# VISHNU D1.0 - General specifications

**COLLABORATORS**

| | | | |
|---|---|---|---|
| | *TITLE* :<br>VISHNU D1.0 - General specifications | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | Benjamin Isnard, Daouda Traoré, Eugène Pamba Capo-Chichi, and Kevin Coulomb | December 21, 2010 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| 01 | 07/12/2010 | Formatting example | B.Isnard |
| 02 | 13/12/2010 | Pre-delivrable | B.Isnard |

# Contents

# List of Figures

# Chapter 1

# Document presentation

## 1.1 Document objectives

This document presents the external specifications of the Vishnu system at a general level. At this level, we describe the interaction of a user with the system without providing implementation details. The different steps that constitute the scenario are detailed as well as the content of the messages exchanged. The main objective is to describe the system from the user point of view.

These general specifications are a prerequisite for the detailed specifications step in the software development process.

## 1.2 Document structure

The document is divided into 4 parts corresponding to the 4 modules that compose the Vishnu system:

- UMS: Users Management System
- TMS: Tasks Management System
- FMS: Files Management System
- IMS: Information Management System

Each module corresponds to a chapter in the document, and each chapter contains two sections:

- A first section containing "Use case descriptions" that follow the standard UML description of a use case
- A second section containing the "Use case diagrams" that describe the organization of the different use cases. These diagrams follow the UML2.0 standard.

## 1.3 References

## 1.4 Glossary

# Chapter 2

# Use cases for User Management System (UMS)

## 2.1 Use case descriptions

### 2.1.1 U1 - Session with manual closure

| | |
|---|---|
| Title | U1 - Session with manual closure |
| Summary | The user opens a new session and closes it manually |
| Actors | User |
| Precondition | - The user is authenticated<br>- VISHNU is installed and running on the client system |
| Postcondition | - The session state is closed<br>- A session log has been created<br>- All user requests submitted within the session are completed |
| Base sequence | 1. Include::U1.1 Open session<br>2. System is ready to process user commands<br>3. Include::U1.2 Close session (before the maximum inactivity delay if option CLOSE_POLICY is equal to CLOSE_ON_TIMEOUT) |
| Branch sequence | 2a. U1.3 Execute synchronous user request<br>2b. U1.4 Execute asynchronous user request<br>2c. U1.5 Reconnect to session |
| Exception sequence | 1a. Include::U1.1 exceptions<br>3a. If session cannot be closed due to running commands, user must wait until all commands are completed before trying step 3 again |
| Extensions | U1.3 - Execute synchronous user request<br>U1.5 - Reconnect to session<br>U1.4 - Execute asynchronous user request |

### 2.1.2 U1.1 - Open session

| | |
|---|---|
| Title | U1.1 - Open session |
| Summary | The user opens a session |
| Actors | User |
| Precondition | - The user is connected on a client host in which VISHNU is installed and that can be connected to the VISHNU infrastructure |

| | |
|---|---|
| Postcondition | - A session is active<br>- The user's environment contains a session certificate |
| Base sequence | 1. User provides login, password and optionnally the way of closing the session manually or automatically (on disconnect or on timeout) to the "connect" command (when the default option is not set the closing mode is manual)<br>2. System validates login, password (User is authenticated) and optionnally, the name of the closing mode (CLOSE_ON_DISCONNECT or CLOSE ON TIMEOUT) if the SESSION_CLOSE_POLICY is set.<br>3. System creates the session and activates it<br>4. System provides the session certificate to the user |
| Branch sequence | 2a. If the password is a temporary password (after reset by the administrator) the System asks the user to enter a new password, then asks for a confirmation, and registers the new password if both steps are ok. If non-interactive request then this is an exception (a change password request is required). |
| Exception sequence | 2a. The user login is unknown<br>2a1. The System returns an error message<br>2b. The user password is invalid<br>2b1. The System returns an error message<br>2c. The SESSION_CLOSE_POLICY option is unknown<br>2c1. The System returns an error message<br>2d. VISHNU infrastructure is unreachable or unavailable<br>2d1. The System returns an error message<br>2e. The user password is temporary and request is non-interactive: the System returns an error message |

### 2.1.3 U1.2 - Close session

| | |
|---|---|
| Title | U1.2 - Close session |
| Summary | The user closes the session manually |
| Actors | User |
| Precondition | - The user is connected on the client system<br>- The user has an open session on the client system |
| Postcondition | - The session is closed<br>- A session log has been created<br>- All user requests submitted during the session are completed |
| Base sequence | 1. The System checks that there are no running commands within the session<br>2. The System closes the session<br>3. The System informs the user that the session has been closed |
| Branch sequence | |
| Exception sequence | 1a. If there are running commands within the session, the System informs the user that the session can not be closed |

### 2.1.4 U1.3 - Execute synchronous user request

| | |
|---|---|
| Title | U1.3 - Execute synchronous user request |
| Summary | The user submits a synchronous request to the System |
| Actors | User |

| | |
|---|---|
| Precondition | - A session (for the current user and client host) is active |
| Postcondition | - The request is completed<br>- A request log is created |
| Base sequence | 1. The user sends the request to the System<br>2. The System returns the results to the user |
| Branch sequence | |
| Exception sequence | 1.a Invalid session (bad session certificate or unavailable session)<br>1.b Invalid request<br>1.c Permission denied (admin request issued by normal user)<br>1.d Ressource not available<br>1.e VISHNU System crashed |
| Extension of | U1 - Session with manual closure<br>U3 - Session with automatic closure on disconnect<br>U2 - Session with automatic closure on timeout |

### 2.1.5 U1.3.1 - Configure Option

| | |
|---|---|
| Title | U1.3.1 - Configure Option |
| Summary | The user wants to modify the value of an option attached to his/her VISHNU account |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1. The user sends a request for modifying the value of an option to the System<br>2. System registers the new value for the option<br>3. System returns an acknowledgment to the usere |
| Branch sequence | |
| Exception sequence | 2a. Invalid option name<br>2b. Invalid option value |

### 2.1.6 U1.3.2 - Display options

| | |
|---|---|
| Title | U1.3.2 - Display options |
| Summary | The user displays all options concerning his/her VISHNU account |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1. The user sends a request to list all his/her options<br>2. The System returns all options of the user |
| Branch sequence | |
| Exception sequence | |

### 2.1.7 U1.3.3 - Change password

| | |
|---|---|
| Title | U1.3.3 - Change password |
| Summary | The user wants to change his/her password |
| Actors | User |
| Precondition | |
| Postcondition | - The password is changed |

| | |
|---|---|
| Base sequence | - The user sends a request containing a new password<br>- The System registers the new user's password<br>- The System returns an acknowledgment to the user |
| Branch sequence | |
| Exception sequence | |

### 2.1.8 U1.3.4 - Display session command history

| | |
|---|---|
| Title | U1.3.4 - Display session command history |
| Summary | The user displays all the commands sent during one session |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1 - The user sends a request containing the session id<br>2 - The System returns the list of all commands issued by the user during the session which id corresponds to the provided session id. Each command has exactly the same format and parameters as the original submission and can be resubmitted as-is to the System. |
| Branch sequence | |
| Exception sequence | 2a - Invalid session id (unknown / belonging to another user, if the current user is not an administrator) |

