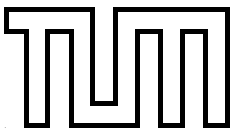


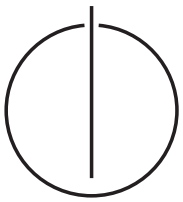
Technische Universität München

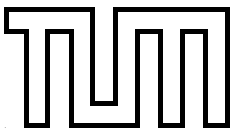
Fakultät für Informatik

Bachelorarbeit in Informatik

Simulation and Visualization of Procedures in distributed IT-Infrastructures

Young chul Jung





Technische Universität München

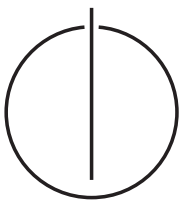
Fakultät für Informatik

Bachelorarbeit in Informatik

Simulation and Visualization of Procedures in distributed IT-Infrastructures

Simulation und Visualisierung von Abläufen in verteilten IT-Infrastrukturen

Bearbeiter:	Young chul Jung
Supervisor:	Prof. Dr. Heinz-Gerd Hegering
Advisor:	Ralf König Roland Hartinger (FSC)
Abgabedatum:	15. October 2008



I assure the single handed composition of this bachelor's thesis only supported by declared resources.

Munich, 15. October 2008

. . . . . .
(Signature of candidate)

Abstract

The distributed system is used in many IT-fields. Because it consists of a lot of processors and huge storage device, it is a suitable for complex computing solution by company, which uses many applications at the same time. For management of this system many tasks are necessary. These tasks consist of sequential execution of single activities. For automation of tasks I have defined preconditions of system and procedure of activities for each tasks. I have designed basic tasks for life cycle management of distributed system. And with the simulator it easy to understand, how tasks works.

Contents

1 Intro

1.1 Motivation

All data of the day have become more and more diverse and huge. Additionally application of computer user is also getting more varied and more. A high-performance computer is necessary in order to deal and calculate with such data and applications. Of course, computer's performance is also getting better, but it is not enough to cover the virtually unlimited data and applications.
So usually for this today's situation it is used, to connect multiple computers, and they process applications or handle data together at the same time. Afterwards these computers share the results together. It is basic idea of the distributed system. The distributed system is not one of new technology, but it is much used in the field of IT-infrastructure. For this reason its research is continually required.

The definition of distributed system[1] is, the active interacting of processors, which have no common memories, and they communicate with each other via messages.

We are calling separate one processor or some processors and its own memory together as a server. Servers operate applications, and these applications can be chosen by user or can be required for installed hardware component. They can share and also modify same data. Therefore diverse relationships are existent between applications, e.g. conflict, complement relationship.
Because of these relationships and limitation of hardware resource, the distribution of applications must be handled carefully for stable computing. If applications are wrong or instable distributed in system, it can cause errors of applications. And user can't trust any more this system.
In order to manage server and application, varied tasks for theirs life cycle required. The tasks consist of a series of sequential and various activities. The topic of my research is modeling of tasks for life cycle management in distributed system. These tasks are categorized according to the managed objects and theirs roles. And they will be finally visualized and simulated.

1.2 Principle

A typical task consists of 4 phases, (cf. Figure 1.1). All activities of a task will be run in these 4 phases. And activities use results of other activities, which are processed already before. They follow procedure of the task.

[1] `http://de.wikipedia.org/wiki/Verteiltes_System`

1. Phase is monitor phase.
2. Phase is analyze phase.
3. Phase is plan and decide phase.
4. Phase is execute phase.

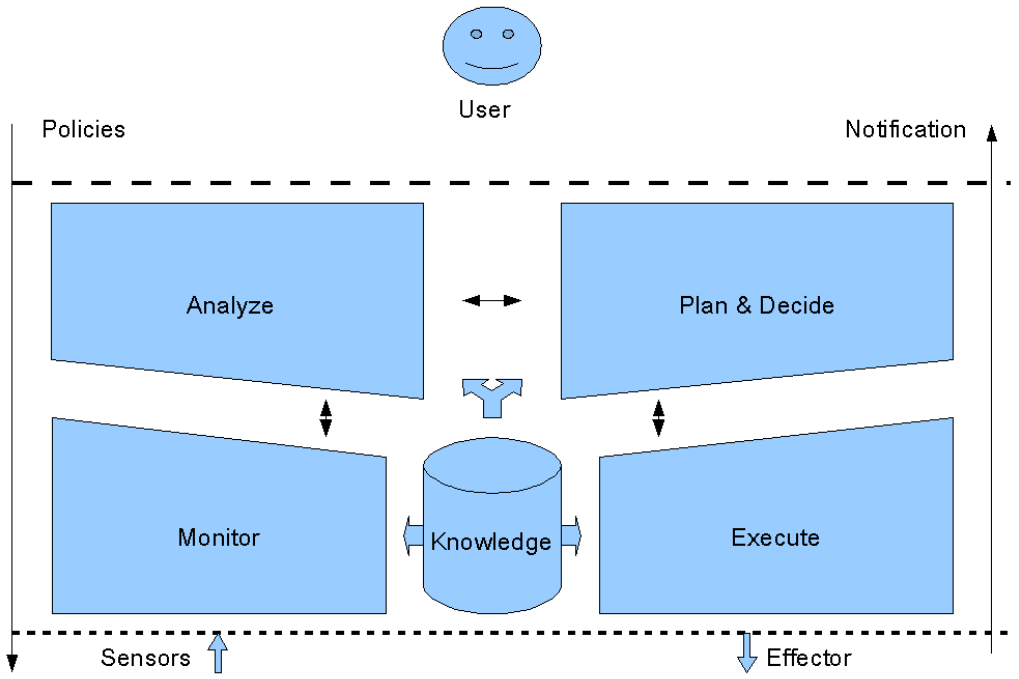


Figure 1.1: Procedure of task

Knowledge is shared in all phases, which means repository of knowledge for computing environment, E.g. compatibility of SW version from SW-repository. The whole task must be followed user's policies and diverse errors must be announced to user.

when a task is triggered, system starts its activities in 1. monitor phase.

In 'monitor phase' all required information will be collected. Information can be obtained easily through a variety of sensors, e.g. temperature and usage of processor. Or these information can be got from user as input parameters.

In 'analyze phase' all collected information will be analyzed and new information will be created from them for next phases.

In 'plan and decide phase' all executions will be planned and decided. The experience of system or user has an effect on decision. This phase is ruled out in my research. Procedure of all activities can be planned in this phase , but this procedure can be shown in other

phases. And the Scheduler of whole task can be planned, but in my research real time factor plays no role. This phase must be considered in implementation of real system, not in virtual simulation.
In 'execute phase' all concrete activities for the task will be executed according to theirs procedure and condition of system. Substantial activities will be executed in this phase.

1.3 Used tools

I used BPMN (Business process management notation) in order to design procedure of task. And I used BizAgi Process Modeler as GUI(graphic user interface) environment tool for BPMN.
For simulation of the tasks I sought for possible tools, but I could not find suitable one for distributed system. For this reason I looked for two separate tools, one for visualization of system, which can show process of tasks, and the other for behind logic of model. Graphviz is used for visualization. And with Visual Basic I have implemented behind logic and GUI of simulator.

1.3.1 BPMN with BizAgi Process Modeler

Usually BPMN is used for business process modeling. You can design a process model with BPMN according to its work flow, and activities can be designed easily with symbols of BPMN, and procedure is represented with arrows. In this model must be predefined start and end point. Because of such characters I used BPMN for designing of tasks for life cycle mangement. And Process Modeler by Bizagi[2] is used as GUI environment tool. It is freeware and can convert created result to many data formats.

1.3.2 Graphviz

Graphviz[3] uses DOT Script language of AT&T, and was developed to draw a diagram. It is an open source program, and has CPL (Common Public License). It can be used as a library by other software. Graphviz can create diverse images from DOT data with its filter. This DOT data can be created and modified with any editor like text data, and you can execute Graphviz's filter in command line or by its GUI.

```
digraph sample {
  1[label="hello",color=red];
  2[label="to",color=blue,shape=box];
  3[label="world",color=yellow,shape=octagon];
  1->2->3;
  }
```

This is a small code for sample result of Graphviz(figure 1.2).

[2] http://http://www.bizagi.com
[3] http://www.graphviz.org/

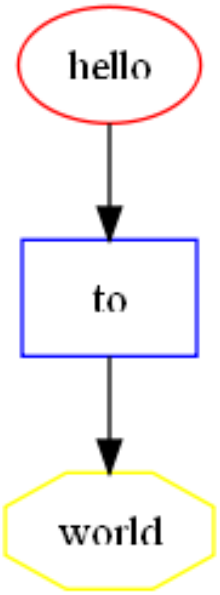


Figure 1.2: sample figure of this code

1.3.3 Visual Basic

Visual basic is a language program for the development of windows application software by Microsoft. With visual basic you can easily develop a program, which has GUI(graphic user interface). To add diverse objects (button and picture box etc.) in a form, and then to code functions according to theirs events, it is basic way to program with visual basic.

