e.g. Fire, Failure)

-u *recipient*

displays notifications having the specified recipient. This could be a user name or any tag that identifies the intended reader of this notification.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Event, Type, Status, Code, Message, Key, Recipient, EndTime, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Code

event command exit code

CreationTime

time this notification was created

Deleted

boolean indicating whether this notification is deleted or not

EndTime

time after which a notification will be deleted

Event

event id

Key

object primary key value

Id

notification id

Message

event command message

ModificationTime

time this notification was last modified

Recipient

recipient to notify

Status

event command status

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

Type

displays the type of notification. Notifications may be created by event "Fire" commands or by event "Failure" commands.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-organizations

Synopsis

mam-list-organizations [[-o] *organization_pattern*] [-X, --extension *property*=*value*]... [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-organizations is used to display organization information.

The fields which are displayed by default by this command can be customized by setting the organization.show configuration parameter in mam-client.conf.

Options

[-o] *organization_pattern*

displays only organizations matching the pattern. If no pattern is specified then all organizations are displayed. The following wildcards are supported:

*

matches any number of characters

?

matches a single character

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Name, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

CreationTime

time this organization was created

Deleted

boolean indicating whether this organization is deleted or not

Description

organization description

ModificationTime

time this organization was last modified

Name

organization name

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-quotes

Synopsis

mam-list-quotes [[-q] *quote_id*] [-J *instance_name*] [-A | -I] [-X, --extension *property*=*value*]... [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-quotes is used to display quote information.

The fields which are displayed by default by this command can be customized by setting the quote.show configuration parameter in mam-client.conf.

Options

-a *account_name*

display only quotes for the given account

-A

displays only unexpired quotes

-c *class_name*

displays only quotes for the specified class

--filter *filter_name*=*filter_value*

displays quotes whose constraints do not conflict with the specified filters. For example mam-list-quotes --filter User=amy will display quotes for the user amy. Multiple filter options may be specified which are logically anded together.

-g *group_name*

displays quotes for the specified group

-I

displays only expired quotes

-J *instance_name*

displays only quotes having the specified instance name (or job id)

-m *machine_name*

display only quotes for the given machine

-o *organization_name*

display only quotes for the given organization

[-q] *quote_id*

display only info for the specified quote

-u *user_name*

display only quotes for the given user

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Amount, Pinned, Instance, UsageRecord, StartTime, EndTime, Duration, ChargeRates, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId. Additionally, unambiguous usage record properties may also be specified for display (User, Group, Account, Organization, Class, QualityOfService, Machine, Nodes, Processors, Memory, etc).

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Amount

quoted amount

ChargeRates

saved charge rates to be used when the quote is referenced. These are displayed in the format charge_rate_name[{charge_rate_value}]=charge_rate_amount

CreationTime

time this quote was created

Deleted

boolean indicating whether this quote is deleted or not

Description

quote description

Duration

expected duration of the quoted usage in seconds

EndTime

time the quote becomes inactive

Id

quote id

Instance

the quote may only be used by the specified instance

ModificationTime

time this quote was last modified

Pinned

boolean indicating whether the quote is pinned or not

RequestId

id of the last modifying request

StartTime

time the quote becomes active

TransactionId

id of the last modifying transaction

UsageRecord

the id of the usage record instantiated by the quote and containing the usage properties

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-roles

Synopsis

mam-list-roles [[-r] *role_name*] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-roles is used to display role information.

Options

[-r] *role_name*

displays information for only the specified role

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Name, Users, Actions, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Actions

list of actions permitted by the role. Actions are displayed in the format object->action{instance}

CreationTime

time this role was created

Deleted

boolean indicating whether this role is deleted or not

Description

role description

ModificationTime

time this role was last modified

Name

role name

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

Users

list of users granted access to the role

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-transactions

Synopsis

mam-list-transactions [[-T] *transaction_id*] [-R *request_id*] [-O *object*] [-A *action*] [-k *primary_key_value*] [-U *actor*] [-f *fund_id*] [-i *allocation_id*] [-u *user_name*] [-a *account_name*] [-m *machine_name*] [-j *usage_record_id*] [-J *instance_name*] [-s *start_time*] [-e *end_time*] [-X, --extension *property*=*value*]... [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-transactions is used to display transaction information.

The fields which are displayed by default by this command can be customized by setting the transaction.show configuration parameter in mam-client.conf.

Options

-a *account_name*

displays only transactions involving the given account

-A *action*

displays only transactions invoking the specified action

-e *end_time*

displays transactions occuring before the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-f *fund_id*

displays only transactions involving the specified fund

-i *allocation_id*

displays only transactions logged against the specific allocation

-j *usage_record_id*

displays only transactions affecting the given usage record

-J *instance_name*

displays only transactions affiliated with the given instance name (e.g. job id)

-k *primary_key_value*

displays only transactions involving the objects having the specified primary key value (i.e. having the specified Id or Name) or associations with the given parent name

-m *machine_name*

displays only transactions involving the given machine

[-O] *object*

displays only transactions performing actions on the given object type

-R *request_id*

displays only transactions with the specified request id. A unique request id is associated with each request, while each request may be associated with more than one transaction.

-s *start_time*

displays transactions occuring on or after the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

[-T] *transaction_id*

displays only transactions with the specified transaction id. A transaction occurs when an action is invoked on an object. A complex request may involve multiple transactions.

-u *user_name*

displays only transactions involving the given user

-U *actor*

displays only transactions invoked by the specified user

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Object, Action, Actor, Key, Child, Instance, Count, Amount, Delta, Balance, Remaining, User, Account, Machine, Fund, Allocation, UsageRecord, Duration, Description, Details, CreationTime, ModificationTime, Deleted, RequestId, TransactionId. Additionally, when the transaction record refers to a Usage Record, unambiguous usage record properties may also be specified for display (e.g. Group, Organization, Class, QualityOfService, Nodes, Processors, Memory), as well as the derived fields: NodeHours, NodeSeconds, ProcHours and ProcSeconds.

Aggregate values may be requested for specified attributes by using operators. Aliases may be used to specify the column name for the aggregated field. Aggregated fields are specified in the form: *operator*(*attribute_name*)[=*alias*]. Valid operators include: Sum, Average, Count, Unique, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Account

account name associated with the transaction

Action

action name

Actor

user that performed the action

Allocation

allocation id associated with the transaction

Amount

amount

Balance

effective active balance. If the allocation is active, this is the same as the remaining allocation amount (Remaining). If the allocation is inactive, this is zero.