### 2.1.9 U1.3.5 - Display sessions log

| | |
|---|---|
| Title | U1.3.5 - Display sessions log |
| Summary | The user displays his/her sessions (active or closed) |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | - The user sends a request to list all his/her sessions (active and/or closed) that have an open timestamp within an interval provided by the user (start and finish date)<br>- The System returns all (or only active, or only closed) sessions of the user matching the search criteria with the following information for each session: id, date of opening, client host name, closure policy (timeout or disconnect), time before automatic closure (if applicable) |
| Branch sequence | |
| Exception sequence | |

### 2.1.10 U1.4 - Execute asynchronous user request

| | |
|---|---|
| Title | U1.4 - Execute asynchronous user request |
| Summary | Tshe uer submits an asynchronous request to the system |
| Actors | User |
| Precondition | - A session (for the current user and client host) is active |
| Postcondition | - The request is completed<br>- A request log is created |

| | |
|---|---|
| Base sequence | 1. The user sends the request to the system<br>2. The System returns an acknowledgment to the user<br>3. The System runs the request in the background<br>4. When the request is completed the system updates the status of the request |
| Branch sequence | |
| Exception sequence | 1.a Invalid session (bad session certificate or session unavailable)<br>1.b Invalid request<br>1.c Permission denied<br>1.d Ressource not available<br>1.e VISHNU System crashed |
| Extension of | U1 - Session with manual closure<br>U2 - Session with automatic closure on timeout<br>U3 - Session with automatic closure on disconnect |

### 2.1.11 U1.5 - Reconnect to session

| Title | U1.5 - Reconnect to session |
|---|---|
| Summary | The user wants to use a session in which he/she was disconnected previously without closing it |
| Actors | User |
| Precondition | - The user is connected on a client host in which VISHNU is installed and that can be connected to the VISHNU infrastructure |
| Postcondition | - A session is active<br>- The user's environment contains a session certificate |
| Base sequence | 1. User provides its login, password and the short identifier of the session (in fact, for each session, a short identifier is assigned) to the System<br>2. The System validates the user's login, password and the identifier of the session<br>3. The System provides the chosen session certificate to the user |
| Branch sequence | 1a. If the user is already within an active session then go to step 3 directly. The current session will be automatically closed according to the UC U2 or U3 depending on the close policy for that session. |
| Exception sequence | cf U1.1 (Open session)<br>2.f The identifier of the session is nonexistent<br>2f1. The System returns an error message<br>2.g The identifier relates to a session belonging to another user<br>2g1. The System returns an error message<br>2.h The identifier is for a session closed<br>2h1. The System returns an error message |
| Extension of | U1 - Session with manual closure<br>U2 - Session with automatic closure on timeout<br>U3 - Session with automatic closure on disconnect |

### 2.1.12 U2 - Session with automatic closure on timeout

| Title | U2 - Session with automatic closure on timeout |
|---|---|
| Summary | The user opens a new session that is closed by the System after the inactivity delay has expired |

| | |
|---|---|
| Actors | User |
| Precondition | - VISHNU is installed and running on the client system<br>- The client system can be connected to VISHNU<br>- The option SESSION_CLOSE_POLICY is CLOSE_ON_TIMEOUT (either user option is defined or this is the default policy) |
| Postcondition | - A session log has been created<br>- The session state is closed<br>- All user requests submitted during the session are complete |
| Base sequence | 1. U1.1 Open session<br>2. The System is ready to process user commands<br>3. After inactivity delay has expired: U1.3 Close session auto |
| Branch sequence | 2a. U1.3 Execute synchronous user request<br>2b. U1.4 Execute asynchronous user request<br>2c. U1.5 Reconnect to session<br>2d. If the user disconnects from the client terminal or the client system crashes or is shutdown, the session remains open and all asynchronous commands that were not completed are kept running |
| Exception sequence | see U1 |
| Extensions | U1.5 - Reconnect to session<br>U1.4 - Execute asynchronous user request<br>U1.3 - Execute synchronous user request |

### 2.1.13 U2.1 - Close session auto

| | |
|---|---|
| Title | U2.1 - Close session auto |
| Summary | The session is closed by the system |
| Actors | |
| Precondition | - The user is connected on the client system<br>- The user has an open session on the client system either the inactivity timeout for the session has expired or the user disconnected from its shell session |
| Postcondition | - The session is closed<br>- The session close event is stored in the system’s log |
| Base sequence | 1. The system checks if the user has got no running commands (file transfers or tasks)<br>2. The system registers session closure |
| Branch sequence | 1a. If the user has got some running commands, the system does not close the session and reset the inactivity timeout. After this delay is expired, back to step 1. |
| Exception sequence | |

### 2.1.14 U3 - Session with automatic closure on disconnect

| | |
|---|---|
| Title | U3 - Session with automatic closure on disconnect |
| Summary | The user opens a new session that will be closed by the system after the user disconnects from the client terminal |
| Actors | User |

| | |
|---|---|
| Precondition | - VISHNU is installed and running on the client system<br>- The client system can be connected to VISHNU<br>- The option SESSION_CLOSE_POLICY is CLOSE_ON_DISCONNECT (either user option is defined or this is the default policy) |
| Postcondition | - A session log has been created<br>- The session state is closed<br>- All user requests submitted during the session are complete |
| Base sequence | 1. U1.1 Open session<br>2. System is ready to process user commands<br>3. The user disconnects from the terminal (either by typing an exit command, by closing the window, or by remote disconnection)<br>4. U1.3 Close session auto |
| Branch sequence | 2a. U1.4 Execute synchronous user request<br>2b. U1.5 Execute asynchronous user request<br>3a. if the client system crashes or is shutdown, go to step 4 |
| Exception sequence | see U1 |
| Extensions | U1.3 - Execute synchronous user request<br>U1.5 - Reconnect to session<br>U1.4 - Execute asynchronous user request |

### 2.1.15 U4 - Create new local user config

| | |
|---|---|
| Title | U4 - Create new local user config |
| Summary | The user creates a new local user config for a given user on a given machine |
| Actors | User |
| Precondition | - The user has an account on VISHNU<br>- The user has no local user config defined for the machine |
| Postcondition | - Local user config is registered<br>- An email is sent to the user with a message containing a SSH public key |
| Base sequence | 1. The user provides local user config information for a given machine: user login (unix), home directory<br>2. The System generates a ssh private/public key pair<br>3. The System sends an email to the user containing the public key and asking the user to add this key to the authorized_keys on the machine<br>4. The user updates his/her authorized_keys file on the machine by adding the public key |
| Branch sequence | |
| Exception sequence | 1a. Invalid login<br>3a. Invalid email address |