I built first a elementary DOT data for Graphviz, which represents default system. This DOT data will be called from simulator before simulation. And it will be modified according to varied triggered tasks. A bitmap image(png format) will be created from this DOT data by Graphviz's filter, which is called from simulator with shell command. This image will be shown on simulator to user. User can see simulation's process with this image. And simulator will show process model with BPMN of a triggered task and also generate description of process at the same time.

2 Static Model

In this chapter I will define the whole system in my research. Hardware components stand by user's place. And system provider supports this system via remote connecting.(cf. figure2.1)

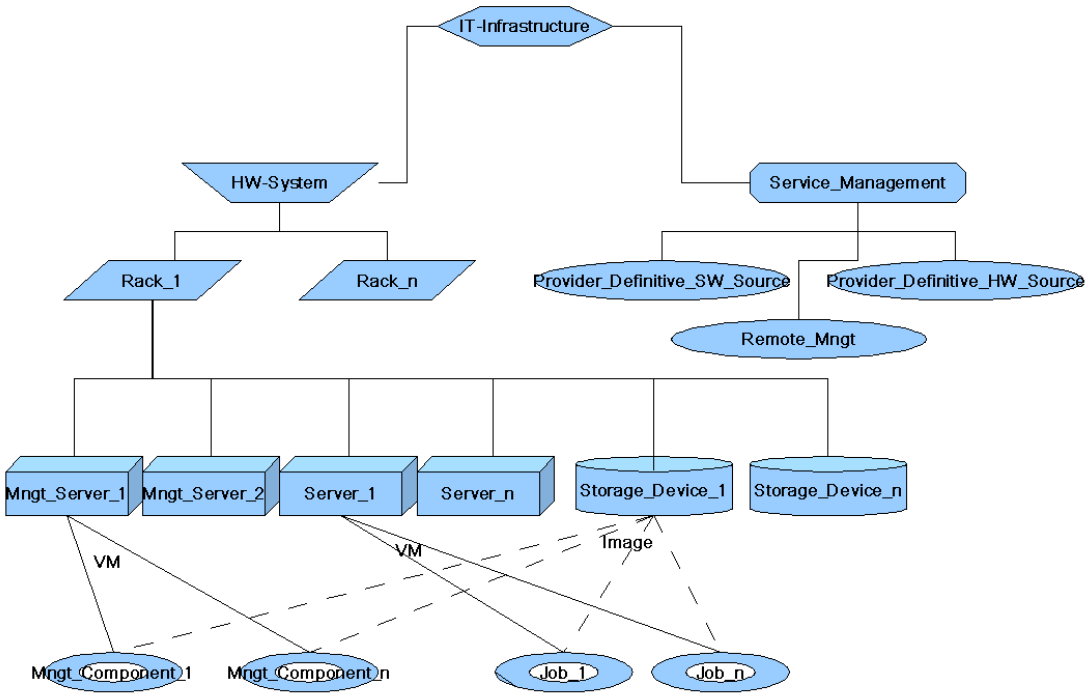


Figure 2.1: Distributed System in IT-Infrastructure

2.1 Hardware component

The system consists of 3 kinds of hardware component, there are server, network and storage device. All this three parts can be placed in a rack. With racks user can easy to place own system. Network device organizes the network between servers, storage devices and extern network, e.g. switch. I defined,that the network of system is built already good. Therefore network is ruled out in this research.

2.1.1 Server /Blade Server

A server has an independent memory for operating system(OS) and virtual machine(VM) host software, and processor unit for work. It supports the Hot-Plug function. That is, user can add this server without to turn whole system off and in the same way by its removal. If a server operates applications for management of system, user calls this server as a management server. And other server, which operates application of user, is called as a worker server.

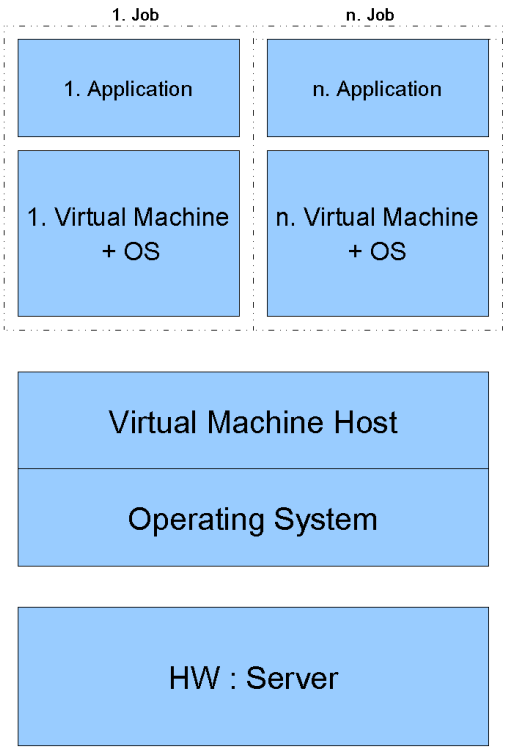


Figure 2.2: Abstraction of server

Like figure 2.2, a server operates applications on each VMs. The user can handle application safely and easily with VM technology, because user can define the boundary of available resource for each VMs. And all data of applications will be saved as VM image. This image is able to clone or to backup, and is reusable.

Management Server

Management servers operate management components. Management component will be explained by storage device. The number of management server is fixed as two. Management server can't be added or removed according to the user's will. If one of them has a problem, the system perceives immediately it, and all management components will be operated on other management server. The two management servers must be always on.

Worker Server

Worker servers operate applications of user, which is called as a job in this research. And it can be added or removed according to the user's will. The number of worker server is not limited in this research.

2.1.2 Storage Device

All VM images of application and additional data are stored in storage device, e.g. NAS (network attached server). Like jobs in figure2.2, operating system SW for VM and application SW are together in a VM image. I have supposed in this research, that the capacity of storage device is unlimited. Therefore unlimited many images can be stored. There are two kinds of application. One is for management and the other of user.

Management Component

Management component means an application for system management,e.g. grid controller. Therefore management component must be handled carefully. Before its removal it must be checked, whether it is still required for any device. For reason of a fixed number of management server, the number of running management components can also be limited.

Job

Job means an application of user, e.g. database host or client program. It can be added or removed according to user's will. Its Addition depends on free HW resource. If free HW resource by worker servers is existent, new job can be run.

2.2 Service management by system provider

The system provider must support many services for the user. This service point is unique in the world. Like figure2.1, it provides definitive hardware and software source of user's platform. And the provider can remote control user's system. System provider can offer more services according to diverse user's needs.

3 Dynamic Model

I will deal with tasks for life cycle management in this chapter. These tasks can be categorized according to managed objects. First I categorized managed objects. Work pool consists of worker servers and jobs. And management pool consists of two management servers and management components.
Basic tasks for worker server are addition, removal, power management, and change VM host version of server. Basic tasks for job are addition, removal, redistribution and change version of job. Additionally an auxiliary task is designed for other tasks.
And basic tasks for management pool are addition, removal and change version of management component and change VM host version of management server.

I define now all environments of research system.
All servers, jobs and management components get an identification number(ID). This number will be distributed only one-time, because it is to eliminate overlapping in log data.
All servers are of same type and they have same capacity : 100. In other words, theirs free resource is at first 100. And all jobs and all management components have demand factor between 1 and 100. Demand factor of application depends on its maximum work load. It must be first measured and proffered. To operate one job is necessary one server, and free resource of server will be reduced about its demand factor after start of the job. The minimum free resource of server is 0. Many jobs can be operated on one server, until its free resource is 0.

All tasks are designed with BPMN. I will explain procedures of tasks with figures, which are created by BizAgi process modeler.

Used Variables

Status: power status of server
freeRes: free resource of server
demFact: demand factor of management component or job
JROS: Association of job with server. Job runs on server, e.g. JROS(I)=J : Job I runs on server J