Child

if the transaction object is an association, this is the value of the child

Count

number of objects affected by the transaction

CreationTime

time this transaction was created

Deleted

boolean indicating whether this transaction is deleted or not

Delta

change (positive or negative) to the effective active balance of an allocation (Balance). This may differ in some cases from the change in the actual allocation amount (Remaining). For example, if an allocation expires, a negative delta will be recorded for the event, while the remaining allocation amount has not changed. On the other hand, a modification of the amount in an expired allocation will be recorded as a delta of zero.

Description

transaction description

Details

Additional assignments, conditions, options and other details of the transaction are recorded here when there is no applicable transaction property to store them in

Duration

duration associated with the transaction in seconds

Fund

fund id associated with the transaction

Id

transaction id

Instance

instance name

Key

if the transaction object is an association, this is the value of the parent, otherwise this is the value of the primary key (id or name) of the object.

Machine

machine name associated with the transaction

ModificationTime

time this transaction was last modified

Object

object name

Remaining

remaining allocation amount. If an allocation amount has the potential for being affected by this transaction, this field stores the remaining allocation amount after the transaction completed. Note that for expired allocations, this will still record the allocation's actual

remaining amount, even though the allocation's effective active balance (Balance) may be zero. Thus it is possible for the Remaining amount to change even though the Delta is zero or the Remaining amount to remain unchanged even though the Delta is non-zero.

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

UsageRecord

usage record id associated with the transaction

User

user name associated with the transaction

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-usagerecords

Synopsis

mam-list-usagerecords [[-j] *usage_record_id*] [-J *instance_name_pattern*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [--stage *lifecycle_stage*] [-X, --extension *property*=*value*]... [-s *start_time*] [-e *end_time*] [--full] [--show *attribute_name*,...] [--format csv|raw|standard] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-usagerecords is used to display usage record information.

The fields which are displayed by default by this command can be customized by setting the usagerecord.show configuration parameter in mam-client.conf.

Options

-a *account_name*

display only usage records affiliated with the given account

-c *class_name*

class or queue name

-e *end_time*

ended before the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-g *group_name*

display only usage records affiliated with the given group

[-j] *usage_record_id*

displays only the usage record with the specified id

-J *instance_name_pattern*

displays only usage records matching the specified instance name (e.g. job id) pattern. The following wildcards are supported:

*

matches any number of characters

?

matches a single character

-m *machine_name*

display only usage records affiliated with the given machine

-Q *quality_of_service*

display only usage records affiliated with the given quality of service

-s *start_time*

ended on or after the specified time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-T *usage_record_type*

display only usage records of the specified type (e.g. Job, Reservation, VPC, etc.)

-u *user_name*

display only usage records affiliated with the given user

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Id, Type, Instance, Charge, Stage, Quote, User, Group, Account, Organization, Class, QualityOfService, Machine, Nodes, Processors, CPUTime, Memory, Disk, Duration, RequestedDuration, QueueDuration, ProcHours, ProcSeconds, NodeHours, NodeSeconds, SubmitTime, StartTime, EndTime, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Aggregate values may be requested for specified attributes by using operators. Aggregated fields are specified in the form: *operator*(*attribute_name*). Valid operators include: Sum, Average, Count, Min, Max and GroupBy. When an operator is specified, fields without an explicit operator are assumed to have the GroupBy operator.

Partial values may be requested for complex (multi-valued) attributes. Partial values are specified in the form: *attribute_name*{*part_name*}.

Aliases may be used to specify the resultant column name. Aliases are specified in the form: *attribute_name*=*alias*.

Aggregate values, partial values and aliases may be combined, e.g. *operator*(*attribute_name*{*part_name*})=*alias*.

Account

account name associated with the usage

BlockedProcessors

number of processors blocked by the job

Charge

cumulative amount charged

Class

class or queue name associated with the usage

CPUTime

CPU time used

CreationTime

time this usage record was created

Deleted

boolean indicating whether this usage record is deleted or not

Description

usage description

Disk

amount of disk used

Duration

duration of the usage

EndTime

overall end time of the usage

Features | Features{*feature_part_name*}

allocated node features. Individual feature counts may be displayed using the partial value syntax.

Group

group name associated with the usage

Id

usage record id

Instance

instance name (i.e. job id)

Licenses | Licenses{*license_part_name*}

licenses used. Individual license counts may be displayed using the partial value syntax.

Machine

cluster name

Metrics | Metrics{*metric_part_name*}

generic metrics. Individual metric values may be displayed using the partial value syntax.

Memory

amount of memory used

ModificationTime

time this usage record was last modified

NodeHours

Nodes * Duration / 3600

NodeSeconds

Nodes * Duration

Nodes

number of nodes used

Organization

organization name associated with the usage

Processors

number of cores or processors allocated

ProcessorEquivalents

number of processor equivalents allocated by the job

ProcHours

Processors * Duration / 3600

ProcSeconds

Processors * Duration

QualityOfService

quality of service associated with the usage

QueueDuration

duration the job was in the idle state

Quote

associated quote id

RequestedDuration

requested wallclock limit

RequestId

id of the last modifying request

Resources | Resources{*resource_part_name*}

generic resources. Individual resource amounts may be displayed using the partial value syntax.

Stage

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

StartTime

latest start time of the usage

SubmitTime

creation or submit time of the item

TransactionId

id of the last modifying transaction

Type

usage record type

User

user name associated with the usage

Variables | Variables{*variable_part_name*}

job variables. Individual variable values may be displayed using the partial value syntax.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-list-users

Synopsis

mam-list-users [[-u] *user_pattern*] [-A | -I] [-X, --extension *property*=*value*]... [-a *account_name*] [--full] [--show *attribute_name*,...] [--long] [--wide] [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--version] [--about]

Description

mam-list-users is used to display user information.

The fields which are displayed by default by this command can be customized by setting the user.show configuration parameter in mam-client.conf.

Options

-a *account_name*

displays only users affiliated with the specified account

-A

displays only active users

-I

displays only inactive users

[-u] *user_pattern*

displays only users matching the pattern. If no pattern is specified then all users are displayed. The following wildcards are supported:

*

matches any number of characters

?

matches a single character

-X, --extension *property*=*value*

extension property. Any number of extra custom conditions may be specified.