### 2.1.16 U4.1 - Update local user config

| | |
|---|---|
| Title | U4.1 - Update local user config |
| Summary | The user updates his/her local user config for a given machine |
| Actors | User |
| Precondition | - The user has an account on VISHNU<br>- The user has a local user config defined for the machine |

| | |
|---|---|
| Postcondition | - local user config is updated |
| Base sequence | 1. The user provides the identifier of his/her local config and information to be updated<br>2. The System updates the local user account information<br>3. The System returns an acknowledgment to the user |
| Branch sequence | |
| Exception sequence | 1a. Invalid account (unknown or inactive) |

### 2.1.17 U4.2 - Delete local user config

| | |
|---|---|
| Title | U4.2 - Delete local user config |
| Summary | The user deletes his/her local user config on a given machine |
| Actors | User |
| Precondition | - The local user config exists for the given machine<br>- There is no job or file transfer running on the given machine |
| Postcondition | - The local user config for the given machine is deleted |
| Base sequence | 1. The user provides the identifier of the local user config<br>2. The System deletes the local user config identified<br>3. The System returns an acknowledgment to the user |
| Branch sequence | |
| Exception sequence | 1a. Invalid login (unknown or inactive) |

### 2.1.18 U4.3 - Display local user configs

| | |
|---|---|
| Title | U4.3 - Display local user configs |
| Summary | The user displays all of his/her local configs |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1. The user sends a request to list all his/her local configs<br>2. The System returns all local configs |
| Branch sequence | |
| Exception sequence | |

### 2.1.19 UA1 - Create new user account

| | |
|---|---|
| Title | UA1 - Create new user account |
| Summary | The administrator creates a new user account in the System (database) |
| Actors | Admin |
| Precondition | - The user has not an account on VISHNU |
| Postcondition | - The user account is created and in an active state<br>- The account's password must be changed at the first connection |

| | |
|---|---|
| Base sequence | 1. The administrator provides the new user's last name, first name, email address and specifies wether the user has standard or admin rights<br>2. The System creates the new user account and initializes the password with a random-generated string (temporary password)<br>3. The System sends an email to the user containing the temporary password<br>4. The System returns an acknowledgment to the Admin |
| Branch sequence | |
| Exception sequence | 2a. Login already used by another active user<br>3a. Invalid email address |

### 2.1.20 UA1.1 - Update user account

| | |
|---|---|
| Title | UA1.1 - Update user account |
| Summary | The administrator updates the user account (database) |
| Actors | Admin |
| Precondition | - The user has an account on VISHNU |
| Postcondition | - The user account is updated |
| Base sequence | 1. The administrator provides the user's information changes<br>2. The System updates user account (database)<br>3. The System returns an acknowledgment to the administrator |
| Branch sequence | |
| Exception sequence | 1.a Invalid login or login unknown<br>1.b The user information set are invalid<br>1.c The user information set are not consistent with the System (inconsistency between user id and login) |

### 2.1.21 UA1.2 - Delete user account

| | |
|---|---|
| Title | UA1.2 - Delete user account |
| Summary | The administrator deletes a user account |
| Actors | Admin |
| Precondition | - The user has an account on VISHNU<br>- There is no job or file transfer running for the user |
| Postcondition | - The user account is no longer in the System<br>- System does not contain any information related to the user |
| Base sequence | 1. The administrator provides a user's login<br>2. The System deletes the user account together with all the information (configuration, history) related to it.<br>3. The System returns an acknowledgment to the administrator |
| Branch sequence | |
| Exception sequence | 1a. Invalid login (unknown or inactive) |

### 2.1.22 UA2 - Reset user password

| | |
|---|---|
| Title | UA2 - Reset user password |
| Summary | The administrator resets a user password |

| | |
|---|---|
| Actors | Admin |
| Precondition | |
| Postcondition | - The password of the user is temporary and must be changed at the first connection by the user |
| Base sequence | 1. The administrator provides a user's login<br>2. The System resets the user's password using a random-generated string<br>3. The System sends an email to the user containing the new temporary password<br>4. The System returns an acknowledgment to the administrator |
| Branch sequence | |
| Exception sequence | 1a. Invalid login (unknown or inactive)<br>3a. Invalid email address |

### 2.1.23 UA3 - Save configuration

| | |
|---|---|
| Title | UA3 - Save configuration |
| Summary | The administrator saves the configuration of the system |
| Actors | Admin |
| Precondition | |
| Postcondition | - The configuration is saved on a file<br>- A log information is created |
| Base sequence | 1. The administrator requests to save the configuration in a file<br>2. The System creates a configuration file containing: the list of users, the list of local users configs and the list of machines according to the local users configs<br>3. The Systems saves the file on the client host |
| Branch sequence | |
| Exception sequence | 2a. File creation problems<br>2b. VISHNU System crashed |

### 2.1.24 UA4 - Restore configuration

| | |
|---|---|
| Title | UA4 - Restore configuration |
| Summary | The administrator restores a saved configuration |
| Actors | Admin |
| Precondition | - All users are disconnected from VISHNU<br>- The configuration file was saved using the "save configuration" feature. |
| Postcondition | - The System is operational on all the machines that are both properly configured in the saved configuration and where the VISHNU processes are running. |
| Base sequence | 1. The administrator opens a session as the Root user<br>2. The administrator checks that there is no other user/admin connected to VISHNU<br>3. The administrator loads the configuration file<br>4. The System replaces the current configuration with the loaded configuration. |
| Branch sequence | |

| | |
|---|---|
| Exception sequence | 1a. If the Root user already has an open session, the System cannot open a second session with the Root user<br>3a. If the configuration file cannot be loaded, the System provides an error message. The System configuration is reset to a basic configuration with only the Root user configured. |

### 2.1.25 UA5.1 - Display sessions

| | |
|---|---|
| Title | UA5.1 - Display sessions |
| Summary | The administrator displays all past and present sessions stored in the database. |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | - The administrator sends a request to list all sessions (active and/or closed) that have an open timestamp within an interval provided by the user (start and finish date)<br>- The System returns the list of sessions that match the search criteria and provides detailed information about these sessions (user id, open and close timestamp, client machine id) |
| Branch sequence | |
| Exception sequence | |

### 2.1.26 UA5.2 - Display users

| | |
|---|---|
| Title | UA5.2 - Display users |
| Summary | The administrator displays the description of all users registered in the database |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | 1. The administrator sends a request to list all users<br>2. The System returns all users with the following information for each user: id, firstname, lasname, login, status, email and password state. |
| Branch sequence | |
| Exception sequence | |

### 2.1.27 UA5.3 - Display local users configs

| | |
|---|---|
| Title | UA5.3 - Display local users configs |
| Summary | The administrator displays the local user configs for all users registered in the database |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | - The administrator sends a request to list all local users configs<br>- The System returns all the local users configs for all users |
| Branch sequence | |
| Exception sequence | |

### 2.1.28 UA6.1 Add a machine

| | |
|---|---|
| Title | UA6.1 Add a machine |
| Summary | An administrator registers a new machine |
| Actors | Admin |
| Precondition | |
| Postcondition | The infrastructure has one more machine |
| Base sequence | 1) The administrator add a machine at the infrastructure giving :<br>- The machine name<br>- The machine state (private or accessible to users)<br>- The public adress (IP)<br>- A structure describing the machine state<br>- A structure describing the network<br>2) The machine is added to the infrastructure and the machine ID is returned. |
| Branch sequence | |
| Exception sequence | For base sequence :<br>* A machine with the same name exists, the ALREADYREGISTERED exception is raised.<br>* A machine with the same public adress already exists, the ALREADYREGISTERED exception is raised<br>* The machine is not reachable, the NOTREACHABLE exception is raised. |

