

---

# **VMRC Documentation**

***Release 1.0***

**VMRC**

November 13, 2015



<b>1</b>	<b>About VMRC</b>	<b>3</b>
<b>2</b>	<b>VMRC Service</b>	<b>5</b>
2.1	Introduction . . . . .	5
2.2	Prerequisites . . . . .	5
2.3	Installing the pre-requisites . . . . .	5
2.4	Deploying the VMRC Web Service . . . . .	6
2.5	Configuring the VMRC Web Service . . . . .	7
2.6	Operations performed by the Administrator . . . . .	9
<b>3</b>	<b>VMRC Client</b>	<b>11</b>
3.1	Introduction . . . . .	11
3.2	Pre-requisites . . . . .	11
3.3	Assumptions . . . . .	11
3.4	Deploying the VMRC Client . . . . .	12
3.5	Using the VMRC Client . . . . .	13
3.6	Adding a VMI to the catalogue without uploading the VMI files . . . . .	13
3.7	Uploading the VMI files to the entry of the catalogue . . . . .	13
3.8	List all the VMI entries in the catalogue . . . . .	14
3.9	Search for the Most Appropriate VMIs . . . . .	14
3.10	Download the VMI image files to a local directory . . . . .	14
3.11	Removing a VMI from the catalogue . . . . .	14
3.12	Managing Permissions to VMI entries . . . . .	15
3.13	Operations performed by the Administrator . . . . .	15
3.14	Add a new User to the VMRC catalog . . . . .	15
3.15	Delete a User from the VMRC catalog . . . . .	15
3.16	Obtaining a list of Users from the VMRC catalog . . . . .	15
3.17	Change a User's ACL . . . . .	15
<b>4</b>	<b>The VMI Definition &amp; Query Language for VMRC</b>	<b>17</b>
4.1	Introduction . . . . .	17
4.2	General Language Description . . . . .	17
4.3	Specific Features to Describe the VMI Requirements . . . . .	18
4.4	Language Examples . . . . .	18
4.5	Specifying the Requirements to search for a VMI . . . . .	19
<b>5</b>	<b>Indices and tables</b>	<b>21</b>



Contents:



---

# About VMRC

---

### **VMRC is client-server system (based on Web Services) to index and store Virtual Machine Images (VMI)**

along with its metadata (OS, applications, etc.). It supports matchmaking to obtain the appropriate VMIs that satisfy a given set of hard (must) requirements and soft (should) requirements.

It is useful as a catalog of VMIs that can be stored on the VMI repository systems of the different Cloud Management Platforms (such as OpenNebula or OpenStack) or on public Clouds (such as Amazon Web Services). This way, customized VMIs are indexed in VMRC and applications can query, as an example, for a VMI based on Ubuntu 12.04 LTS with Java and Octave already installed.

Current version: 2.1.2

This repository only includes the VMRC server. Additional packages available in GitHub are:

- [vmrc-client](#) (CLI and Java-based API to access VMRC)
- [vmrc-web](#) (Web-based GUI to access VMRC)

The easiest and fastest way to deploy VMRC is using Docker:

- [VMRC Docker Image - with GUI \(Recommended\)](#)
- [VMRC Docker Image - no GUI](#)





---

## VMRC Service

---

### 2.1 Introduction

VMRC (Virtual Machine image Repository & Catalog) is client-server system (based on Web Services) to catalog and store Virtual Machine Images (VMI) along with its metadata (OS, applications, etc.). It supports matchmaking to obtain the appropriate VMIs that satisfy a given set of hard (must) requirements and soft (should) requirements.

This section describes the server-side component of the VMRC.

### 2.2 Prerequisites

The VMRC system requires the Java Development Kit (JDK) 1.7.0 (or greater version) and the Apache Tomcat 7 application server. Version 7.0.28 is the last version in which the VMRC Server can be deployed successfully. It has been tested that from Tomcat version  $\geq 7.0.29$  it no longer works properly.