--site *site_name*

obtain response from specified site

--full

displays all attributes

--show *attribute_name*[,*attribute_name*...]

displays only the specified attributes in the given order. Valid attributes include: Name, Active, CommonName, PhoneNumber, EmailAddress, DefaultAccount, Accounts, Description, CreationTime, ModificationTime, Deleted, RequestId, TransactionId

Active

boolean indicating whether this user is active or not

CommanName

common name for the user

CreationTime

time this user was created

Deleted

boolean indicating whether this user is deleted or not

DefaultAccount

default account

Description

user description

EmailAddress

email address

ModificationTime

time this user was last modified

Name

user name

PhoneNumber

phone number

Accounts

list of accounts to which the user belongs

RequestId

id of the last modifying request

TransactionId

id of the last modifying transaction

--long

long format. Display multi-valued fields in a multi-line format.

--wide

wide format. Display multi-valued fields in a single-line comma-separated format.

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress headers and success messages

--debug

log debug info to screen

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-modify-account

Synopsis

mam-modify-account {[-a] *account_name*} [-A | -I] [-o *organization_name*] [-d *description*] [-X, --extension *property*=*value*]... [--add-user(s) [^|!][+|-]*user_name*,...]... [--del-user(s) *user_name*,...]... [--mod-user(s) [^|!][+|-]*user_name*,...]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-account modifies an account.

Options

[-a] *account_name*

name of the account to modify

-A

activate the account

-I

deactivate the account

-d *description*

new description

--add-user(s) [^|!][+|-]*user_name*[,[^|!][+|-]*user_name*...]

adds user members of the account. The optional caret or exclamation symbol indicates whether the user should be created as an admin (^) or not (!) for the account. The optional plus or minus signs can preceed each member to indicate whether the member should be created in the active (+) or inactive (-) state. By default, a user will be created in the active state but not an admin. Multiple users may be passed to the addUser option in a comma-delimited list. Alternatively, multiple addUser options may be specified.

--del-user(s) *user_name*[,*user_name*...]

removes user members of the account. Multiple users may be passed to the delUser option in a comma-delimited list. Alternatively, multiple delUser options may be specified.

--mod-user(s) [^|!][+|-]*user_name*[,[^|!][+|-]*user_name*...]

modifies user members of the account. The caret symbol or exclamation symbol indicates the user should be changed to become an admin (^) or not (!) for the account. The plus or minus signs indicate whether the user should be changed to become active (+) or inactive (-). If an active or admin modifier is not specified, that aspect of the user member will remain unchanged. Multiple users may be passed to the modUser option in a comma-delimited list. Alternatively, multiple modUser options may be specified.

-o *organization_name*

new name of the organization to which the account belongs

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-allocation

Synopsis

mam-modify-allocation {[-i] *allocation_id*} [-s *start_time*] [-e *end_time*] [-L *credit_limit*] [-d *description*] [-X, --extension *property*=*value*]... [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-allocation modifies an allocation. This includes changing the credit limit or description, or adjusting the start time or end time.

Options

-d *description*

new description

-e *end_time*

new end time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

[-i] *allocation_id*

id of the allocation to modify

-L *credit_limit*

new credit limit

-s *start_time*

new start time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--hours

treat currency as specified in hours. In systems where the currency is measured in resource-seconds (like processor-seconds), this option allows the credit limit to be specified in resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-chargerate

Synopsis

mam-modify-chargerate {[-n] *charge_rate_name*} [-x *charge_rate_value*] [-z *charge_rate_amount*] [-d *description*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-chargerate modifies a charge rate. Only the amount or the description of a charge rate may be modified.

Options

-d *description*

new charge rate description

[-n] *charge_rate_name*

name of the charge rate to change.

-x *charge_rate_value*

charge rate value expression to change. If value is not specified, an empty value is assumed.

-z *charge_rate_amount*

new amount for the charge rate. The amount is an integer or decimal and may include operators indicating how to apply the charge rate as well as divisors and time-based units. See the Charge Rates chapter in the Administrator Guide for more details.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-event

Synopsis

mam-modify-event {[-E] *event_id*} [--fire-command *fire_command*] [-s *fire_time*] [-e *end_time*] [--rearm-period *rearm_period*] [--rearm-on-failure True|(False)] [--failure-commmand *failure_command*] [--notify *notification_url*] [--catch-up (True)|False] [-d *description*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-event modifies an event.

Options

--catch-up *boolean*

If the CatchUp boolean is set to True and the server was down during the time this event should have fired, the event scheduler will attempt to make up for the past due events by progressively firing them (rearming based on previous arm time) until catching up to the present. The actions will still show as having occurred in the present rather than in the past. If set to False, and the server is brought back up after an outage, the event scheduler will still fire immediately for a past due event, but it will only fire once and then rearm relative to the current time. The default is True.

-d *description*

new description

-e *end_time*

time this event becomes inactive in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-E *event_id*

id of event to modify

--failure-command *mam_shell_command*

new command to be executed if the fired command results in an unsuccessful response status. This command is expressed in a serialized form of the request identical to the syntax used in the interactive control program (mam-shell). The option argument will need to be appropriately quoted and/or escaped in order to avoid misinterpretation or alteration by the shell.

--fire-command *mam_shell_command*

command to be executed. This command is expressed in a serialized form of the request identical to the syntax used in the interactive control program (mam-shell). The option argument will need to be appropriately quoted and/or escaped in order to avoid misinterpretation or alteration by the shell.

--notify [+-=][*delivery_method*:][*recipient*]

A Notification method logs the result of the fired command. If the term is a -, the notification is sent only on failure. If the term is a +, the notification is sent only on success. Otherwise the notification is always sent. See the chapter on Managing notifications in the Moab Accounting Manager Administrator Guide for more information about delivery method and recipient. The default is to log all event statuses to the Notification table.

--rearm-on-failure *boolean*

If the RearmOnFailure boolean is set to False, the event will not be rearmed if the command was unsuccessful. If set to True, the event will be evaluated for rearming even if the command response has a status of Failure. The standard default is False.

--rearm-period *period*[[@*instant*][~|^]|!]

The RearmPeriod is a time period expression specifying when the event will be rearmed. This period expression is of the form: "*period*[[@*instant*][~|^]|!]". The period is expressed as an integer number followed by a designator of minute(s), hour(s), day(s), week(s), month(s) or year(s). For example, the period might be 1 day, 2 hours, or 5 minutes. The optional Instant locks the period to a specific instant within the time period such as 1 day @ hour 12 or 1 month @ day 3. The modifiers indicate whether the time period should be relative to now (!), or relative to the start of this (~) designator (month or minute, etc.), or relative to the start of the first (^) designator (month or minute, etc.). For example, assuming the FireTime was 7:15, if you specified 4 hours ! as the rearm period it would be rearmed at 11:15, if you specified 4 hours ~ as the rearm period it would be rearmed at 11:00, and if you specified 4 hours ^ as the rearm period it would be rearmed at 8:00.