### 2.1.29 UA6.2 Remove a machine

| | |
|---|---|
| Title | UA6.2 Remove a machine |
| Summary | An administrator unregisters a machine |
| Actors | Admin |
| Precondition | - the machine is registered in the system |
| Postcondition | - ?? preciser l'état de la BDD |
| Base sequence | 1) The administrator remove a machine from the infrastructure giving the machine id |
| Branch sequence | |
| Exception sequence | For remove :<br>* The public adress is unknown, the UNNOWNMACHINE exception is raised. |

### 2.1.30 UA6.3 Display machines

| | |
|---|---|
| Title | UA6.3 Display machines |
| Summary | |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | |
| Branch sequence | |
| Exception sequence | |

## 2.2 Use case diagrams

### 2.2.1 UC UMS Admin

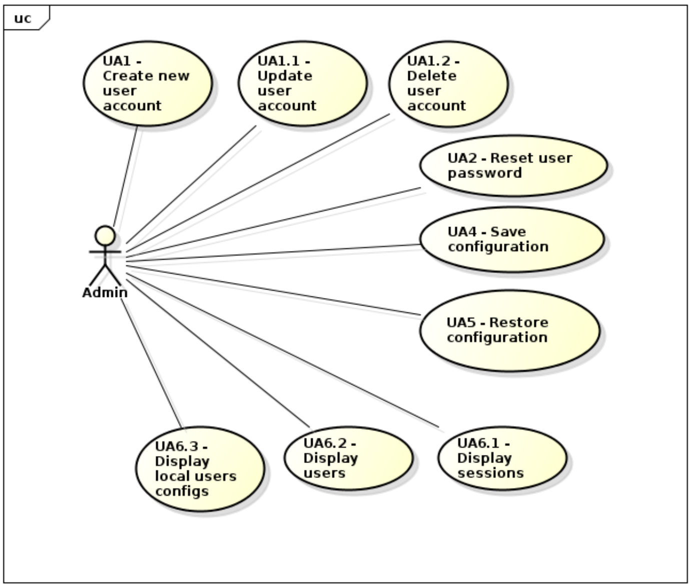
uc
UA1 - Create new user account
UA1.1 - Update user account
UA1.2 - Delete user account
UA2 - Reset user password
UA4 - Save configuration
UA5 - Restore configuration
Admin
UA6.3 - Display local users configs
UA6.2 - Display users
UA6.1 - Display sessions

Figure 2.1: UC UMS Admin

### 2.2.2 UC UMS Admin Machines

UMS Admin Machines.png UMS Admin Machines.png

Figure 2.2: UC UMS Admin Machines

### 2.2.3 UC UMS User Auto

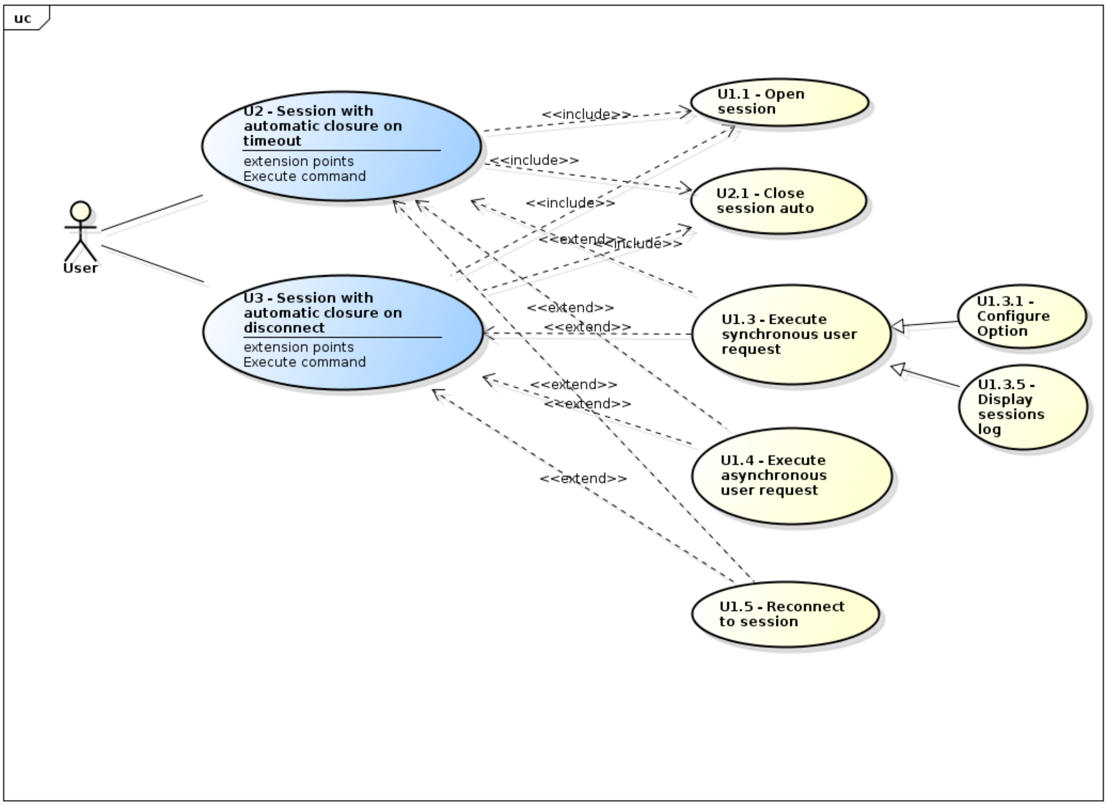
uc
User
U2 - Session with automatic closure on timeout
extension points
Execute command
U3 - Session with automatic closure on disconnect
extension points
Execute command
U1.1 - Open session
U2.1 - Close session auto
U1.3 - Execute synchronous user request
U1.3.1 - Configure Option
U1.3.5 - Display sessions log
U1.4 - Execute asynchronous user request
U1.5 - Reconnect to session
<<include>>
<<include>>
<<include>>
<<include>>
<<extend>>
<<extend>>
<<extend>>
<<extend>>
<<extend>>
<<extend>>