### 2.3 Installing the pre-requisites

- Installing the Java JDK:

The easiest way to install the Java JDK in a Linux box is via the package management tools. For example, installing the Java JDK in Ubuntu requires the following command:

```
$ sudo apt-get install openjdk-7-jdk
```

You can find out where the Java JDK has been installed if you list the `/etc/alternatives/java` file. Then define the `JAVA_HOME` variable. Add the following lines at the end of the `$HOME/.bashrc` file (use your specific configuration):

```
$ export JAVA_HOME=/usr/lib/jvm/java-7-openjdk
```

Do not forget to load the environment variable by invoking:

```
$ source $HOME/.bashrc
```

- Installing the Apache Tomcat

Download and install the [Apache Tomcat 7](http://tomcat.apache.org/download-70.cgi), (<http://tomcat.apache.org/download-70.cgi>), application server (do not forget to choose version  $\leq 7.0.28$ ). Choose the binary distribution for simplicity. This way, you will only have to unpack the package into your preferred destination folder:

```
$ tar zxvf apache-tomcat-7.0.14.tar.gz
```

Define the required environment variables. Add the following lines at the end of the `$HOME/.bashrc` file (specify your precise configuration).

```
$ export TOMCAT_HOME=<path_to_the_folder>
```

Do not forget to load the environment variable by invoking:

```
$ source $HOME/.bashrc
```

## 2.4 Deploying the VMRC Web Service

Make sure you have the environment variables `JAVA_HOME` and `TOMCAT_HOME` correctly configured before attempting these steps.

1. Download the VMRC Server package (either binary or source version) from <http://www.grycap.upv.es/vmrc>
  - If you downloaded the source version, follow the installation procedure detailed in the `README.txt` file in order to generate the `vmrc.war` file
  - If you downloaded the binary version, the package includes a previously generated `vmrc.war` file.
2. About the `vmrc.war` and `vmrc-web-gui.war` files.
  - The `vmrc.war` file includes the VMRC server.
  - The `vmrc-web-gui.war` includes the simple web-based GUI. Installing this component is optional and you might find it still not available, since it will be released in the near future. You can safely skip its installation if you only want programmatic access to the VMRC server.
3. (Optional) Configure Tomcat to enable HTTPS access to the VMRC (for enhanced privacy, if required)

- Create a Java Key Store with the `keytool` utility available in the JDK

```
$ keytool -genkey -alias tomcat -keyalg RSA -keystore $TOMCAT_HOME/conf/vmrcert
```

You will have to specify the details about the certificate and a password. Use whatever password and information about the certificate you want. Include the following connector into the `$TOMCAT_HOME/conf/server.xml`, in the `Service` section.

```
<Service name="Catalina">
```

```
...
```

```
  <Connector port="8444" protocol="HTTP/1.1" SSLEnabled="true"
```

```
    maxThreads="150" scheme="https" secure="true"
```

```
    clientAuth="false" sslProtocol="TLS"
```

```
    keystoreFile="/opt/apache-tomcat-7.0.12/conf/vmrcert"           key-  
    storePass="p9i8uj7"/>
```

```
...
```

```
</Service>
```

4. Deploy the `vmrc.war` and `vmrc-web-gui.war` into Tomcat.
  - Just drop both files into `$TOMCAT_HOME/webapps`
5. Start Tomcat
  - `$TOMCAT_HOME/bin/startup.sh`
6. Verify that it has been successfully deployed.

- Access the VMRC endpoint at:
  1. <http://localhost:8080/vmrc/vmrc> (HTTP)
  2. <https://localhost:8444/vmrc/vmrc> (HTTPS)
- Access the VMRC Web GUI to ensure it is up and running (notice that you won't be able to access the Web GUI unless you deployed the vmrc-web-gui.war file into Tomcat)
  1. <http://localhost:8080/vmrc-web-gui> (HTTP)
  2. <https://localhost:8444/vmrc-web-gui> (HTTPS)

Notice that the server listens at ports 8080 (http) and 8444 (https). You can change the ports by modifying the \$TOMCAT\_HOME/conf/server.xml Connector sections.

#### 7. Shutdown the VMRC service

This can be achieved by shutting down Tomcat.

- \$TOMCAT\_HOME/bin/shutdown.sh

## 2.5 Configuring the VMRC Web Service

All the data employed by VMRC is available in the following folder:

\$TOMCAT\_HOME/webapps/vmrc

From now on, this path will be denoted \$VMRC\_SERVER\_HOME, although you do not need to define such variable.