-s *fire_time*

new target time for the event to be triggered by the event scheduler. The actual fire time may be dependent on the state of the server and will be recorded in the CreationTime property of the corresponding "Event Fire" Transaction. An event may also be fired manually with the mam-shell Event Fire action.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-fund

Synopsis

mam-modify-fund [[-f] *fund_id*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] {{[-n *fund_name*] [--priority *fund_priority*] [--default-deposit *deposit_amount*] [-d *description*] [-X, --extension *property*=*value*]... [--add-constraint *constraint_name*=[!]*constraint_value*,...]... [--del-constraint(s)

constraint_name[=*constraint_value*],...]... [--parent *parent_fund_id*]} | {--reset [--all]}}
[--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-fund modifies a fund. This includes adding to or deleting from constraints for the account. After applying all filter options, if there is exactly one applicable fund, that fund will be modified. Otherwise, a list of funds will be displayed for the specified filters and you will be prompted to rerun mam-modify-fund against one of the enumerated funds.

Options

-a *account_name*

the fund to modify should be restricted to one usable by the specified account

--add-constraint
constraint_name=[!]*constraint_value*[,*constraint_name*=[!]*constraint_value*...]

adds a constraint to the fund. The constraint value may be a perl5 regular expression. An exclamation point may be prepended to the constraint value to express a negation of the constraint. Multiple constraints may be passed to the --add-constraint option in a comma-delimited list or multiple --add-constraint options may be specified.

--all

used with the --reset option to specify that you want to reset all active allocations for all funds

-c *class_name*

the fund to modify should be restricted to one usable by the specified class

-d *description*

new description

--default-deposit *deposit_amount*

used as the default amount for any deposit that is made to this fund that does not specify a deposit amount. A zero value will result in the creation of an allocation with a zero balance (or add nothing if an allocation already exists and a reset is not being requested). A negative value can be used to stipulate that the allocations in the fund should be ended if the fund is reset. An empty value (”) or NULL can be used to stipulate that no change will be made to the allocations if the fund is reset.

--del-constraint(s)
constraint_name=*constraint_value*[,*constraint_name*[=*constraint_value*]...]

removes a constraint from the fund. Multiple constraints may be passed to the --del-constraint option in a comma-delimited list or multiple --del-constraint options may be specified.

[-f] *fund_id*

id of the fund to modify

--filter *filter_name*=*filter_value*

restricts the fund to one whose constraints do not conflict with the specified filters. For example mam-modify-fund --filter User=amy will restrict the fund to one usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

the fund to modify should be restricted to one usable by the specified group

-m *machine_name*

the fund to modify should be restricted to one usable by the specified machine

-n *fund_name*

new fund name

-o *organization_name*

the fund to modify should be restricted to one usable by the specified organization

--parent *parent_fund_id*

sets a new parent fund (replacing the current one if it exists)

--priority *fund_priority*

new fund priority

--reset

ends all active allocations and initiates a new default deposit. If the default deposit amount is positive, a new allocation is created with this amount, otherwise no deposit is made and the fund becomes inactive. You may reset the allocations for a specified fund using the [-f] *fund_id* option, all funds using the --all option, or use filtering options to filter the funds to be reset. This option may not be used with any other modifying option.

-u *user_name*

the fund to modify should be restricted to one usable by the specified user

-X, --extension *property*=*value*

extension property to modify. Multiple extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-lien

Synopsis

mam-modify-lien {[-l] *lien_id*} [-s *start_time*] [-e *end_time*] [-t *lien_duration*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-lien modifies a lien.

Options

-d *description*

new description

-e *end_time*

new expiration time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

[-l] *lien_id*

id of the lien to modify

-s *start_time*

new start time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-t *lien_duration*

duration of the lien in seconds. Although the lien start time and end time are enforced, the duration is not authoritative. If the timeframe between the end time and the start time is greater than the duration, the difference is the allotted grace period (which defaults to 10 minutes).

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-organization

Synopsis

mam-modify-organization {[-o] *organization_name*} [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-organization modifies an organization.

Options

-d *description*

new description

-o *organization_name*

name of organization to modify

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-quote

Synopsis

mam-modify-quote {[-q] *quote_id*} [-s *start_time*] [-e *end_time*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-quote modifies a quote.

Options

-d *description*

new description

-e *end_time*

new expiration time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

[-q] *quote_id*

id of the quote to modify

-s *start_time*

new start time in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-role

Synopsis

mam-modify-role {[-r] *role_name*} [-d *description*] [--add-user(s) *user_name*,...]... [--add-action(s) *object_name*->*action_name*[{*instance_name*}],...]... [--del-user(s) *user_name*,...]... [--del-action(s) *object_name*->*action_name*[{*instance_name*}],...]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-role modifies a role. This can include adding or deleting users from a role and adding or deleting actions from a role.

Options

-d *description*

new description

--add-action(s) *object_name*->*action_name*[{*instance_name*}][,*object_name*->*action_name*[{*instance_name*}]...]

adds actions to the role. The object, action and instance must be specified in the form shown. Unless specified, the instance will default to a value of ANY. Multiple actions may be passed to the addAction option in a comma-delimited list. Alternatively, multiple addAction options may be specified.

--add-user(s) *user_name*[,*user_name*...]

adds users to the role. Multiple users may be passed to the addUser option in a comma-delimited list. Alternatively, multiple addUser options may be specified.

--del-action(s) *object_name*->*action_name*[{*instance_name*}][,*object_name*->*action_name*[{*instance_name*}]...]

removes actions from a role. The object and action must be specified while the instance is optional. Multiple actions may be passed to the delAction option in a comma-delimited list. Alternatively, multiple delAction options may be specified.

--del-user(s) *user_name*[,*user_name*...]

removes users from a role. Multiple users may be passed to the delUser option in a comma-delimited list. Alternatively, multiple delUser options may be specified.