Figure 2.3: UC UMS User Auto

### 2.2.4 UC UMS User Manual

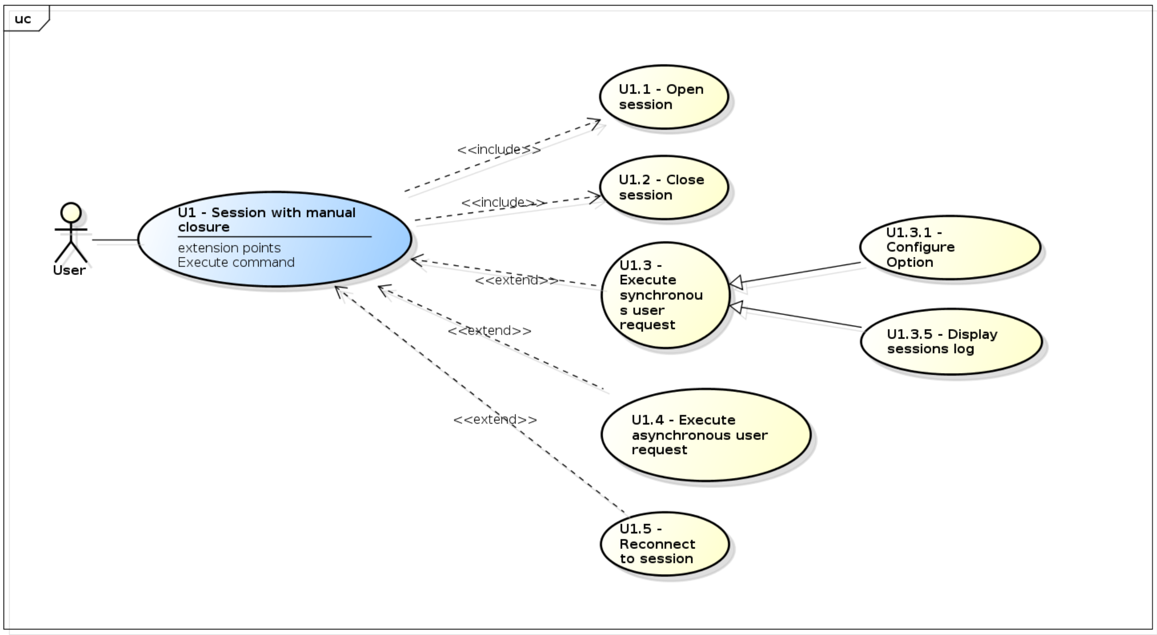
uc
User
U1 - Session with manual closure
extension points
Execute command
<<include>>
<<include>>
<<extend>>
<<extend>>
<<extend>>
U1.1 - Open session
U1.2 - Close session
U1.3 - Execute synchronous user request
U1.3.1 - Configure Option
U1.3.5 - Display sessions log
U1.4 - Execute asynchronous user request
U1.5 - Reconnect to session

Figure 2.4: UC UMS User Manual

### 2.2.5 UC UMS User account

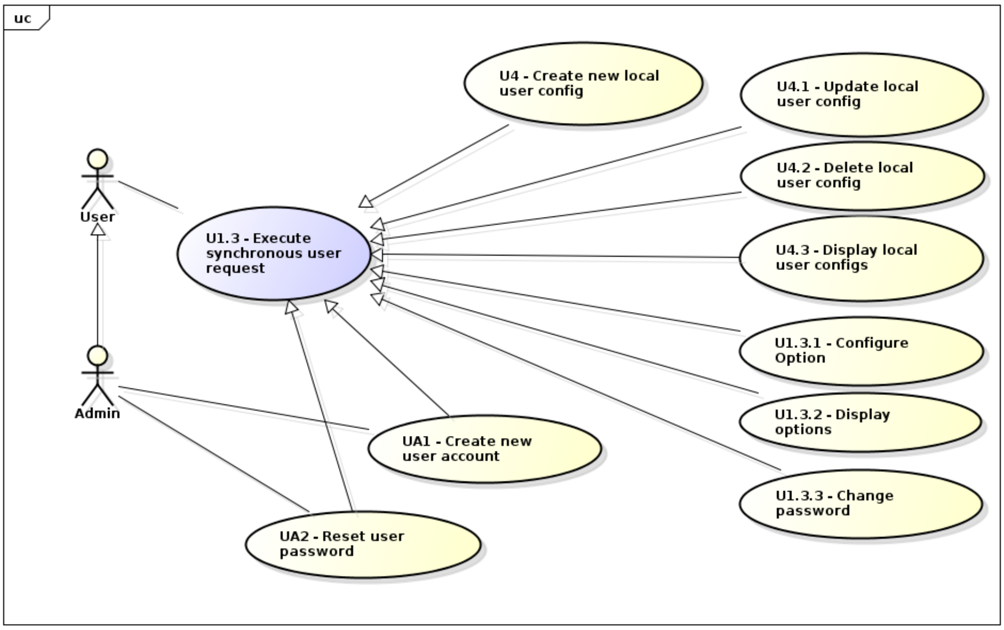
uc
User
Admin
U1.3 - Execute synchronous user request
U4 - Create new local user config
U4.1 - Update local user config
U4.2 - Delete local user config
U4.3 - Display local user configs
U1.3.1 - Configure Option
U1.3.2 - Display options
U1.3.3 - Change password
UA1 - Create new user account
UA2 - Reset user password

Figure 2.5: UC UMS User account

## 2.3 Data dictionary

- **Configuration** : The configuration contains all information about the users and machines registered in the database. It does not contain chronological information about the users or the infrastructure (logs, metrics values)
- **Local user config** : The local user config is the description of the given user on a specific machine described in the database
- **Option** : The option is a parameter of the user account that is not mandatory. Default value for each option is defined by the administrator. This features can be used by all VISHNU modules (not only UMS).
- **Password state** : Records the current state of the password of a user: either ’temporary’ if the password must be changed next time the user connects to the System, or ’valid’ if the password is in a normal state.
- **Root user** : Special user that is pre-configured in the VISHNU system and that has administrator privileges. This user cannot be deleted from the system.
- **Session** : A session is the context in which VISHNU commands are executed (ex: job submission, file transfers). It is created following authentifcation of a user and lasts until the session is closed either manually or automatically.
- **User account** : The user account is the description in the database of a VISHNU user

# Chapter 3

# Use cases for Tasks Management System (TMS)

## 3.1 Use case descriptions

### 3.1.1 T1 - AsyncCommandOnMachine

| | |
|---|---|
| Title | T1 - AsyncCommandOnMachine |
| Summary | User starts an asynchronous command on a given machine |
| Actors | User |
| Precondition | - User has an active open session |
| Postcondition | - The command is in active state until completed<br>- The system log has been updated and contains the request parameters |
| Base sequence | 1. User sends the request with parameters including session id and machine id<br>2. System checks that the machine id is valid and machine is available<br>3. System checks that the session id is valid<br>4. If command parameters contain a file the System verifies that the file is available and readable<br>5. System returns information to the user about the request status<br>6. System records request information (time, user, machine, request parameters) in the system log |
| Branch sequence | 5a. T1.1 SubmitJob |
| Exception sequence | 1a. The TMS server is unavailable<br>- The system returns an error message<br>2a. The name of the given machine is unknown<br>-The system prints an error message that informs the user<br>3a. The session id is not valid<br>- The system prints an error message that informs the user.<br>- The user revises the id.<br>4a. The path to a file parameter is invalid<br>- The system prints an error message that informs user<br>- The user revises the path |
| Extensions | T1.1 - SubmitJob |