3.1 Auxiliary task

Find Server For Job

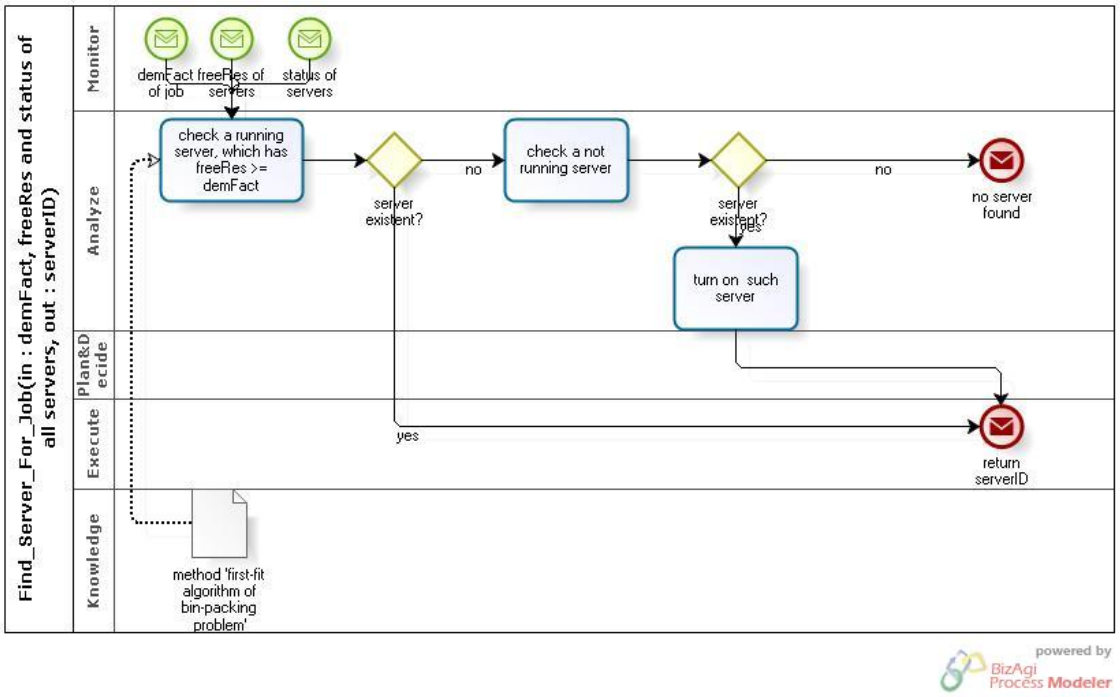


Figure 3.1: Task : Find For Server Job

First I have designed an auxiliary task. This task does actually only analyzing, that the system finds a suitable server for a job. This auxiliary task can be triggered in many other tasks. For trigger of this task three input parameters are required. There are demand factor of job,which will operated, free resource status and power status of worker server. The system finds with them a suitable server for such job. The first fit algorithm of bin-packing problem is used in this auxiliary task . This algorithm is in the literal sense of the word, that first server is searched first, whether this server is fit for such job.
If a running server for such job is existent, system will return such server's ID. Or system finds next a server, which is off. And if it is existent, it will be turned on and task returns such server's ID. Else task returns a message: "no server found".

3.2 Work Pool

3.2.1 Worker Server

Add Server

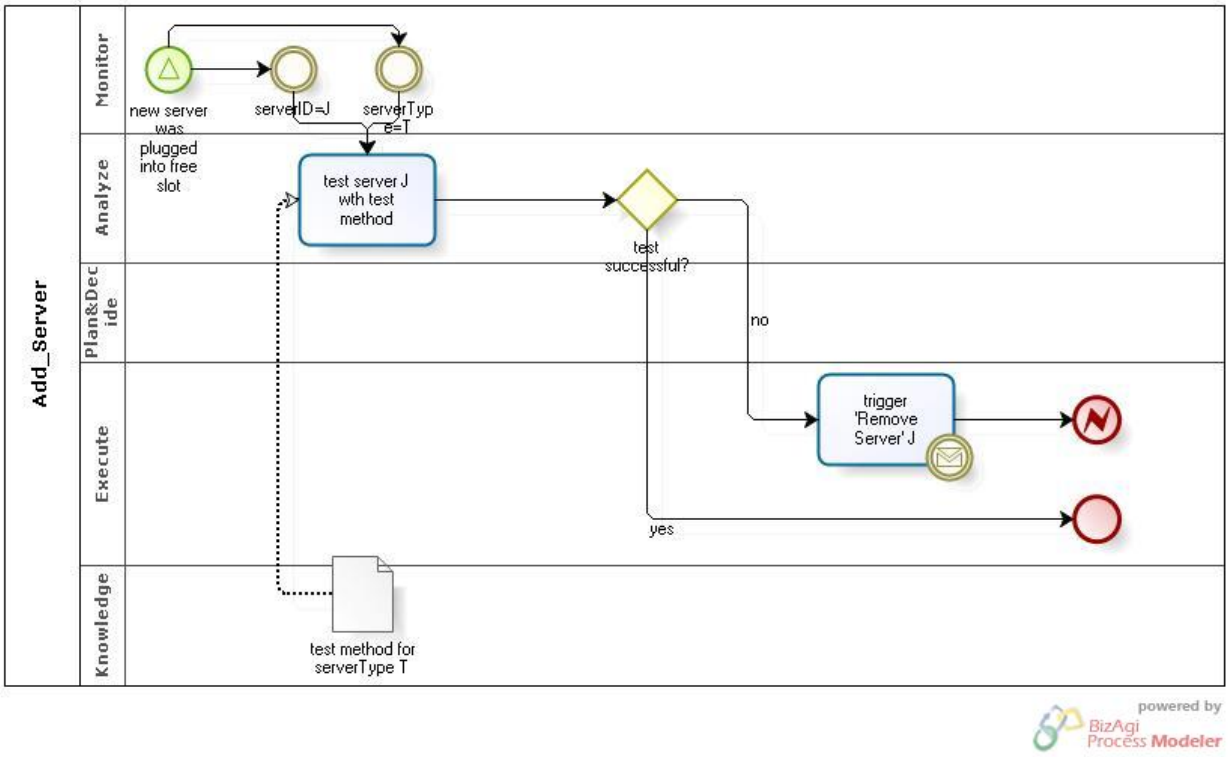


Figure 3.2: Task : Add Server

When the user has plugged a new server into a free slot, this task will be triggered. The new server will get automatically an ID from system. And the system will perceive type of new server from the sensor, then it gets a suited test method from the method repository. At this time the new server will be tested with its test method. If test was successful, server will be ready for jobs. Or if test was not successful, task 'Remove server' for new server will be triggered and system will result error event .

Remove Server

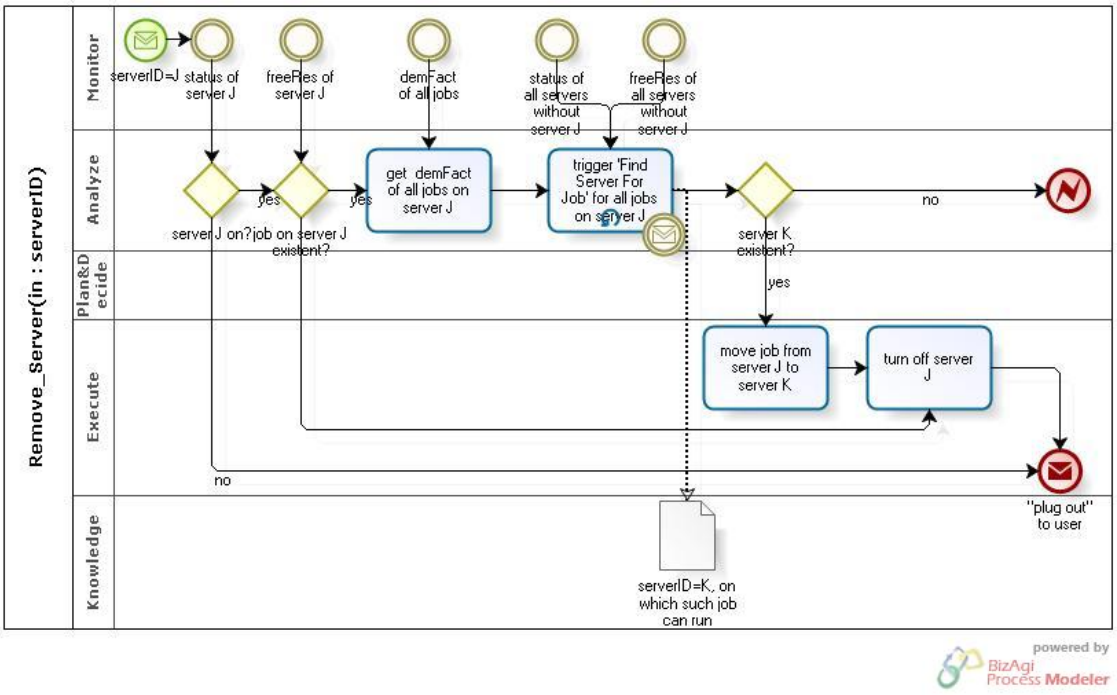


Figure 3.3: Task : Remove Server

User can plug out a server after task 'remove server', which has problems or is not required more. In order to trigger of this task, server's ID is necessary as an input parameter. When this task is triggered with such server's ID, system will get its status from sensor and checks, whether it is still running. If it is not running, system will notify message to user that such server can be plugged out.
Or if it is still running, the system will get its free source status, and check existence of any jobs on this server. And if it has no jobs, it will be turned off and system will notify to user the same message. Or if it has jobs, the auxiliary task 'Find Server for Job' will be triggered with demand factor of all its jobs, power and free resource status of other servers in order to move its jobs to other servers. The result of auxiliary task will be remembered, and can be used continually. With this result all jobs on such server move to other servers. All its jobs must be moved to other servers, otherwise it can't be turned off or removed. If all its jobs are moved it will be turned off. And system will notify to user the same message. If system have got the message 'no found server' after the auxiliary task, the system will return an error event.