[-r] *role_name*

name of the role to modify

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-modify-usagerecord

Synopsis

mam-modify-usagerecord {[-j] *usage_record_id* | -J *instance_name*} [-n *designated_name*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [-N *nodes*] [-P *processors*] [-C *cpu_time*] [-M *memory*] [-D *disk*] [-E *energy*] [-F "{\"*feature_name*\":*feature_count*,...}"] [-R "{\"*resource_name*\":*resource_count*,...}"] [-L "{\"*license_name*\":*license_count*,...}"] [-Z "{\"*metric_name*\":*metric_amount*,...}"] [-V "{\"*variable_name*\":\"*variable_value*\",...}"] [-W *requested_duration*] [-t *actual_duration*] [-s *start_time*] [-e *end_time*] [-x *exit_code*] [--stage *lifecycle_stage*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-usagerecord modifies a usage record.

Options

-a *account_name*

new account name

-c *class_name*

new class or queue

-C *cpu_time*

new CPU time used

-d *description*

new description

-D *disk*

new disk used

-e *end_time*

new date and time the usage ended in the format YYYY-MM-DD[hh:mm:ss]

-E *energy*

new energy used

-F "{\"*feature_name*\":*feature_count*,...}"

allocated node features. Features represent counts of the node features allocated to the job.

-g *group_name*

new group name

[-j] *usage_record_id*

id of the usage record to modify. Instance names can be non-unique (i.e. resource managers often recycle job ids). This option allows you to specify a usage record uniquely using the unique usage record identifier.

-J *instance_name*

instance name (e.g. job id) of the usage record(s) to modify. If there is exactly one matching usage record, that usage record will be modified. Otherwise, a list of usage records will be displayed for

the specified instance and you will be prompted to rerun mam-modify-usagerecord against one of the enumerated usage records.

-L "{\"*license_name*\":*license_count*,...}"

licenses used. Licenses represent software licenses that are used in integer units.

-m *machine_name*

new name of the cluster

-M *memory*

new memory used

-n *designated_name*

user-specified job name

-N *nodes*

new number of nodes used

-o *organization_name*

new organization name

-P *processors*

new number of processors used

-Q *quality_of_service*

new quality of service used

-R "{\"*resource_name*\":*resource_count*,...}"

consumable resources allocated. Resources represent consumable resources that may be allocated in integer amounts.

-s *start_time*

new date and time the usage started in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-t *actual_duration*

total actual duration in seconds

-T *usage_record_type*

specifies the usage record type (Job, Reservation, etc.)

-u *user_name*

new user name

-V "{\"*variable_name*\":\"*variable_value*\",...}"

job variables. Variables represent arbitrary string variables passed into the job.

-W *requested_duration*

total estimated wallclock duration in seconds

-x *exit_code*

new exit code

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

-Z "{\"*metric_name*\":*metric_amount*,...}"

generic metrics. Metrics represent floating point metrics of the job or average metric values across the nodes in the job.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified usage records

--version

display product version

--about

display product information

mam-modify-user

Synopsis

mam-modify-user {[-u] *user_name*} [-A | -I] [-n *common_name*] [--phone *phone_number*] [--email *email_address*] [-a *default_account*] [-d *description*] [-X, --extension *property*=*value*]... [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-modify-user modifies a user.

Options

-a *default_account*

specifies the account which will be charged when no account is specified

-A

activate the user

-d *description*

new description

--email *email_address*

email address

-I

deactivate the user

-n *common_name*

common name for the user

--phone *phone_number*

phone number

[-u] *user_name*

name of user being modified

-X, --extension *property*=*value*

extension property. Any number of extra field assignments may be specified.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-quote

Synopsis

mam-quote [-J *instance_name*] [[-j] *usage_record_id*] [-q *quote_template_id*] [-n *designated_name*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [-N *nodes*] [-P *processors*] [-C *cpu_time*] [-M *memory*] [-D *disk*] [-E *energy*] [-F "{\"*feature_name*\":*feature_count*,...}"] [-R "{\"*resource_name*\":*resource_count*,...}"] [-L "{\"*license_name*\":*license_count*,...}"] [-Z "{\"*metric_name*\":*metric_amount*,...}"] [-V "{\"*variable_name*\":\"*variable_value*\",...}"] [-W *requested_duration*] [--stage *lifecycle_stage*] [-d *description*] [-X, --extension *property*=*value*]... [-zt *quote_duration* [-G *grace_duration*]] [-zs *quote_start_time*] [-z *quote_amount*] [--cost-only | --guarantee] [--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...]... [--hours] [--itemize] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-quote is used to obtain a quote for usage. This command and its options can be used to estimate the cost of using resources, to validate that a requestor has sufficient access and funds to use the requested resources, and to guarantee that the charge rates used to generate the quote do not change when applying subsequent liens and charges.

Options

-a *account_name*

account name

-c *class_name*

class or queue used

-C *cpu_time*

estimated CPU time used

--cost-only

returns the cost, ignoring all balance and validity checks. This option is mutually exclusive with --guarantee.

-d *description*

description of the usage

-D *disk*

amount of disk used

-E *energy*

amount of energy used

-F "{\"*feature_name*\":*feature_count*,...}"

allocated node features. Features represent counts of the node features allocated to the job.

-g *group_name*

group name

-G *grace_duration*

grace period in seconds. If the quote duration is specified but not the quote end time, the quote end time will be calculated as the quote start time plus the quote duration plus the grace duration.

--guarantee

guarantees the quote and returns a quote id to secure the current charge rates. This results in the creation of a quote record and a permanent usage record. This option is mutually exclusive with --cost-only.

[-j] *usage_record_id*

usage record id for the quote (if already created with mam-create-usagerecord or a previous mam-quote)

-J *instance_name*

instance name (e.g. job id) for the quote (if known).

-L "{\"*license_name*\":*license_count*,...}"

licenses used. Licenses represent software licenses that are used in integer units.

-m *machine_name*

name of the cluster

-M *memory*

amount of memory used

-n *designated_name*

user-specified job name

-N *nodes*

number of nodes used

-o *organization_name*

organization name

-P *processors*

number of processors used

-q *quote_template_id*

quote template used to override standard charge rates

-Q *quality_of_service*

quality of service used

-R "{\"*resource_name*\":*resource_count*,...}"

consumable resources allocated. Resources represent consumable resources that may be allocated in integer amounts.

--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...

uses the specified charge rates in the quote. The specified rates override the general rates. If the --guarantee option is specified, these charge rates will be saved and used when this quote is referenced in a charge action. Multiple charge rates may be passed to the --rate option in a comma-delimited list. Alternatively, multiple --rate options may be specified.