### 3.1.2 T1.1 - SubmitJob

| Title | T1.1 - SubmitJob |
|---|---|
| Summary | User submits a job on a given machine |
| Actors | User |
| Precondition | |
| Postcondition | - The job is submitted on the specified machine<br>- The job state and id are recorded on the system's log<br>- The job id is sent to the user |
| Base sequence | 1. The System checks that request parameters contain:<br>- job script path<br>- job options<br>2. The TMS server on the given machine is contacted<br>3. The job is submitted by the TMS server to the batch scheduler<br>4. The id of the submitted job is returned to the user |
| Branch sequence | |
| Exception sequence | 1a. Invalid options or script<br>4a. The batch scheduler server is unavailable<br>4b. The batch scheduler server rejects the request |
| Extension of | T1 - AsyncCommandOnMachine |

### 3.1.3 T2 - SyncCommandOnMachine

| Title | T2 - SyncCommandOnMachine |
|---|---|
| Summary | User executes a synchronous command on a given machine |
| Actors | User |
| Precondition | - User has an active open session |
| Postcondition | - Request is in completed state<br>- The system log has been updated and contains the request parameters |
| Base sequence | 1. User sends the request with parameters including session id and machine id<br>2. System checks that the machine id is valid and machine is available<br>3. System checks that the session id is valid<br>4. If command parameters contain a file the System verifies that the file is available and readable<br>5. System returns information containing the results of the request<br>6. System records request information (time, user, machine, request parameters) in the system log |
| Branch sequence | 5a. T2.1-GetJobs<br>5b. T2.2-CancelJob<br>5c. T2.3-ListQueue<br>5d. T2.4-ListJobs<br>5e. TA1-SetMachineRefreshRate<br>5f. TA2-SetMachineEnv |

| | |
|---|---|
| Exception sequence | 1a. The TMS server is unavailable<br>- The system returns an error message<br>2a. The name of the given machine is unknown<br>-The system prints an error message that informs the user<br>3a. The session id is not valid<br>- The system prints an error message that informs the user.<br>- The user revises the id.<br>4a. The path to a file parameter is invalid<br>- The system prints an error message that informs user<br>- The user revises the path |
| Extensions | T2.1 - GetJob<br>T2.2 - CancelJob<br>T2.3 - ListQueue<br>T2.4 - ListJobs<br>TA1 - SetMachineRefreshPeriod<br>TA2 - SetMachineEnv |

### 3.1.4 T2.1 - GetJob

| | |
|---|---|
| Title | T2.1 - GetJob |
| Summary | User requests the TMS server to get information about a job |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1. The Systems checks the job id<br>2. The TMS server on the given machine is contacted<br>3. The TMS server asks job information to the batch schduler server<br>4. The user receives job information |
| Branch sequence | |
| Exception sequence | 1a. The job id is invalid<br>3a. The batch scheduler server is unavailable<br>3b. The batch scheduler server rejects the request |
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.5 T2.2 - CancelJob

| | |
|---|---|
| Title | T2.2 - CancelJob |
| Summary | The user cancels a job |
| Actors | User |
| Precondition | |
| Postcondition | - The job is canceled on the specified machine<br>- The job state and id are removed to the system's log<br>- An information is sent to the user |
| Base sequence | 1. The systems checks the job id<br>2. The TMS server on the given machine is contacted<br>3. The TMS server cancels the job<br>4. The user receives a message |
| Branch sequence | |
| Exception sequence | 1a. The job id is invalid<br>3a. The batch scheduler server is unavailable<br>3b. The batch scheduler server rejects the request |
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.6 T2.3 - ListQueue

| | |
|---|---|
| Title | T2.3 - ListQueue |
| Summary | User lists all queues or classes of a specific batch scheduler |
| Actors | User |
| Precondition | |
| Postcondition | - The system collects the informations on each queue or classes<br>- The system records request parameters in system's log<br>- The system send the list contained the information on all queues to the user. |
| Base sequence | 1. The TMS server on the given machine is contacted<br>2. The TMS server asks queues or classes information to the batch schduler server<br>3. The System sends all queues list to the user |
| Branch sequence | |
| Exception sequence | 2a. The batch scheduler server is unavailable<br>2b. The batch scheduler server rejects the request |
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.7 T2.4 - ListJobs

| | |
|---|---|
| Title | T2.4 - ListJobs |
| Summary | User lists all jobs submitted |
| Actors | User |
| Precondition | -User has an active open session |
| Postcondition | - The System sends information on all jobs to the user<br>- The System records request parameters in the system's log |
| Base sequence | 1. The TMS server on the given machine is contacted<br>2. The TMS server asks job information to the batch schduler server<br>3. The System sends full information on all jobs to the user |
| Branch sequence | |
| Exception sequence | 2a. The batch scheduler server is unavailable<br>2b. The batch scheduler server rejects the request |
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.8 TA1 - SetMachineRefreshPeriod

| | |
|---|---|
| Title | TA1 - SetMachineRefreshPeriod |
| Summary | The admin sets the refresh period of output and error file content |
| Actors | Admin |
| Precondition | |
| Postcondition | - The refresh period value is stored by the system |
| Base sequence | 1. System saves the refresh period for the given machine.<br>2. System applies the refresh period to all current jobs and future requests |
| Branch sequence | |
| Exception sequence | 1a. Refresh period value is too short (minimum value : see technical requirements)<br>- System returns an error message |

| | |
|---|---|
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.9 TA2 - SetMachineEnv

| | |
|---|---|
| Title | TA2 - SetMachineEnv |
| Summary | The admin sets an environment variable |
| Actors | Admin |
| Precondition | |
| Postcondition | - Environment variable is set |
| Base sequence | 1. System saves the environment variable for the given machine.<br>2. System applies the environment variable to all current jobs and future requests |
| Branch sequence | |
| Exception sequence | |
| Extension of | T2 - SyncCommandOnMachine |

### 3.1.10 TA3 - LaunchTmsServer

| | |
|---|---|
| Title | TA3 - LaunchTmsServer |
| Summary | The administrator launches the VISHNU TMS server on a given machine |
| Actors | Admin |
| Precondition | - The Vishnu server software (TMS Module and dependencies) is installed on the machine<br>- The machine is configured in the Vishnu system database<br>- The batch scheduler processes are up and running on the same machine<br>- The network connection between the machine and the Vishnu database server is up and running |
| Postcondition | - The TMS server is up and running<br>- A server log has been created |
| Base sequence | 1. Admin connects to the machine as vishnu user<br>2. Admin updates the Vishnu configuration if necessary (database server hostname and credentials, DIET configuration, Batch scheduler configuration)<br>3. Admin launches the Vishnu TMS Server executable<br>4. System checks the connections to its peers within the Vishnu platform<br>5. System retrieves the list of active jobs (not completed jobs) that were launched on the same machine<br>6. System checks that all the active jobs (from previous step) are still running on the batch scheduler, and eventually updates the job status (for ex. from waiting to running, or from running to finished)<br>7. System returns a status message to the administrator |
| Branch sequence | |
| Exception sequence | 4a. A connection to a Vishnu peer is down. System returns an error message and stops<br>6a. The batch scheduler does not recognize some job ids. In this case the System updates the job status to completed. |