Power Management

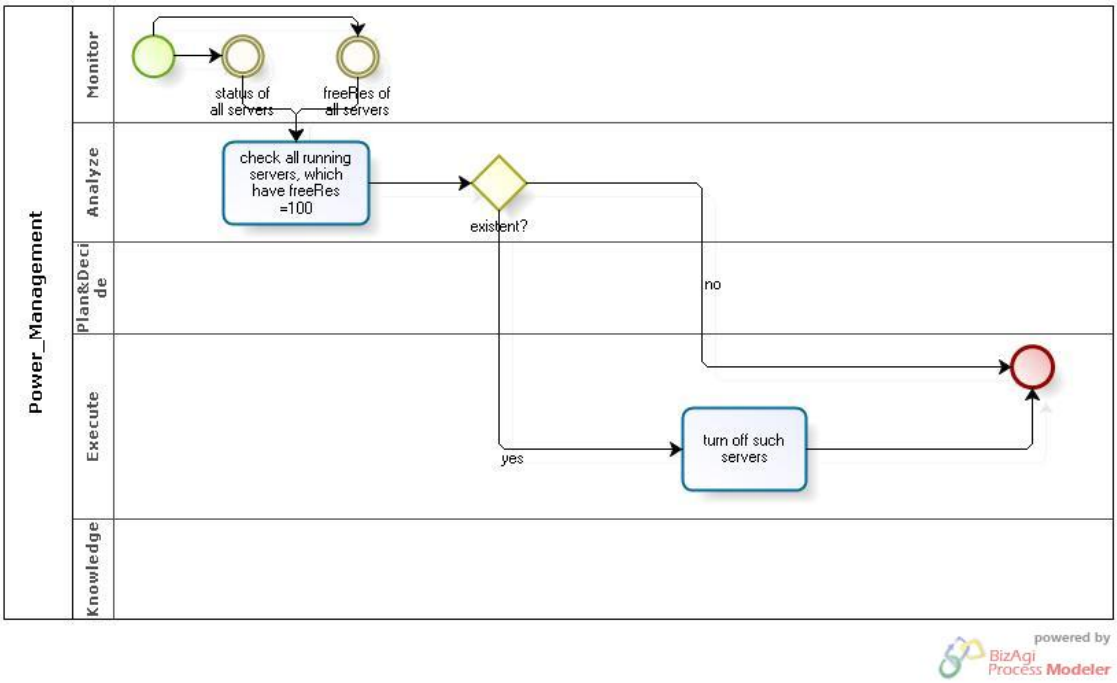


Figure 3.4: Task : Power Management

This task turns off all running servers, which have no jobs. User can save the charge of electricity with it. When this task is triggered, the system will get power and free resource status of all servers from sensors, and it will find all running servers, which have free resource 100. Then they will be turned off, and task will be finished. If such servers are not existent, task can also be finished without an error event. This task is not mandatory. According to user's will or in other tasks it can be triggered.

Change VM host Version on Worker Server

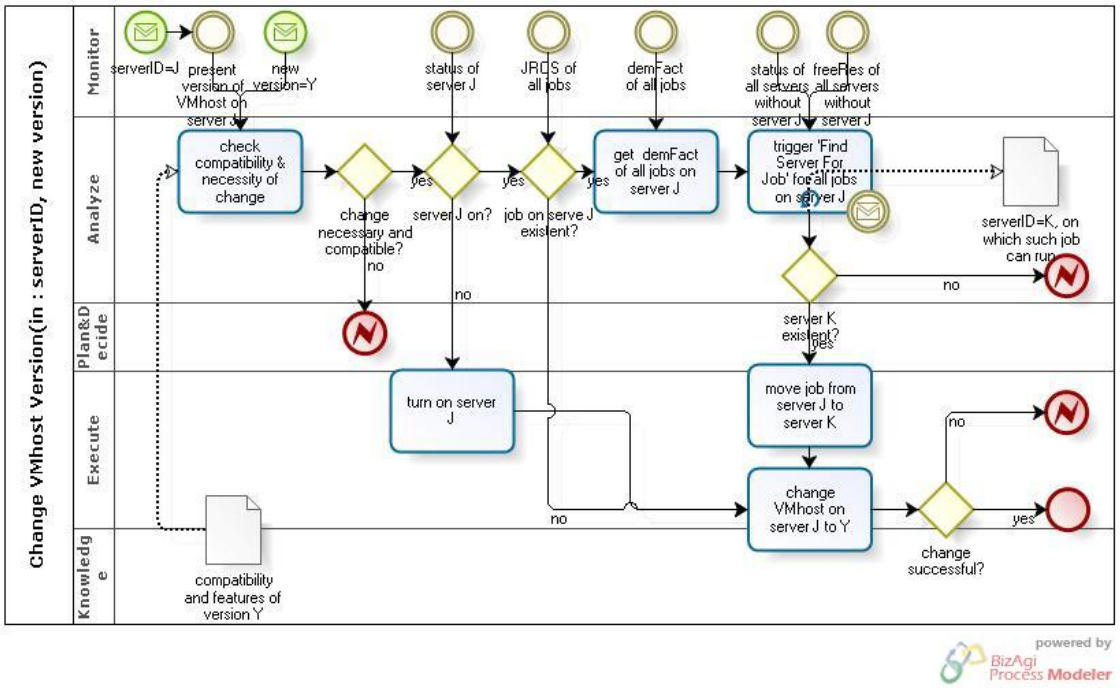


Figure 3.5: Task : Change VM host Version

If a new update version is available or feature of old version is required, user can change present version. This Task changes VM host version of worker server. In order to trigger of this task server's ID, on which VM host version will be changed, and a new version, to which VM host version will be changed, are necessary. When this task is triggered, the system will get present version of VM host with server's ID from sensor. And it will get compatibility and features of new version from compatible SW version repository, and check compatibility and necessity of new version. If change is not necessary, the system will return an error event. Or all jobs on such server must be moved before its VM host version is changed, because change of VM host has instable effects on its jobs. If change is necessary and the server is off, it will be turned on. And its VM host will be changed to new version. If change is necessary and the server has no job and is running, its jobs will be moved at the same way of task 'Server Remove'. If it has still jobs, change will be canceled and an error event will be return. Otherwise its VM host version will be changed. If change was not successful, system will return an error event. Or the server will be ready for job.

3.2.2 Job

Add Job

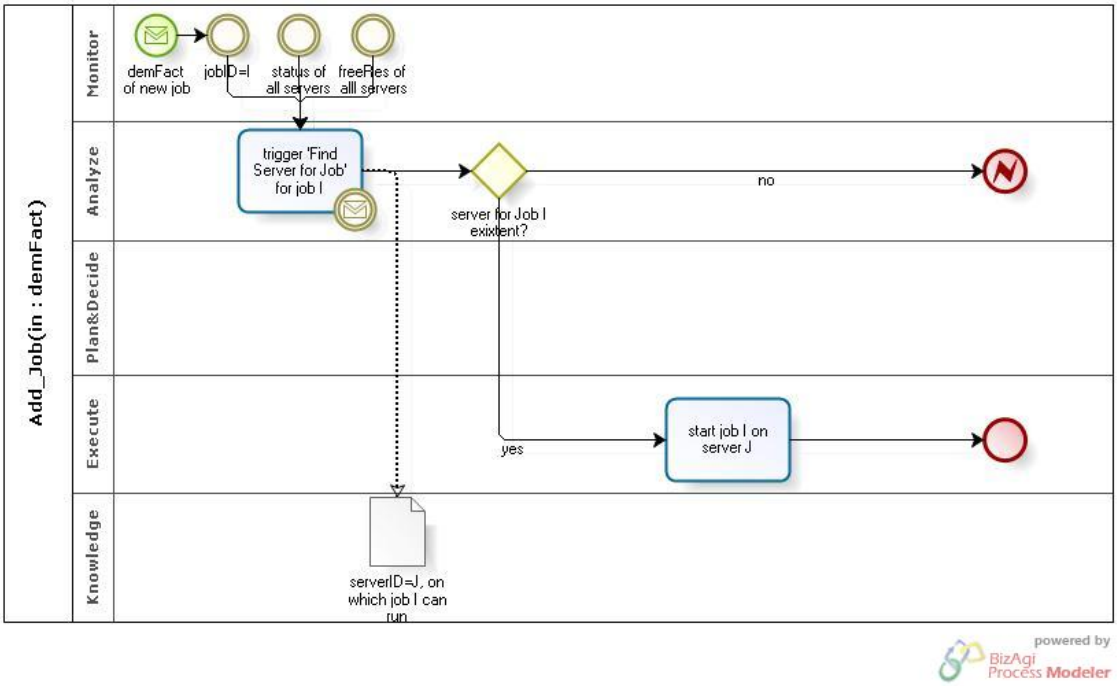


Figure 3.6: Task : Add Job

According to user's will or need, a new job can be added. If the user wants to start a new job, he should know first, how much resource is necessary in order to start it. Demand factor of new job is necessary as an input parameter for this task. This task distributes and starts job, which is stored already in storage device. When this task is triggered, this job will get automatically an ID. Afterward the system will get power and free resource status from all servers in order to trigger the auxiliary task. Server's ID will be got from the auxiliary task, on which new jobs can run. If this auxiliary task returns server's ID as a result, new jobs can start on such server and task can be finished. Otherwise system will return an error event.