### Change the Default Administrator User and Password

1. Change the default VMRC admin name and/or password
  1. Modify the \$VMRC\_SERVER\_HOME/WEB-INF/classes/vmrc.properties
 

```
admin_password=password1
```
2. Restart Tomcat
 

```
$TOMCAT_HOME/bin/shutdown.sh
```

```
$TOMCAT_HOME/bin/startup.sh
```

### Database Configuration

The VMRC Server comes preconfigured with an in-memory HSQLDB database. This means that no database is required to be configured in order to test the functionality of the VMRC catalog. However, once you stop Apache Tomcat, all the data will be gone. Therefore, do not plan to use it for production purposes.

You can configure another database backend (such as MySQL) by changing the configuration in \$TOMCAT\_HOME/webapps/vmrc/WEB-INF/classes/hibernate.cfg.xml

You should only specify the connection details to the database. The DB schema will be automatically created upon the service startup the next time you restart Apache Tomcat.

Sample configuration is provided on that file for both HSQLDB and MySQL databases. Since Hibernate is employed as the persistence tool, the underlying backend should be supported by Hibernate (typically any SQL-based oriented DB).

### Database Configuration for MySQL

To use MySQL as the database backend you can use the following instructions (for Ubuntu):

1. Install MySQL server
  1. `sudo apt-get install mysql-server`
2. Connect to MySQL as the root user (and the password specified during installation)
  1. `mysql -u root -p`
3. Create the database and the user that will be used to connect from VMRCServer
  1. `mysql> create database vmrc;`
  2. `mysql> create user vmrc identified by 'password';`
  3. `mysql> grant all on vmrc.* to vmrc@localhost;`
4. Modify `$VMRC_SERVER_HOME/WEB-INF/classes/hibernate.cfg.xml` to switch from HSQLDB to MySQL
  1. Uncomment the MySQL section and comment the HSQLDB section
  2. Specify the values for:
 

```
<property name="hibernate.connection.username">vmrc</property>
<property name="hibernate.connection.password">password</property>
<property name="hibernate.connection.url">jdbc:mysql://localhost/vmrc</property>
```
5. Restart the VMRCServer service
  1. `$TOMCAT_HOME/bin/shutdown.sh`
  2. `$TOMCAT_HOME/bin/startup.sh`

### Permissions Model

The VMRC Server supports a permission model that enables to authorize specific operations to registered users and on a per-VMI basis.

### VMI permissions

Each VMI has one Access Control List (ACL) with the following permissions:

PER-MIS-SION	DESCRIPTION	VAL-UES	DEF. VALUE
LIST	This VMI will be included in the listings performed by a user with the 'LIST' permission granted.	owner   all	all
SEARCH	This VMI will be included in the list of VMIs obtained by the search operation performed by a user with the permission granted. If set to owner, only the owner of the VMI is allowed to obtain this VMI as the result of a search operation.	owner   all	all
UP-LOAD	A user with the 'UPLOAD' permission granted will be able to upload a file for this VMI. If set to owner, only the owner of the VMI is allowed to obtain this VMI as the result of a list operation.	owner   all	owner
DOWN-LOAD	A user with the 'DOWNLOAD' permission granted will be able to download the file for this VMI. If set to owner, only the owner of the VMI is allowed to download the VMI.	owner   all	all
DELETE	A user with the 'DELETE' permission granted will be able to delete this VMI. If set to owner, only the owner of the VMI is allowed to delete this VMI.	owner   all	owner
ADD	A user with the 'ADD' permission granted would be able to update the values of the VMI. If set to owner, only the VMI owner is allowed to update. This feature is currently unimplemented.	owner   all	owner

### User permissions

Each user has one ACL with the following permissions:

PERMISSION	DESCRIPTION	VALUES	DEF. VALUE
LIST	The user is allowed to list the VMIs which include the ‘LIST’ permission.	owner   all   none	all
SEARCH	The user is allowed to search the VMIs which include the ‘SEARCH’ permission.	owner   all   none	all
UPLOAD	The user is allowed to upload files for the VMIs which include the ‘UPLOAD’ permission.	owner   all   none	owner
DOWN-LOAD	The user is allowed to download files from the VMIs which include the ‘DOWNLOAD’ permission.	owner   all   none	all
DELETE	The user is allowed to delete the VMIs which include the ‘DELETE’ permission.	owner   all   none	owner
ADD	The user is allowed to register a VMIs in the catalog.	owner   all   none	owner