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-T *usage_record_type*

usage record type (e.g. Job, Reservation, VPC, Service, etc.)

-u *user_name*

user name

-V "{\"*variable_name*\":\"*variable_value*\",...}"

job variables. Variables represent arbitrary string variables passed into the job.

-W *requested_duration*

total estimated wallclock duration in seconds

-X, --extension *property*=*value*

extension property. Any number of extra usage record properties may be specified with the quote. When expressing accumulating properties, *value* may be an expression of the form: [*cumulative_value*][(*incremental_value*)]. If both *incremental_value* and *cumulative_value* are specified, *incremental_value* will be used for the quote while the *cumulative_value* value will be recorded as the cumulative value used in the usage record. If only *incremental_value* is specified, this value will be used for the quote only and no cumulative value will be recorded in the usage record. If only *cumulative_value* is specified, this value will be used both in the quote and recorded in the usage record.

-z *quote_amount*

specifies the quote amount if calculated externally

-zs *quote_start_time*

start time for the quote in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. This is only needed for non-cost-only quotes and is used to determine the appropriate allocation to apply the quote to. Defaults to now if unable to derive by other means.

-zt *quote_duration*

incremental duration for the quote in seconds. This is only needed for incremental quotes when the incremental duration differs from the estimated wallclock duration and is used to compute the incremental quote amount.

-Z "{\"*metric_name*\":*metric_amount*,...}"

generic metrics. Metrics represent floating point metrics of the job or average metric values across the nodes in the job.

--hours

display time-based credits in hours. For cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--itemize

returns the composite charge information in the response data. This must be used in conjunction with the --verbose flag to display the data.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-refund

Synopsis

mam-refund {-J *instance_name* | [-j] *usage_record_id*} [-z *refund_amount*] [-i

allocation_id] [-d *description*] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-refund issues a refund for the specified usage. The command will return a list of usage records if the usage search does not yield a unique match. If an amount is not specified, the appropriate allocations will be credited for the full amount the overall usage was charged. A lesser amount may be specified for a partial refund. The refund will go to the allocations that were charged unless an allocation is specified, in which case the specified allocation will be credited.

Options

-d *description*

explanatory message for the refund

-i *allocation_id*

allocation to be credited. If this is omitted, the allocations that were debited in the original charges will be credited.

[-j] *usage_record_id*

the unique usage record identifier assigned by the accounting manager to distinguish between usage with non-unique instance names.

-J *instance_name*

name of the instance (e.g. job id). This id might not be unique among the historical list of usage records managed by the accounting manager.

-z *refund_amount*

amount to refund. This amount must be non-negative and less than or equal to the amount charged for the overall usage.

--hours

treat currency as specified in hours. In systems where the currency is measured in resource-seconds (like processor-seconds), this option allows the amount to be specified in resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-reserve

Synopsis

mam-reserve {-J *instance_name*} [[-j] *usage_record_id*] [-q *quote_id*] [-n *designated_name*] [-T *usage_record_type*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization*] [-c *class_name*] [-Q *quality_of_service*] [-m *machine_name*] [-N *nodes*] [-P *processors*] [-C *cpu_time*] [-M *memory*] [-D *disk*] [-E *energy*] [-F "{\"*feature_name*\":*feature_count*,...}"] [-R "{\"*resource_name*\":*resource_count*,...}"] [-L "{\"*license_name*\":*license_count*,...}"] [-Z "{\"*metric_name*\":*metric_amount*,...}"] [-V "{\"*variable_name*\":\"*variable_value*\",...}"] [-W *requested_duration*] [-s *start_time*] [--stage *lifecycle_stage*] [-d *description*] [-X, --extension *property*=*value*]... [-zt *lien_duration*] [-zs *lien_start_time* [-G *grace_duration*]] [-z *lien_amount*] [--modify | --replace] [--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...]... [--hours] [--itemize] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-reserve is used to obtain a lien for usage.

Options

-a *account_name*

account name

-c *class_name*

class or queue used

-C *cpu_time*

estimated CPU time used

-d *description*

description of the usage

-D *disk*

amount of disk used

-E *energy*

amount of energy used

-F "{\"*feature_name*\":*feature_count*,...}"

allocated node features. Features represent counts of the node features allocated to the job.

-g *group_name*

group name

-G *grace_duration*

grace period in seconds. If the lien duration is specified but not the lien end time, the lien end time will be calculated as the lien start time plus the lien duration plus the grace duration.

[-j] *usage_record_id*

usage record id for the lien (if already created with mam-create-usagerecord, mam-quote or a previous mam-reserve). This is used to place a hold against an existing usage record if the instance name (e.g. job id) is ambiguous or if usage has already been debited and you want to reserve an additional amount associated with the same usage record.

-J *instance_name*

instance name (e.g. job id) for the lien (if known). This can sometimes be non-unique (such as when a resource manager recycles job ids), and does not always unambiguously identify a usage record to reserve. In such cases, look up and specify the usage record id for the lien.

-L "{\"*license_name*\":*license_count*,...}"

licenses used. Licenses represent software licenses that are used in integer units.

-m *machine_name*

name of the cluster

-M *memory*

amount of memory used

--modify

Causes the reserve operation to augment existing liens instead of creating new ones. This new option is mutually exclusive with the --replace option which deletes existing matching liens and recreates a new one. The default action is to create a new lien even if a lien for an instance of the same name exists. The modify behavior supports extending liens out dynamically and is often used with incremental charging.

-n *designated_name*

user-specified job name

-N *nodes*

number of nodes used

-o *organization_name*

organization name

-P *processors*

number of processors used

-Q *quality_of_service*

quality of service used

-q *quote_id*

quote used to determine charge rates

-R "{\"*resource_name*\":*resource_count*,...}"

consumable resources allocated. Resources represent consumable resources that may be allocated in integer amounts.

--rate *charge_rate_name*[{*charge_rate_value*}]=*charge_rate_amount*,...

uses the specified charge rates in the lien. The specified rates override the general rates or rates guaranteed through a quote. Multiple charge rates may be passed to the --rate option in a comma-delimited list. Alternatively, multiple --rate options may be specified.

--replace

If this option is specifed, similarly named liens will be deleted before this lien is created. The default action is to create a new lien while leaving any existing liens for instances of the same name. The replace option should be specified if you want this lien to replace existing liens for instances of the same name such as when a system reuses instance names. This new option is mutually exclusive with the --modify option which modifies any existing matching lien instead of creating a new one.