Remove Job

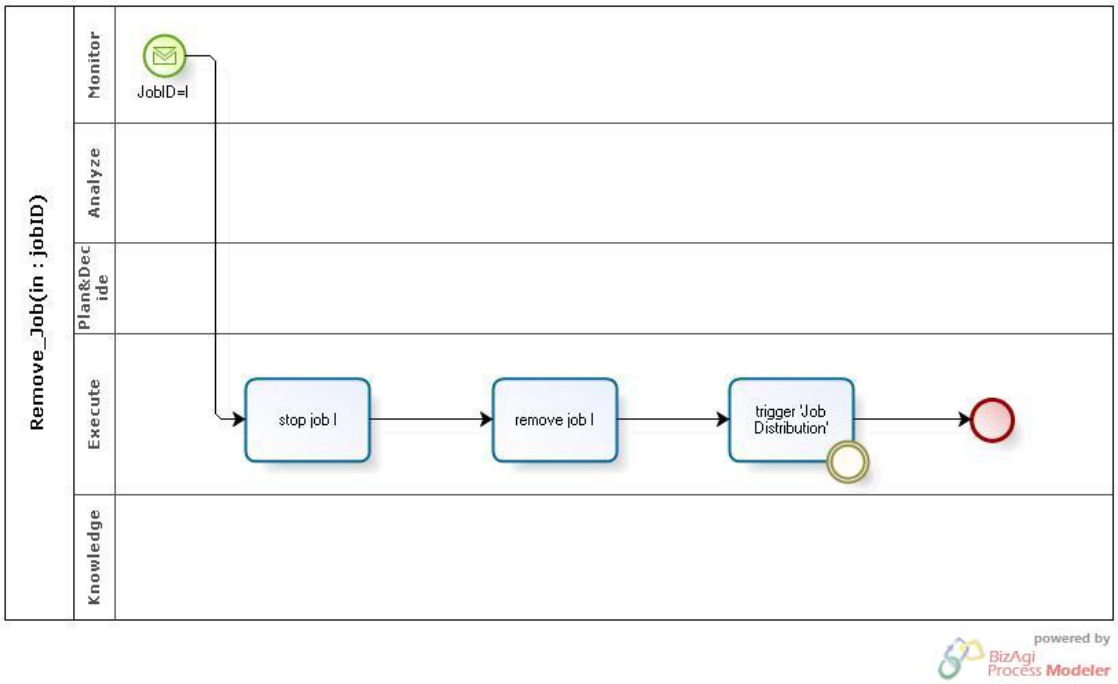


Figure 3.7: Task : Remove Job

User can remove jobs if they are completed or not necessary more. In order to trigger of task 'Remove Job' the job's ID is necessary, which will be removed. When this task is triggered, the job will be stopped, before it will be removed from system. After Removal of the job, task 'Job Distribution' will be triggered in order to use resource efficiently.

Job Distribution

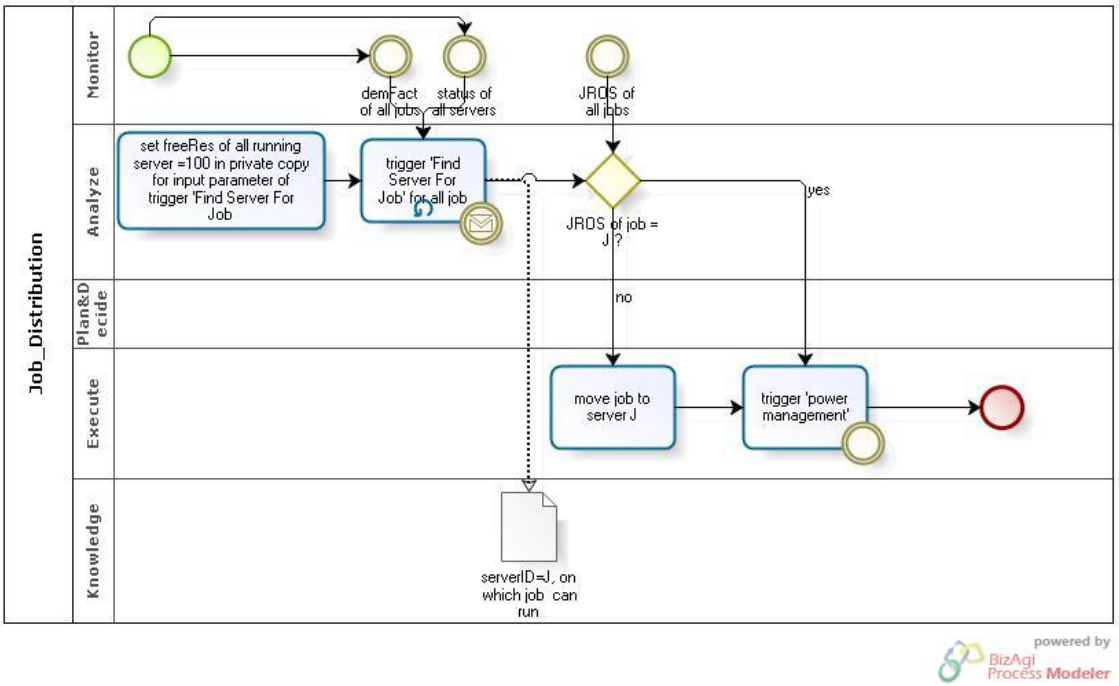


Figure 3.8: Task : Job Distribution

Task ‘Job Distribution’ redistributes all jobs for efficient using of resource, and triggers task ‘power management’. When this task is triggered, system will get demand factor of all jobs and power status of all servers from sensor. And additionally it will create new free resource parameter of all running servers as 100. These three parameters are required in order to trigger the auxiliary task. For all jobs the auxiliary task will be triggered and system will compare server’s ID, which is result of auxiliary task, with present host server’s ID of this job. If they are different, this job will be moved to other server, which is result of auxiliary task. Or job stays on present host server. After auxiliary task of all jobs task ‘power management’ will be triggered in order to turn off free servers. This trigger can be triggered by user or also in other tasks.

Change Job Version

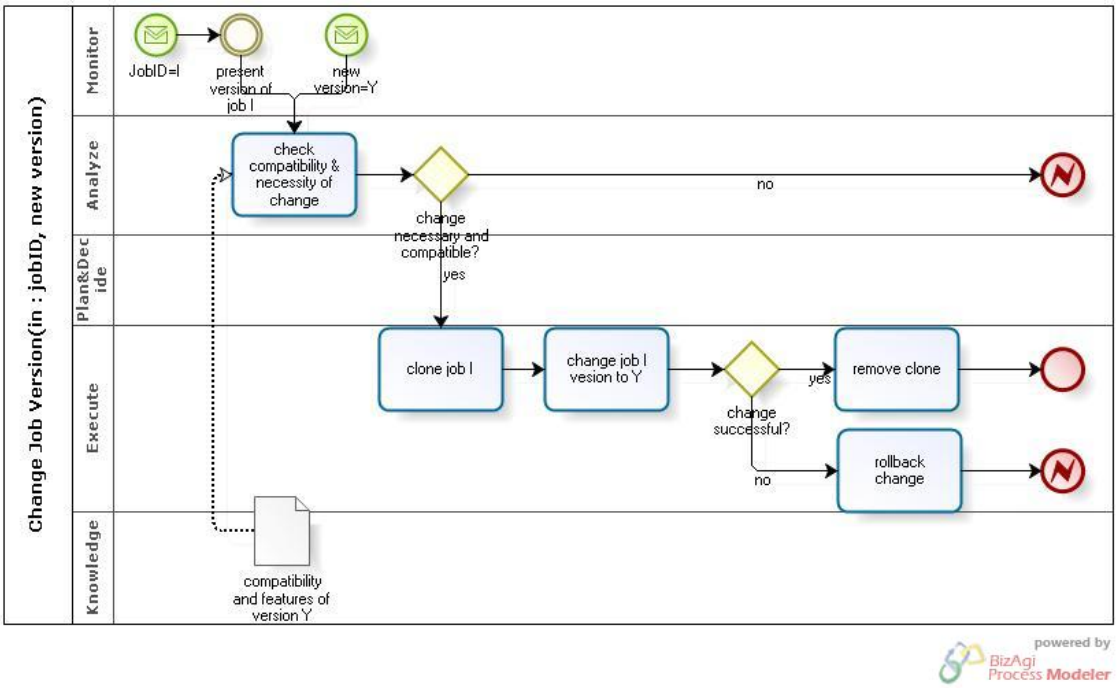


Figure 3.9: Task : Change Job Version

If user or system wants to change present job version into another version, then he or it can trigger this task with new other version and job's ID, which will be changed. When this task is triggered, system will present job version with job's ID from sensor. And it will get compatibility and features of new version from compatible SW version repository, and check compatibility and necessity of new version. If change is not necessary, system will return an error event. Or if change is necessary, system will clone this job and change job version. Unlike VM host, Job is operated on virtual machine. Therefore job version can be changed safely with its clone. If change was not successful, job can rollback with its clone and system will return an error event. Or its clone will be removed.