## 3.2 Use case diagrams

### 3.2.1 UC TMS Overview

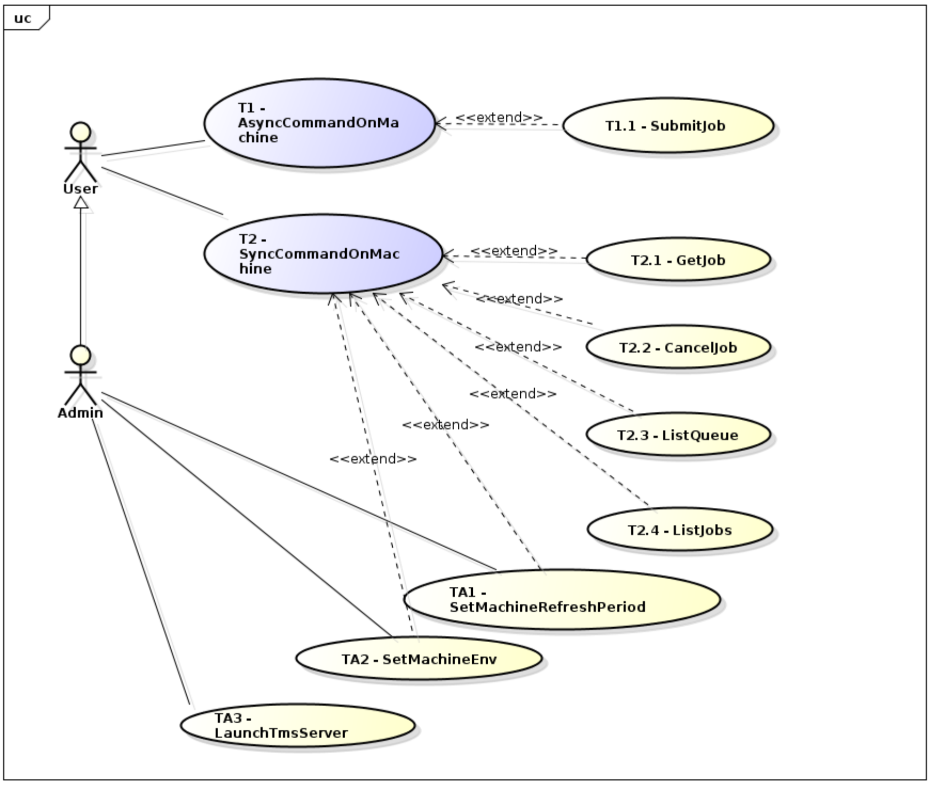

Figure 3.1: UC TMS Overview

## 3.3 Data dictionary

- **Batch Scheduler** : A batch scheduler is a distributed resource manager that enables to allocate at best the resources to the jobs on a machine according to user needs (the needs are spiciefed by the user by batch directives (batch options) in file or command line).
- **Job** : A job is a sequence of instructions (included batch scheduler directives) written to launch and to perform by a specified batch scheduler.
- **Job id** : A job id allows to identifie the job in the batch scheduler system.
- **JobPath** : A jobPath is the path to the file (script) containing the instructions (batch directives or job characteristiques, job execution command) of the job.
- **Queue ou Classe** : A queue or class allows to associate the resource limits (CPU wallclock time, CPU memory) and execution priorities of the jobs.
- **TMS** : Task Management System

# Chapter 4

# Use cases for Information Management System (IMS)

## 4.1 Use case descriptions

### 4.1.1 I1. Get the update frequency

| | |
|---|---|
| Title | I1. Get the update frequency |
| Summary | The user gets how often the IMS database is updated |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1) The user calls the function to know how often the IMS database is automatically updated<br>2) The System returns the value in second |
| Branch sequence | |
| Exception sequence | The frequency is not defined by the admins, he gets the default value used. |

### 4.1.2 I2 Get metric data

| | |
|---|---|
| Title | I2 Get metric data |
| Summary | The user gets data concerning the metrics on a machine |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | 1) The user calls to get the metrics data. on a machine identified by a machine id, for a metric type, from start time up to duration seconds. The metrics are within {number of cpu, number of free cpu, total diskSpace, free diskSpace, total RAM, free RAM, number of processes running}<br>2) The System returns the results by groups (metric,value, time). |
| Branch sequence | |
| Exception sequence | Fail to get the metrics, an UNREACHABLECOMPONENT exception is raised |

### 4.1.3 I3. Export and replay commands

| | |
|---|---|
| Title | I3. Export and replay commands |
| Summary | The user exports and replays a sequence of commands made during a session. |
| Actors | User |
| Precondition | |
| Postcondition | All the commands submitted during a session have been re-executed keeping the same order they were originally launched. |
| Base sequence | 1) The user calls to export the history of a session identified by an id in python format<br>2) The System provides a python script containing all the commands of the session with the same parameters as provided initially by the user (including file paths, numbers, strings, options)<br>3) The user executes the python script in VISHNU |
| Branch sequence | 1a) The user calls to export the history of a session identified by an id in shell format.<br>2a) The System provides a shell script containing all the commands of the session with the same parameters as provided initially by the user (including file paths, numbers, strings, options)<br>3a) The user executes the shell script in a shell |
| Exception sequence | 1 -> The session id is invalid, an INVALIDPARAMETER exception is raised.<br>3 -> A command in the execution fails, a COMMANDERROR is raised |

### 4.1.4 I4. Get data on the infrastructure

| | |
|---|---|
| Title | I4. Get data on the infrastructure |
| Summary | The user gets System information about the machines |
| Actors | User |
| Precondition | |
| Postcondition | |
| Base sequence | B1 * The user calls to get the use of the CPUs on a machine identified by an ID.<br>* The System returns him the use of the CPUs on the machine in percentage. |

| | |
|---|---|
| Branch sequence | B2 * The user calls to get the number of CPUs on a machine identified by an ID.<br>* The System returns him the number of CPUs on the machine.<br>---<br>B3 * The user calls to get the total diskSpace.<br>* The System returns him the value.<br>---<br>B4 * The user calls to get the free diskSpace.<br>* The System returns him the value.<br>---<br>B5 * The user calls to get the total RAM on a machine identified by an id.<br>* The System returns him an id.<br>---<br>B6 * The user calls to get the free RAM on a machine identified by an id.<br>* The System returns him the value |
| Exception sequence | The machine id is invalid, an INVALIDPARAMETER exception is raised |

### 4.1.5 IA1. Get the running processes

| | |
|---|---|
| Title | IA1. Get the running processes |
| Summary | The admin gets the list of the running processes on a machine |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | 1) The admin calls to get the list of the processes on a machine referenced by a machine id<br>2) The System returns a list of processes |
| Branch sequence | 2 -> No process is running on the machine, the admin gets an empty list |
| Exception sequence | 1 -> machineId is invalid, an INVALIDPARAMETER exception is raised. |