### Special Users

ADMIN	
PERMISSION	VALUE
LIST	all
SEARCH	all
UPLOAD	all
DOWNLOAD	all
DELETE	all
ADD	all

ANONYMOUS	
PERMISSION	VALUE
LIST	none
SEARCH	all
UPLOAD	all
DOWNLOAD	none
DELETE	none
ADD	none

### Firewall Configuration

The VMRC Server uses port 21000 for FTP transfers.

## 2.6 Operations performed by the Administrator

All the following commands require using the \$VMRC\_SERVER\_HOME/bin/vmrc-admin.sh tool and admin client-side credentials. In a Windows platform, the admin CLI tool can be invoked as follows:

```
java -cp vmrc-client.jar org.grycap.vmrc.client.cmd.admin.VMRCAdminCLI --aduser john johndoe
```

### **Add a new User to the VMRC Server**

To add a new user called john with password johndoe, you have to issue the following command:

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --adduser john johndoe
```

### **Delete a User from the VMRC Server**

To delete user john, the following command is required:

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --deleteUser john
```

### **Obtaining a list of Users from the VMRC Server**

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --listUsers
```

You can obtain extended information by producing XML output.

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --listUsers --xml
```

### **Change a User's ACL**

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --userAcl username operation perm
```

Where operation = [add|list|upload|search|delete] perm = [all|owner|none]

Please refer to the previous section for a detailed explanation of the permissions model.

---

## VMRC Client

---

### 3.1 Introduction

VMRC (Virtual Machine image Repository & Catalog) is client-server system (based on Web Services) to index Virtual Machine Images (VMI) along with its metadata (OS, applications, etc.). It supports matchmaking to obtain the appropriate VMIs that satisfy a given set of hard (must) requirements and soft (should) requirements.

This section describes the command-line client (CLI) of the VMRC.

### 3.2 Pre-requisites

The binary version of the VMRC client has been compiled with Java JDK 1.7+. Therefore, it requires a JRE (Java Runtime Environment) 1.7+. The source version might be compiled with Java JDK 1.5+. Make sure you have the environment variable `JAVA_HOME` correctly configured before attempting these steps. This guide assumes that you will use a Unix-based OS to run the VMRC client (such as GNU/Linux or OS/X). Since it has been developed in Java it might work in other operating systems.

### 3.3 Assumptions

This guide assumes that a VMRC server has been properly configured at the `deployed_vmrc` host (by default *localhost*). Therefore, the following URI is available:

- VMRC WS endpoint: [http://the\\_deployed\\_vmrc\\_host:8080/vmrc/vmrc](http://the_deployed_vmrc_host:8080/vmrc/vmrc)

The following URI might also be available in case you installed the GUI:

- VMRC Web GUI: [http://the\\_deployed\\_vmrc\\_host:8080/vmrc-web-gui](http://the_deployed_vmrc_host:8080/vmrc-web-gui)

Notice that if an HTTPS connector has been defined in Tomcat, then the following URLs can also be employed:

- VMRC GUI (HTTPS): [https://the\\_deployed\\_vmrc\\_host:8444/vmrc-web-gui](https://the_deployed_vmrc_host:8444/vmrc-web-gui)
- VMRC WS endpoint: [https://the\\_deployed\\_vmrc\\_host:8444/vmrc/vmrc](https://the_deployed_vmrc_host:8444/vmrc/vmrc)

The `vmrc-web-gui` is not required for the client to work. Only the server is required.

The command line client can seamlessly work with HTTP or HTTPS. Use whichever you want. If the VMRC server and client are located on the same machine, you can safely use `localhost` in the previous URLs.

The VMRC client can seamlessly run from any platform with Java support. This includes GNU/Linux, OS X and Windows platforms. Being developed in Java, it should work on other platforms as well.

## 3.4 Deploying the VMRC Client

1. The recommended approach to install the VMRC Client is to check out the latest version from the GitHub repository
1. `git clone https://github.com/grycap/vmrc-client.git`
2. Alternatively, you can download a pre-compiled version of the VMRC client from the web page.
  1. Download the `vmrc-client-<version>.bin.tar.gz` from the aforementioned web site.
  2. Unpack the package into a destination folder

We will assume that the VMRC client has been uncompressed (or checked out) at the `$HOME/vmrc-client` folder.

3. Set the appropriate environment variables.
    1. Add at the end of the `$HOME/.bashrc` file the following lines:  