3.3 Management Pool

Add Management Component

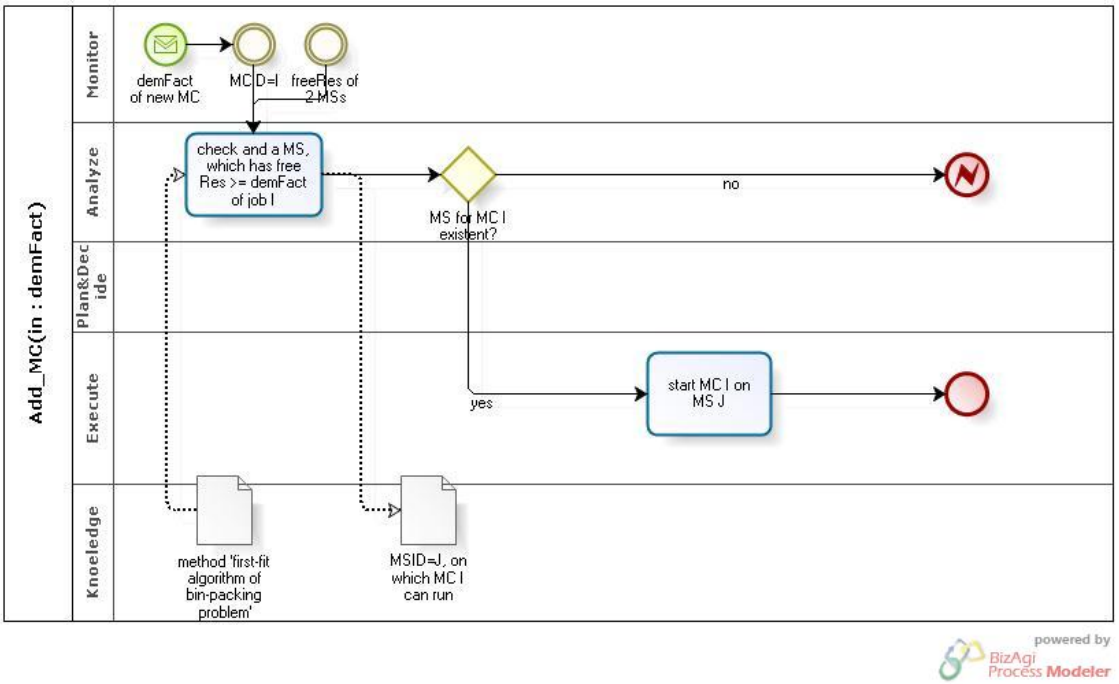


Figure 3.10: Task : Add MC

If the user has installed a new device, he may install also new management component for it. This task adds new management component to the management pool. Management component is operated on virtual machine like job, however the number of management server is fixed as two, and they are always running. Then such auxiliary task 'Find Server for Job' is not necessary. When this task is triggered with demand factor of new management component, this management component will get automatically an ID. And the system will get free resource status of management servers from sensor. System will find a management server for new management component. If free management server is existent, this management component will be started on such management server. Or system will return an error event.

Remove Management Component

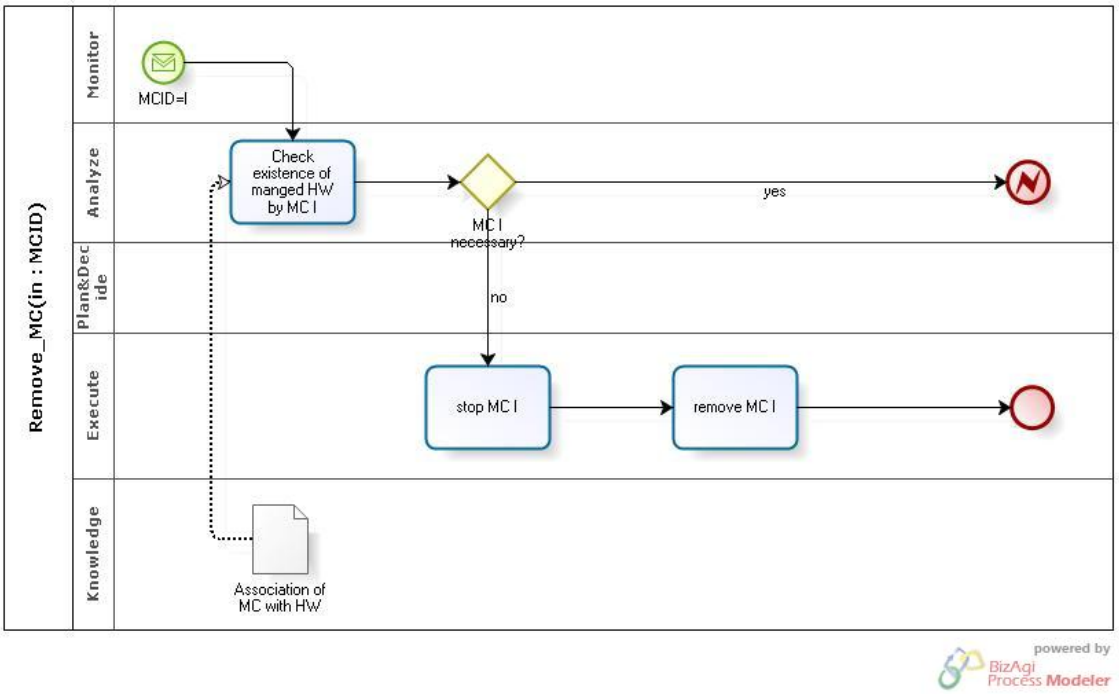


Figure 3.11: Task : Remove MC

Management component can be removed according to user's will, after the device is removed, which is controlled by this management component. When this task 'Remove Management Component' is triggered with management component's ID, which will be removed, system will check, whether this management component is still required. If it is still required, management component will not be removed, and system will return an error event. Or system will stop it and remove it from system.

Change Management Component Version

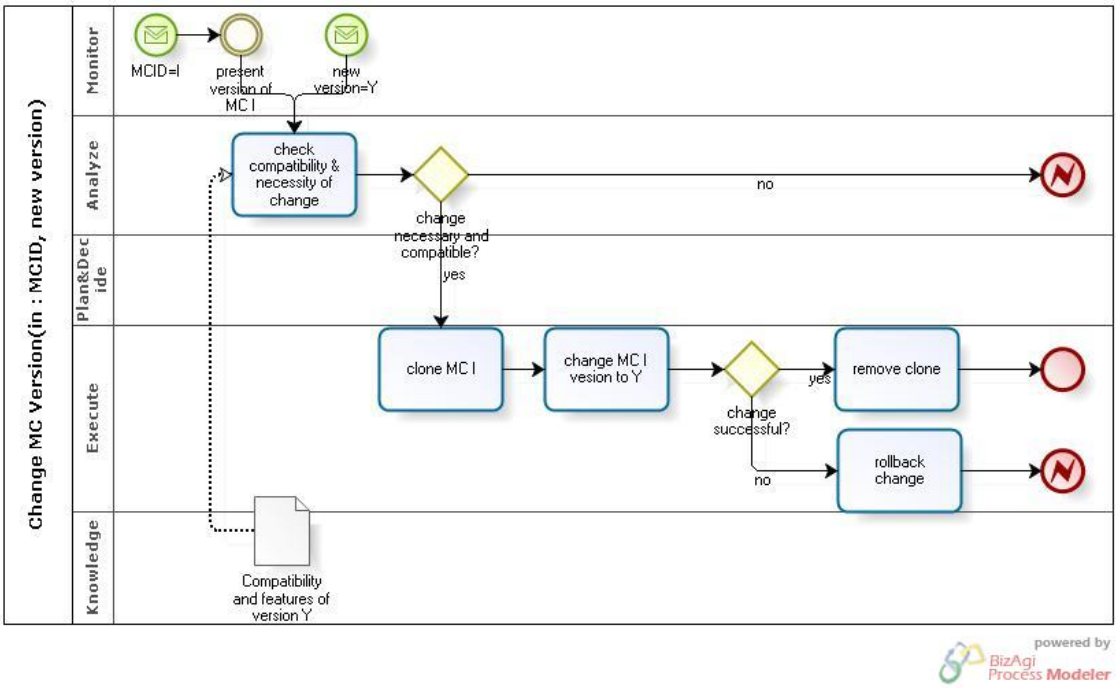


Figure 3.12: Task : Change MC Version

This Task changes the management component version. In order to trigger this task, management component's ID and a new version are necessary. And it follows the same procedure like task 'Change Job Version'.