-s *start_time*

start time for the usage in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now

--stage *lifecycle_stage*

latest stage in the object's accounting lifecycle (e.g. Create, Start, Continue, End)

-T *usage_record_type*

usage record type (e.g. Job, Reservation, etc.)

-u *user_name*

user name

-V "{\"*variable_name*\":\"*variable_value*\",...}"

job variables. Variables represent arbitrary string variables passed into the job.

-W *requested_duration*

total estimated wallclock duration in seconds

-X, --extension *property*=*value*

extension property. Any number of extra usage record properties may be specified with the lien. When expressing accumulating properties, *value* may be an expression of the form: [*cumulative_value*][(*incremental_value*)]. If both *incremental_value* and *cumulative_value* are specified, *incremental_value* will be used for the lien while the *cumulative_value* value will be recorded as the cumulative value used in the usage record. If only *incremental_value* is specified, this value will be used for the lien only and no cumulative value will be recorded in the usage record. If only *cumulative_value* is specified, this value will be used both in the lien and recorded in the usage record.

-z *lien_amount*

specifies the lien amount if calculated externally

-zs *lien_start_time*

start time for the lien in the format YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. This is only needed for incremental liens when the start of the lien interval differs from the original start time

and is used to determine the appropriate allocation to reserve. Defaults to now if unable to derive by other means.

-zt *lien_duration*

incremental duration for the lien in seconds. This is only needed for incremental liens when the incremental duration differs from the estimated wallclock duration and is used to compute the incremental lien amount.

-Z "{\"*metric_name*\":*metric_amount*,...}"

generic metrics. Metrics represent floating point metrics of the job or average metric values across the nodes in the job.

--hours

display time-based credits in hours. For cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--itemize

returns the composite charge information in the response data. This must be used in conjunction with the --verbose flag to display the data.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-set-password

Synopsis

mam-set-password [[-u] *user_name*] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-set-password sets a user password. If the user name is not specified via an option or as the unique argument, then the invoking user will be taken as the user whose password will be set. The invoker will be prompted for the new password.

Options

[-u] *user_name*

name of user whose password is to be set. If no user is specified, the invoking user will be taken as the user whose password will be set.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-statement

Synopsis

mam-statement [[-f] *fund_id*] [-n *fund_name*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] [-s *start_time*] [-e *end_time*] [--summarize] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--version] [--about]

Description

mam-statement is used to display a fund statement. For a specified time frame it displays the beginning and ending balances as well as the total credits and debits to the fund over that period. This is followed by an itemized report of the debits and credits. Filters can be used to select the funds you would like to review.

Options

-a *account_name*

The statement will represent a combination of information for all of the funds available to the specified account. Note that the statement may include information from other accounts if the included funds are shared by multiple accounts.

-c *class_name*

The statement will represent a combination of information for all of the funds available to the specified class. Note that the statement may include information from other classes if the included funds are shared by multiple classes.

-e *end_time*

the end of the reporting period in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. A default of Now is taken if this option is omitted.

[-f] *fund_id*

A fund statement will be displayed for the specified fund.

--filter *filter_name*=*filter_value*

reports on funds whose constraints do not conflict with the specified filters. For example mam-statement --filter User=amy will display funds usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-g *group_name*

The statement will represent a combination of information for all of the funds available to the specified group. Note that the statement may include information from other groups if the included funds are shared by multiple groups.

-m *machine_name*

The statement will represent a combination of information for all of the funds available to the specified machine. Note that the statement may include information from other machines if the included funds are shared by multiple machines.

[-n] *fund_name*

A fund statement will be displayed for funds with the given name.

-o *organization_name*

The statement will represent a combination of information for all of the funds available to the specified organization. Note that the statement may include information from other organizations if the included funds are shared by multiple organizations.

-s *start_time*

the beginning of the reporting period in the format: YYYY-MM-DD[hh:mm:ss]|-Infinity|Infinity|Now. A default of -Infinity is taken if this option is omitted.

--summarize

display transaction summaries only. Deposits, Refunds, Charges etc. will be shown as total as opposed to being itemized.

-u *user_name*

The statement will represent a combination of information for all of the funds available to the specified user. Note that the statement may include information from other users if the included funds are shared by multiple users.

--hours

display time-based credits in hours. In cases where the currency is measured in resource-seconds (like processor-seconds), the currency is divided by 3600 to display resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--version

display product version

--about

display product information

mam-transfer

Synopsis

mam-transfer {--from-fund *source_fund_id* &| --from-allocation *source_allocation_id* &| --from-filter *filter_name*=*filter_value*...} {--to-fund *destination_fund* &| --to-allocation *destination_allocation_id* &| --to-filter *filter_name*=*filter_value*...} [--filter-type ExactMatch|Exclusive|NonExclusive] {[-z] *transfer_amount*} [-d *description*] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-transfer issues a transfer between funds.

Options

-d *description*

reason for the transfer. The annotation applies to the transaction description (seen via mam-list-transactions), not the allocation description.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

--from-allocation *source_allocation_id*

credits will be transferred from the specified allocation id only. If the allocation is omitted, only credits from active allocations will be transferred in the order of earliest expiring first.

--from-fund *source_fund_id*

fund to be debited

--from-filter *filter_name*=*filter_value*

if one or more source filters are specified and there is exactly one matching fund, a transfer will be made from that fund. Otherwise, a list of funds will be displayed for the specified filters and you will be prompted to respecify the transfer against one of the enumerated funds. Multiple fromFilter options may be specified which are logically anded together.

--to-allocation *destination_allocation_id*

credits will be transferred to the specified allocation id only. If the allocation is omitted, credits will be transferred to the allocation having the same start and endtime as the source allocation the funds are taken from, or if such an allocation is non-existent, a new allocation will be created in the target fund having the same start and end time.

--to-fund *destination_fund_id*

fund to be credited

--to-filter *filter_name*=*filter_value*

if one or more destination filters are specified and there is exactly one matching fund, a transfer will be made to that fund. Otherwise, a list of funds will be displayed for the specified filters and you will be prompted to respecify the transfer against one of the enumerated funds. Multiple toFilter options may be specified which are logically anded together.