```
export VMRC_CLIENT_LOCATION=$HOME/vmrc-client
export PATH=$PATH:$VMRC_CLIENT_LOCATION
```
    2. Reload this configuration  
b.i. `source $HOME/.bash`
  4. Set the CLI scripts executable
1. `chmod u+x $VMRC_CLIENT_LOCATION/tool/*.sh`
  5. Setup your user credentials
    1. Create the file `$HOME/.vmrc/vmrc_auth` with your credentials

```
johndoe:mypassword
```

Notice that the johndoe user must exist in the VMRC server. If no valid credentials are supplied, the VMRC client defaults to the “anonymous” user which has reduced privileges. If you do not have your user credentials yet, review the section “Operations performed by the Administrator” to learn how to create them.

6. Verify that the remote VMRC Server catalog can be contacted.

Issue the following command, where `VMRC_URL=https://the_deployed_vmrc_host:8444/vmrc/vmrc:`

```
./vmrc.sh -list -uri $VMRC_URL
```

If no error arises, then the VMRC client has been successfully deployed.

You might receive Log4J related errors and warnings but you can safely ignore them.

In Windows, you can invoke the CLI as follows

```
java -jar vmrc-client.jar -list
```

If you want to avoid specifying the `VMRC_URL` in each CLI invocation, you can define the `$HOME/.vmrc/vmrc.properties` file with the following content:

```
vmrc.uri = https://the\_deployed\_vmrc\_host:8444/vmrc/vmrc
```

This way, the command can be issued as follows:

```
./vmrc.sh -list
```



## 3.5 Using the VMRC Client

The following commands assume these environment variables properly defined with your specific configuration. In particular, remember that the machine icaro.i3m.upv.es does not exist:

```
export VMRC_URL=https://icaro.i3m.upv.es:8444/vmrc/vmrc
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk/
```

## 3.6 Adding a VMI to the catalogue without uploading the VMI files

This method creates a new entry in the catalogue to describe a VMI. This operation does not involve any file uploading to the VMRC repository.

```
./vmrc.sh --add ../src/test/vmis/sample1.vmi
```

The file my\_vmi.vmi describes the VMI. This is a sample file:

```
system.name = MyImage7

system.hypervisor = vmware

system.location = /opt/vm_images/dummy_img.qcow2

cpu.arch = i686

disk.size = 5000

disk.os.name = Linux

disk.os.flavour = Ubuntu

disk.os.version = 11.15

l1l1disk.os.credentials.user = user2

disk.os.credentials.password = passwd2

disk.applications contains (name = com.mathworks.matlab, version = 8.0 )

disk.applications contains (name = net.nbcr.opal, version = 2.2 )

disk.applications contains (name = com.java, version = 1.6, path = /usr/local/bin/java )
```

## 3.7 Uploading the VMI files to the entry of the catalogue

This method uploads the file related to a VMI into VMRC. VMware disks might be split in different files whereas KVM image files are just a single file. Therefore, VMWare disks should be compressed in a bundle (a single file) before uploading the file to VMRC.

```
./vmrc.sh --upload $HOME/images/myvmi.img --vmi MyImage7
```

## 3.8 List all the VMI entries in the catalogue

This method lists all the VMI entries in the catalogue that can be listed considering the credentials supplied by the client. It obtains an XML description of the VMI entries.

```
./vmrc.sh -list
```

## 3.9 Search for the Most Appropriate VMIs

This method searches for the most appropriate VMIs in the VMRC catalogue that satisfy the requirements imposed by the user. Hard requirements will certainly be met by the VMI. Soft requirements will also be considered according to the user ranking. It obtains a ranked XML description of the (up to 10) VMIs that satisfy those requirements.