Change VM Host Version on Management Server

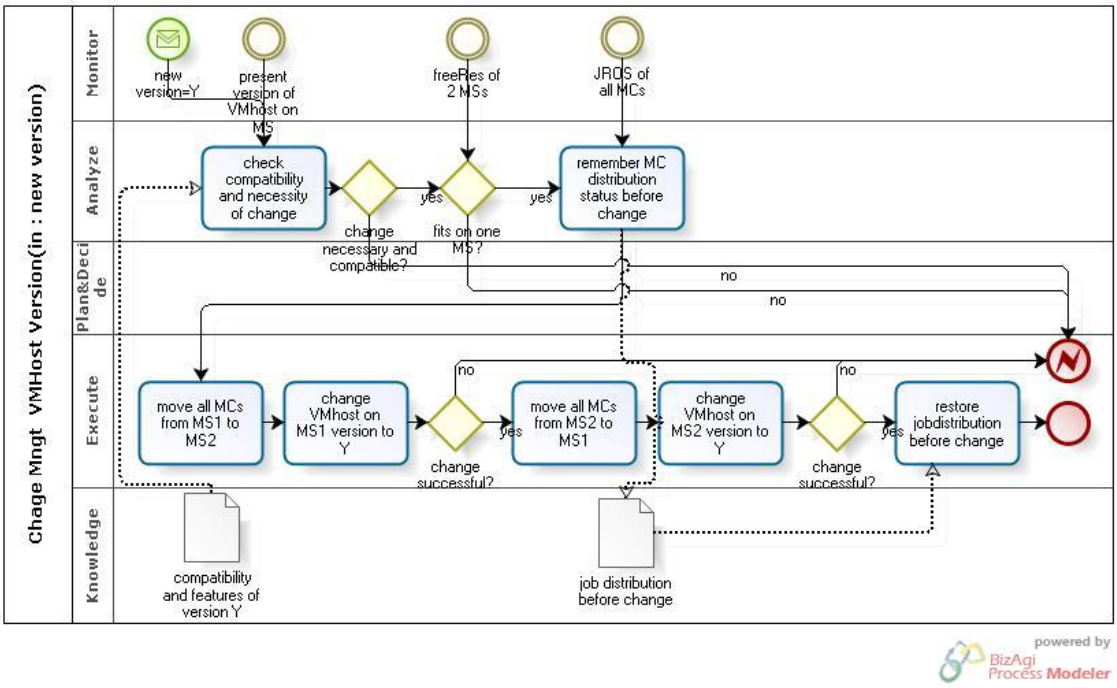


Figure 3.13: Task : Change Mngt VMHost Version

Two management servers should have always the same VM Host, because the system can move management component to the other management server, if one management server has a problem. When this task is triggered, system will check compatibility and necessity of new version in the same way of task 'Change VM Host Version' on worker server. If change is not necessary, system will return an error event. Or system will check, whether all management components can operate on one management server. If it is possible system will remember status of distribution of management components and start rolling change. All management components on 1.management server will be moved to 2.management server, and VM host version of 1.management server will be change to new version. If this change is successful, VM host version of 2.management server will be changed in the same way. Or change will be stopped and system will return an error event. After successful change, management components, which were on 2.management server before change, will be moved from 1.management to 2.management. And 2 management servers will have same VM host.

4 Conclusion

In this chapter I will deal with some problems, which I was confronted with during my whole research.

Necessity of efficient algorithm for distribution of jobs

In order to find a server for a job or to redistribute jobs an efficient algorithm is necessary. The best fit algorithm can be a possible variant algorithm. It is efficient for addition of a new job, but not for redistribution of all jobs. System must avoid possibly frequent move of jobs for stable system. But redistribution with existing bin-packing algorithms can occur frequent move of jobs. By Implementation of real system other algorithm must be found for stable system, and for efficient using of servers.

Synchronization with real time

Real time factor is not dealt in my simulation. However this factor is also important for the real system, e.g. acceptable time for change of job version: When job has low work load, change of version can be started, because possibly little difference between clone and original job must be existent. This time can be gained from experience or record of system.
And time factor play an important role for scheduler in plan and decide phase.

These points are considered by implementation of real system. And naturally user can manage system better with more tasks besides basic tasks. I have modeled and simulated only basic tasks. With them you can understand these tasks of life cycle management in distributed system, and develop more.

5 Appendix

In this chapter I will describe about my simulator. At first I made a manual for my simulator, and a demo-scenario for simulation. And I will show results of single step in order to understand, how Simulator works.

5.1 Manual of Simulator

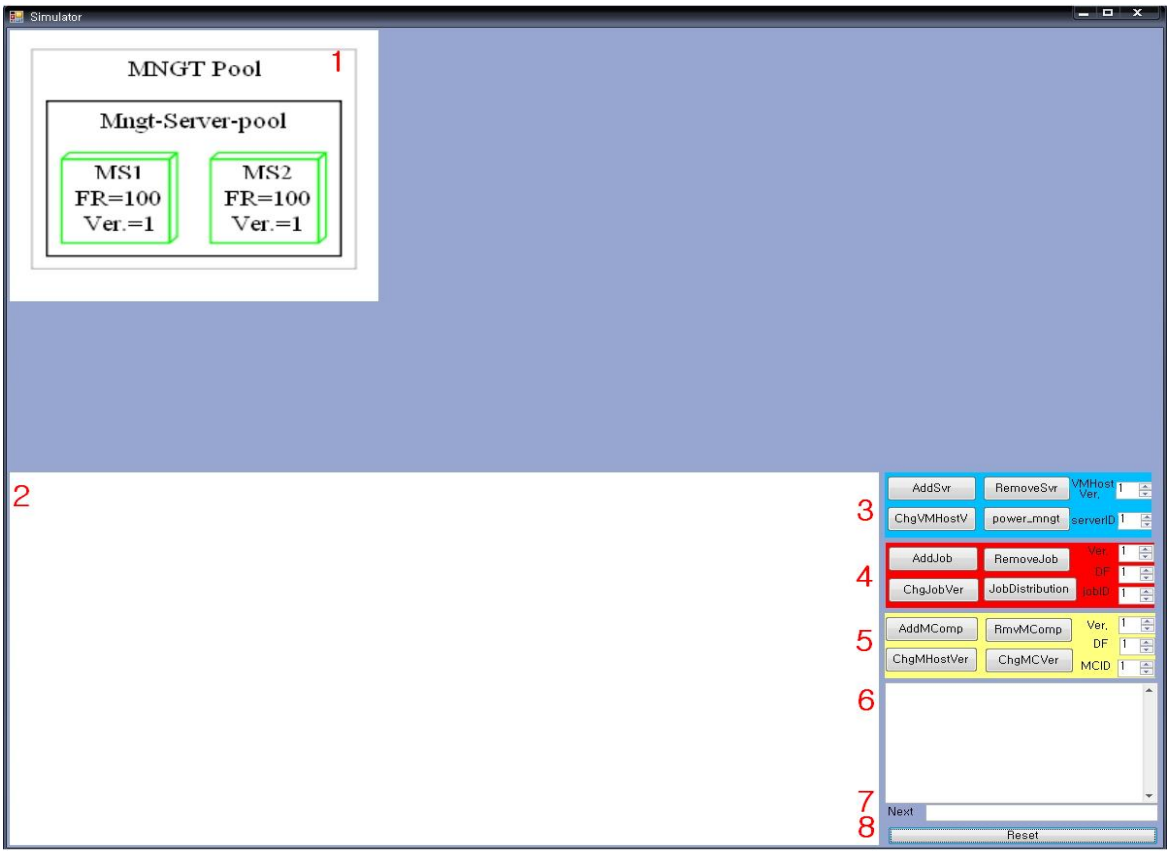


Figure 5.1: GUI of Simulator

Window 1 visualizes abstract figure of the research system, and it consists at first of two running management server. It can represent also management components, job and worker server. Arrows represent associations of jobs with servers. Window 2 shows the process model of task, which is triggered just now. For process model of subtasks new windows will be opened, which are triggered in main task, e.g. task 'Find Server For Job', task 'Job Distribution' and task 'power management'. User can manage the worker servers with

control panel 3. It has 4 buttons for addition, removal, change of VM host version and power management for servers. And user can enter also 2 values each for VM host version and ID of servers. User can manage the jobs with Control panel 4. It has 4 buttons for addition, removal, change of version, and redistribution for jobs. And user can enter also 3 values each for version, demand factor and ID of jobs. User can manage the management pool with Control panel 5. It has 4 buttons for addition, removal and change of version for management components, and change VM host version of two management servers. And user can enter also 3 values each for version of management components and VM host for management servers, demand factor and ID of management components. In the text box 6 all activities of the triggered task are described. In the text box 7 next activity of the task is described, and user must click more same task button, until nothing in this box is existent. User can reset simulator with button 8(reset).

5.2 Demo of Simulation

Add 4 Servers

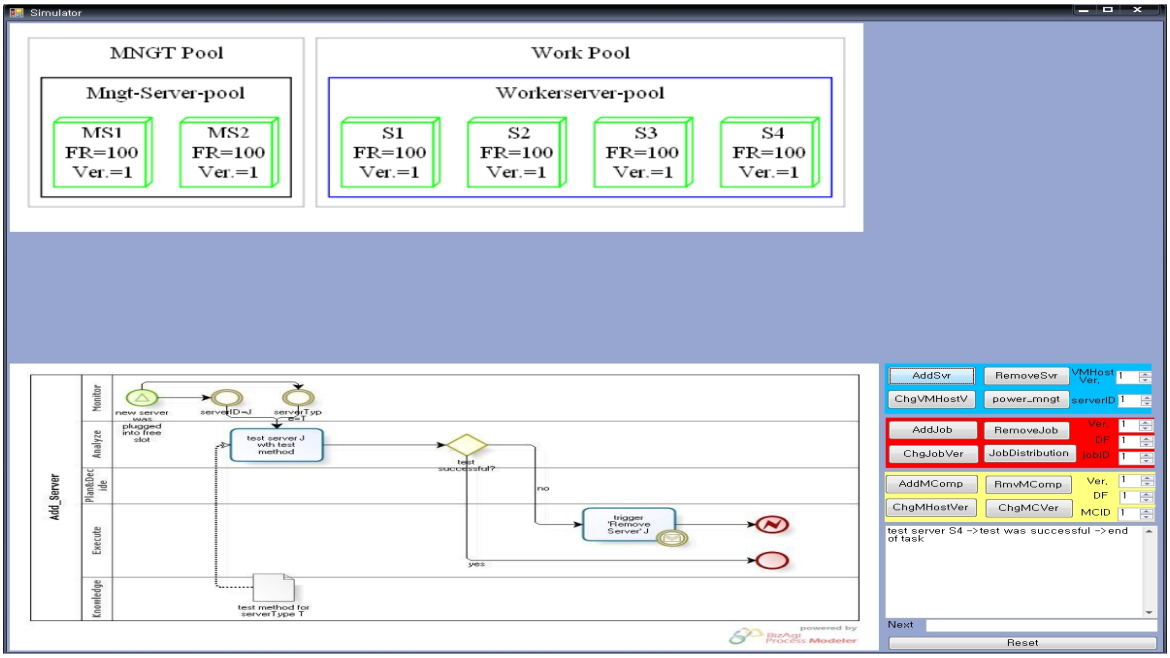


Figure 5.2: 4 servers are added.