### 4.1.6 IA2. Define a system load treshold

| | |
|---|---|
| Title | IA2. Define a system load treshold |
| Summary | The administrator defines a system load treshold for a machine |
| Actors | Admin |
| Precondition | |
| Postcondition | The system load treshold is added to the System database |

| | |
|---|---|
| Base sequence | 1a) The administrator calls to define the limit size of the diskSpace to use with a machine id, a treshold value and an admin id<br>2a) The System updates the database<br>OR-----------------------<br>1b) The administrator calls to define the limit of RAM available to he user with a machine id, a treshold value and an admin id<br>2b) The System updates the database<br>OR-----------------------<br>1c) The administrator calls to define the number of processes treshold on a machine with a machine id, a treshold value and an admin id<br>2c) The System updates the database |
| Branch sequence | |
| Exception sequence | * The admin ID is invalid, the database is not updated and a INVALIDPARAMETER exception is raised<br>* The modification of the database fails, a DATABASEEXCEPTION is raised. |

### 4.1.7 IA2.1 Get a system load treshold

| | |
|---|---|
| Title | IA2.1 Get a system load treshold |
| Summary | The user wants to get the tresholds on a machine |
| Actors | Admin |
| Precondition | |
| Postcondition | |
| Base sequence | 1) The user calls to get the defined limit on a machine identified by an id. These limits are within {free diskSpace, free RAM, number of processes running}<br>2) The System returns the value. |
| Branch sequence | |
| Exception sequence | * No limit is defined for this machine, the user gets an empty list.<br>* The machine id is invalid, the user gets an INVALIDPARAMETER exception |

### 4.1.8 IA3. Define the identifiers

| | |
|---|---|
| Title | IA3. Define the identifiers |
| Summary | The administrator defines the format of the automatic identifiers defined for the System objects. |
| Actors | Admin |
| Precondition | |
| Postcondition | A new format will be used to create the new identifiers |

| | |
|---|---|
| Base sequence | 1) The administrator has a list of variables to define the identifiers shape. He has a method by kind of object (an object is either a user or a machine or a task or a file transfer).<br>Available variables are :<br>YEAR : the last two digits, (e.g. 10 for 2010)<br>MONTH : Numerical value of the month (from 1 to 12)<br>DAY : Day number, from 1 to 31<br>TYPE : The object kind<br>SITE : The place for machine/users<br>NAME : Username or machine name<br>CPT : A counter automatically increased (each kind of object has its counter).<br>2) He calls the function to redefine the format with some of the previous parameters in a string. For example, "$TYPE$DAY$MONTH$YEAR$CPT"<br>3) The System database is updated, the System does not check if the given format creates unique identifiers. If the same identifier is created, it will degenerate the database (the key will not be unique) |
| Branch sequence | 2-> An invalid variable is given, an INVALIDPARAMETER exception is raised and the old format is still used |
| Exception sequence | |

### 4.1.9 IA4. Hard delestage

| | |
|---|---|
| Title | IA4. Hard delestage |
| Summary | Abruptly stop the processes running on a machine (the waiting actions are cancelled and the running ones are cut). |
| Actors | Admin |
| Precondition | |
| Postcondition | The whole machine is flushed |
| Base sequence | 1) The admin launches the hard delestage function on a machine identified by an id.<br>2) The System flushes all the waiting action.<br>3) The System stops all the running processes on this machine. |
| Branch sequence | |
| Exception sequence | |

### 4.1.10 IA5. Soft delestage

| | |
|---|---|
| Title | IA5. Soft delestage |
| Summary | The admin purges all the waiting actions and stops the running ones |
| Actors | Admin |
| Precondition | |
| Postcondition | No jobs are waiting to run or running |
| Base sequence | 1) The admin calls the soft delestage function on the machine identified by an id.<br>2) The System flushes the waiting jobs and stops the running ones. They are stored and can be restarted later |
| Branch sequence | |
| Exception sequence | |

### 4.1.11 IA6. Set the update frequency

| | |
|---|---|
| Title | IA6. Set the update frequency |
| Summary | The administrator sets the update frequency |
| Actors | Admin |
| Precondition | |
| Postcondition | The System updates the IMS database at the new frequency |
| Base sequence | 1) The administrator calls to set the update frequency in seconds<br>2) The System updates its database update frequency value |
| Branch sequence | |
| Exception sequence | The database is is not reachable. A DATABASEERROR is raised. |

### 4.1.12 IA7. Notification of limit overflow

| | |
|---|---|
| Title | IA7. Notification of limit overflow |
| Summary | The admin is informed of a limit overflow |
| Actors | Admin |
| Precondition | A machine on the System has a limit overflow |
| Postcondition | |
| Base sequence | 1) The System gets the email adress of the admin to contact<br>2) The System sends a mail to the admin concerning the overflow. The mail contains the name of the machine and the treshold concerned. |
| Branch sequence | |
| Exception sequence | Sending the mail fails, a MAILERROR exception is raised. |

### 4.1.13 IA8. Restart the System

| | |
|---|---|
| Title | IA8. Restart the System |
| Summary | Restart all the servers, agents, and daemons of the System. The started actions are restarted again. |
| Actors | Admin |
| Precondition | The System platforms needs to be restarted |
| Postcondition | The System is running with the same server, agents and daemons that were running before the crash. The interrupted actions are restarted from the beginning. |
| Base sequence | 1) An admin detects a problem<br>2) An admin calls to restart the System<br>3) The System saves the current actions<br>4) The System restarts components and restarts the stopped action from the beginning |
| Branch sequence | |
| Exception sequence | 4-> Fail to relaunch a component (server, client, agent), an UNREACHABLECOMPONENT exception is raised. |

### 4.1.14 IA9. Automatic restart

| | |
|---|---|
| Title | IA9. Automatic restart |
| Summary | A component is restarted |
| Actors | Admin |

| | |
|---|---|
| Precondition | A component of the platform is down |
| Postcondition | The component is up and running again |
| Base sequence | 1) An admin detects a component has stopped for unknown reasons (a component = server, daemon, agent)<br>2) The admin calls the System to relaunche the component with its name<br>3) The System relauches the component |
| Branch sequence | |
| Exception sequence | 3-> Fail to restart the component, an UNREACHABLECOMPONENT exception is raised. |

### 4.1.15 U1.3 Execute synchronous request

| | |
|---|---|
| Title | U1.3 Execute synchronous request |
| Summary | The user subsmits a synchronous request to the System. c.f. the UMS use case description (U1.3) |
| Actors | User, Admin, Admin |
| Precondition | |
| Postcondition | |
| Base sequence | |
| Branch sequence | |
| Exception sequence | |

## 4.2 Use case diagrams

### 4.2.1 Consult

Figure 4.1: Consult

### 4.2.2 Fonctionnalities Global

Global.png Global.png

Figure 4.2: Fonctionnalities Global

### 4.2.3 PlatformManagement

Figure 4.3: PlatformManagement

### 4.2.4 Replay

Figure 4.4: Replay

### 4.2.5 Stop/Restart sequences

sequences.png sequences.png

Figure 4.5: Stop/Restart sequences

### 4.2.6 UseCase Diagram0

Diagram0.png Diagram0.png

Figure 4.6: UseCase Diagram0

## 4.3 Data dictionary