```
./vmrc.sh -search req1.vmiq
```

The specified file expresses the requirements that the VMI should met in order to be listed. Here comes a sample requirements file:

```
system.hypervisor = kvm

cpu.arch = i686

disk.os.name = Linux

disk.os.flavour = Ubuntu

disk.os.version >= 11.15

disk.applications contains (name = com.java, version >= 1.6)

soft 25 disk.applications contains (name = net.nbcr.opal, version > 2.0)
```

---

By default, requirements are considered ‘hard ‘ and these must be satisfied by the VMI. The soft requirements can be ranked by the user. If you need further information about this language, please refer to specific document that describes it.

## 3.10 Download the VMI image files to a local directory

This method downloads the specified VMI to a local directory in the client machine. The VMI should be stored in the VMRC repository. Otherwise, this command will fail.

```
./vmrc.sh -download /tmp/my_img.img -vmi MyImage7
```

## 3.11 Removing a VMI from the catalogue

This command deletes an entry in the VMRC catalogue. If there is a related VMI image in the repository it also deletes it.

```
./vmrc.sh -delete MyImage7
```

## 3.12 Managing Permissions to VMI entries

Please refer to the VMRC Server document for further information about the permission model.

This is the syntax of the command:

```
./vmrc.sh --vmiAcl <vmi_name> <operation> <perm>
```

Where operation=[list | search | upload | download | delete | add ] and perm = [owner | all]

## 3.13 Operations performed by the Administrator

All the following commands require using the \$VMRC\_SERVER\_HOME/bin/vmrc-admin.sh tool and admin client-side credentials. In a Windows platform, the admin CLI tool can be invoked as follows:

```
java -cp vmrc-client.jar org.grycap.vmrc.client.cmd.admin.VMRCAdminCLI --aduser john johndoe
```

These commands can be executed from any machine with network access to a VMRC Server, not only from the machine that hosts the VMRC Server.

## 3.14 Add a new User to the VMRC catalog

To add a new user called john with password johndoe, you have to issue the following command:

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --adduser john johndoe
```

## 3.15 Delete a User from the VMRC catalog

To delete user john, the following command is required:

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --deleteUser john
```

## 3.16 Obtaining a list of Users from the VMRC catalog

To obtain a list of users from the VMRC catalog, you can use the following command:

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --listUsers
```

You can obtain extended information by producing XML output.

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --listUsers --xml
```

## 3.17 Change a User's ACL