[-z] *transfer_amount*

amount to transfer

--hours

treat currency as specified in hours. In systems where the currency is measured in resource-seconds (like processor-seconds), this option allows the amount to be specified in resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-withdraw

Synopsis

mam-withdraw [-f *fund_id*] [-i *allocation_id*] [-u *user_name*] [-g *group_name*] [-a *account_name*] [-o *organization_name*] [-c *class_name*] [-m *machine_name*] [--filter *filter_name*=*filter_value*]... [--filter-type ExactMatch|Exclusive|NonExclusive] {[-z] *withdrawal_amount*} [-d *description*] [--hours] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about]

Description

mam-withdraw makes a withdrawal from the specified fund.

Options

-a *account_name*

The fund for the withdrawal should be restricted to one usable by the specified account

-c *class_name*

The fund for the withdrawal should be restricted to one usable by the specified class

-d *description*

reason for the withdrawal. The annotation applies to the transaction description (seen via mam-list-transactions), not the allocation description.

-g *group_name*

The fund for the withdrawal should be restricted to one usable by the specified group

-f *fund_id*

id for the fund from which the withdrawal will be made

--filter *filter_name*=*filter_value*

restricts the fund to one whose constraints do not conflict with the specified filters. For example mam-withdraw --filter User=amy will restrict the fund to one usable by the user amy. Multiple filter options may be specified which are logically anded together.

--filter-type ExactMatch|Exclusive|NonExclusive

selects the filtering type. If the exact-match filter type is used, a fund will only be matched if the specified filters exactly match the fund constraints. If the exclusive filter type is used, a fund will only be matched if the specified filters meet all constraints (not only must the filters be a non-conflicting superset of the fund constraints, but all constraint association dependencies must also be satisfied). If the non-exclusive filter type is used, a fund will be matched as long as the specified filters do not conflict with the constraints. The default filter type is NonExclusive.

-i *allocation_id*

credits will be withdrawn from the specified allocation id only. If the allocation is omitted, only credits from active allocations will be withdrawn in the order of earliest expiring first.

-m *machine_name*

The fund for the withdrawal should be restricted to one usable by the specified machine

-o *organization_name*

The fund for the withdrawal should be restricted to one usable by the specified organization

-u *user_name*

The fund for the withdrawal should be restricted to one usable by the specified user

[-z] *withdrawal_amount*

amount to withdraw. The amount may also be specified as the sole argument.

--hours

treat currency as specified in hours. In systems where the currency is measured in resource-seconds (like processor-seconds), this option allows the amount to be specified in resource-hours.

--debug

log debug info to screen

--site *site_name*

obtain response from specified site

--help

brief help message

--man

full documentation

--quiet

suppress headers and success messages

--verbose

display modified object details

--version

display product version

--about

display product information

mam-server

Synopsis

mam-server [-s, --start] [-k, --stop] [-r, --restart] [-c, --reconfig] [-l, --status] [--primary] [--backup] [-d, --debug [*debug_level*]] [--help] [--man] [--version] [--about]

Description

mam-server is a forking server that listens for and services Moab Accounting Manager client requests. It handles the startup and daemonization, shutdown and restart of the application.

Options

--backup

causes the server to start up in the backup server role. When running under the backup server role, events are disabled.

-c, --reconfig

causes the server to reread the configuration files. This can also be accomplished by sending the HUP signal to the main server process.

-k, --stop

shutdown (kill) the server. This can also be accomplished by sending the TERM signal to the main server process.

-l, --status

displays the status of the server, indicating whether it is running or stopped.

--primary

causes the server to start up in the primary server role. When running under the primary server role, events are enabled.

-r, --restart

restart the server

-s, --start

start the server. This option is assumed in the absence of a stop or restart flag and may be ommitted in a start request.

-d, --debug [*debug_level*]

log debug info to screen. An optional debug level parameter can be supplied indicating the logging threshold and can be one of TRACE, DEBUG (default), INFO, WARN, ERROR and FATAL.

--help

brief help message

--man

full documentation

--version

display product version

--about

display product information

mam-shell

Synopsis

mam-shell [--format csv|raw|standard] [--debug] [--site *site_name*] [--help] [--man] [--quiet] [--verbose] [--version] [--about] [*command*]

Description

mam-shell is an interactive control program that can access all funtionality available in Moab Accounting Manager. Commands can be invoked directly from the command line, or an interpreter can parse commands from stdin.

Commands are of the form:

Object[,**Object**...] **Action** [**Predicate**]...

Predicates are of the form:

[**Conjunction**] [**OpenParentheses**] [**Object**.]**Name Operator** [**Subject**.]**Value** [**CloseParentheses**]

Conjunctions default to "And" and include:

&& and

|| or

&! and not

|! or not

OpenParentheses may be any number of literal open parentheses '('.

The **Name** is the name of the condition, assignment or option.

The **Operator** may be one of:

== equals

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

~ matches

!~ does not match

= assignment

+= increment

-= decrement

:= option

:! negated option

The **Value** is the value of the condition, assignment, or option and may be enclosed in double quotes to enclose spaces or special characters.

CloseParentheses may be any number of literal close parentheses ')'.

The desired selections (columns to be displayed) in a query can be specified via a pseudo Show option with a value of comma-separated attribute names and may optionally include an object, operator and alias. It will be in the form: Show:="[**operator**(]

See the mam-shell chapter in the Administrator Guide for more information on constructing requests.

Options

--site *site_name*

obtain response from specified site

--format *output_format*

data output format. Valid values include standard, raw and csv.

--quiet

suppress messages and headers

--debug

log debug info to screen.

--help

brief help message

--man

full documentation

--verbose

display modified object details

--version

display product version

--about

display product information

mam-read-configuration

Synopsis

mam-read-configuration [-c | -s | -g] [-p *parameter_pattern*] [--help] [--man] [--quiet] [--version]

Description

mam-read-configuration is used to display configuration information. It simply parses the configuration files and will only d play enabled (uncommented) parameter values. If none of **-c**, **-s** or **-g** are specified, configuration parameters from all configuration files will be displayed.

Options

-c

display only client configuration parameters

-g

display only gui configuration parameters

-s

display only server configuration parameters

-p *parameter_pattern*

display only configuration parameters matching the specified pattern. The following wildcards are supported:

*

matches any number of characters

?

matches a single character

--help

brief help message

--man

full documentation

--quiet

suppress headers and parameter names

--version

display product version

mybalance

Synopsis

mybalance [--hours] [--help] [--man]

Description

mybalance is used to display balance information for the invoking user.

Options

--hours

displays balance in processor-hours (instead of processor-seconds)

--help

brief help message

--man

full documentation