```
$VMRC_SERVER_HOME/bin/vmrc-admin.sh --userAcl username operation perm
```

Where operation = [add|list|upload|search|delete] perm = [all|owner|none]

Please refer to the VMRC server's section for a detailed explanation of the permissions model.



---

## The VMI Definition & Query Language for VMRC

---

### 4.1 Introduction

This document describes the language employed to express the capabilities of the Virtual Machine Images (VMIs) and the language employed to query the catalog to search for the appropriate VMIs that satisfy the given specified requirements.

### 4.2 General Language Description

The language is structured as a key-value one, where the properties and features of the VMI are expressed. To specify the requirements that a VMI has to meet, the following operators can also be employed (( $<$ ,  $<=$ ,  $>$ ,  $>=$ ,  $\&\&$  and  $\parallel$ ). The keys are hierarchically structured where the dot (.) is employed as a separator between the category and the subcategory.

The values are not subject to a previously defined ontology. Instead, the user is free to assign the specific semantics to each attribute considering the application domain. This way, the language can be employed in different fields or future areas. As an example, restricting the possible values to name an Operating System (OS) would require modification of the language when a new OS appears.

The categories and their specific subcategories are explained:

- system
  - Includes general information about the VMI
  - Possible keys:
    - \* system.name: The name of the VI
    - \* system.hypervisor: The hypervisor for which this VMI was built.
    - \* system.location: An URL that indicates where this VMI can be located (it can be a file:// or an [http://](#) or even a user-defined such as ec2://)
- cpu
  - Includes information about the CPU features of the VMI
  - Possible keys:
    - \* cpu.arch. The CPU architecture.
- disk
  - Includes information about disk of the VMI.

– Possible keys:

- \* `disk.size`: The size of VMI's disk.
- \* `disk.os.name` : Type of Operating System (i.e. Windows, Linux, Mac OS X)
- \* `disk.os.flavour` : Flavour of the OS (i.e. XP, Ubuntu, Leopard)
- \* `disk.os.version` : Version of the OS (i.e., SP1, 9.10, 10.5.4)
- \* `disk.os.credentials.user` : User name to access the VMI
- \* `disk.os.credentials.password` : Password of the account.
- \* `disk.applications` : The applications installed in the VMI

Use the following syntax to specify the applications:

- `disk.applications contains (name = com.mathworks.matlab, version = 8.0, path = /opt/matlab/bin/matlab8)`

The path is an optional attribute.

## 4.3 Specific Features to Describe the VMI Requirements

To describe the application requirements, these can be classified in two groups by the user.

- **Hard requirements:** These are the requirements that must be satisfied by the VMI (i.e. Operating System)
- **Soft requirements:** These are optional features that would be interesting to be met by the VMI, but that they are not mandatory. For example, this is the case of specific application that could be already installed in the VMI. This is typically not a mandatory requirement, since contextualization can be employed to deploy the application afterwards.

By default, requirements are considered hard unless otherwise specified by the user. The soft requirements are ranked by the user. This rank is completely dependent on the user, who expresses the rank value assigned to each soft requirement. This way, the user can indicate which soft requirements are more important than others in order to perform matchmaking.

If you want to specify an application requirement, use the following syntax.

`disk.applications contains (name = Java, version = 1.6)`

In this case, it is possible to use the operators (`<`, `<=`, `>`, `>=`) to specify a greater range of possible values.

## 4.4 Language Examples

### Describing a VMI

This section exemplifies the usage of the language. The chunk of code, which is shown below, provides the description of a VMI created by the KVM hypervisor with a CPU architecture i686, and a disk size of 10000 Mbytes. The operating system is Ubuntu Linux 11.10 with a user account created with the username `jdoe` and password `janed0e!!`. The VMI has the following applications installed (java-jdk version 1.6, ant version 1.8.1. and Apache Tomcat version 7.0.1).

```
system.name = MyImage7

system.hypervisor = kvm
```

```
system.location = /opt/vm_images/dummy_img.qcow2

cpu.arch = i686

disk.size = 5000

disk.os.name = Linux

disk.os.flavour = Ubuntu

disk.os.version = 11.10

disk.os.credentials.user = jdoe

disk.os.credentials.password = janed0e!!

disk.applications contains (name = org.apache.tomcat, version = 7.0.1 )

disk.applications contains (name = org.apache.ant, version = 1.8.1 )

disk.applications contains (name = com.java-jdk, version = 1.6, path = /bin/java )
```

It is important to point out that the VMRC catalog doesn't check that the actual features of the VMI specified by the user actually match the existing VMI. The user is responsible for providing this correct information.

In addition, the meaning of the attributes can be altered for a given application domain. For example, the attribute `system.disk` might represent the amount of free available space (instead of the disk size). Of course, changes in the semantics must be taken into account by the user.

Additionally, declaring an application without version can be used as a simple and effective mechanism to tag a VM. These can be of importance during the matchmaking process in the case the users desires a specific VMI.

## 4.5 Specifying the Requirements to search for a VMI

The following code example shows how the language should be employed to specify the requirements that a given VMI should met in order to be a candidate one to be selected by the matchmaking process. The VMI must have been created for the KVM hypervisor. The VMI must have a disk size greater than 5000 Mbytes.

```
system.hypervisor = kvm

cpu.arch = i686

disk.os.name = Linux

disk.os.flavour = Ubuntu

disk.os.version >=9.10

disk.applications contains (name = com.mathworks.matlab)

disk.applications contains (name = org.java-jdk, version >= 1.6)

soft 25 disk.applications contains (name = org.apache.tomcat, version > 7.0)
```

Concerning the software, the OS should be Linux and, if it is possible, an Ubuntu Linux greater or equal to 9.10. The matlab application should be installed (no specific version). It also requires the Java JDK 1.6+. Additionally, the existence of the Apache Tomcat application installed (version greater than 7.0) should be ranked favorable.



---

## Indices and tables

---

- `genindex`
- `search`