Add 3 Management Components (demand factor=30, 40, 50)

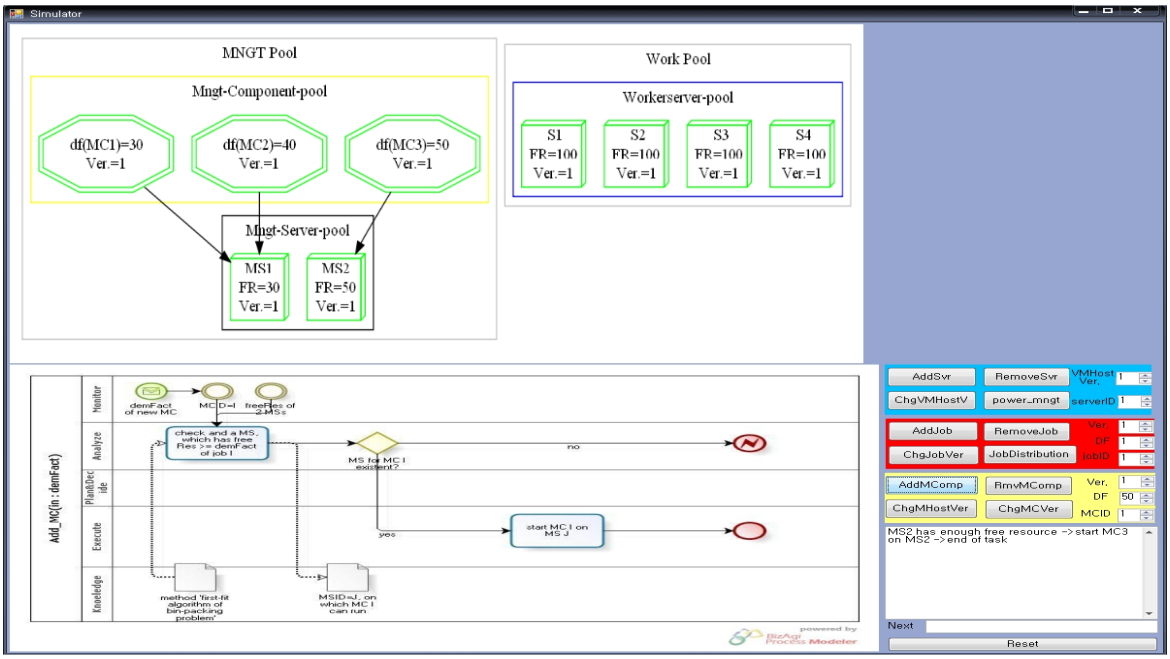


Figure 5.3: 3 management components are added.

Add 4 Jobs (demand factor=70, 40, 30, 80)

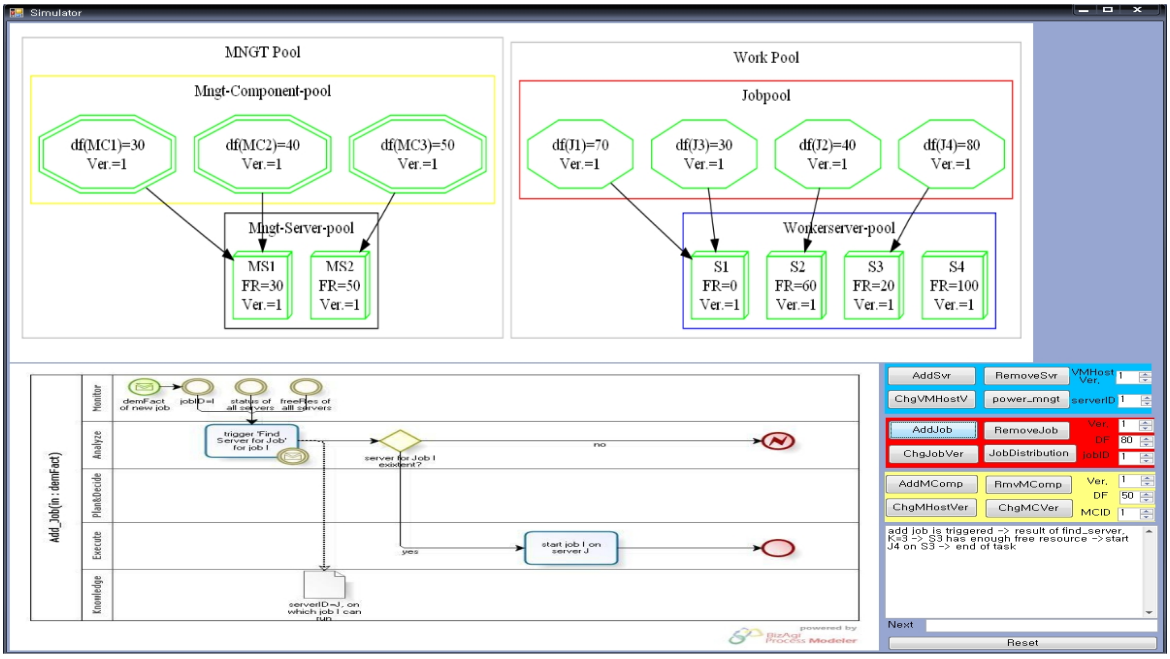


Figure 5.4: 4 jobs are added.

Change job version of J1 to 2

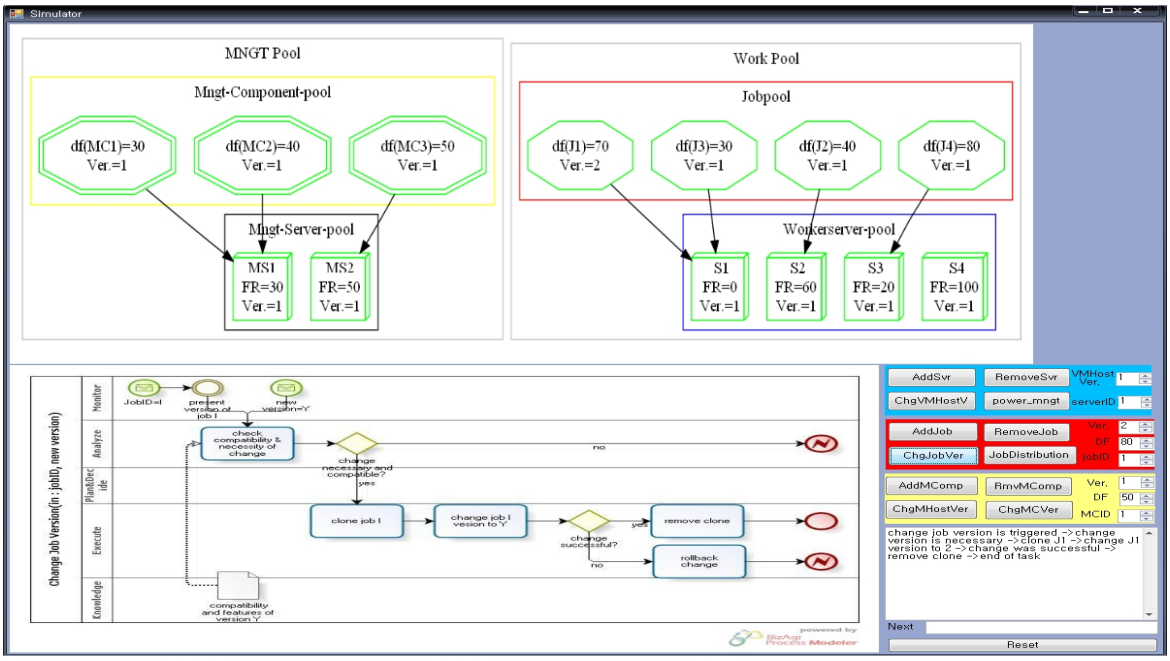


Figure 5.5: Version of J1 is changed to 2.

Remove J1 (After removal 'Job Distribution' is also triggered)

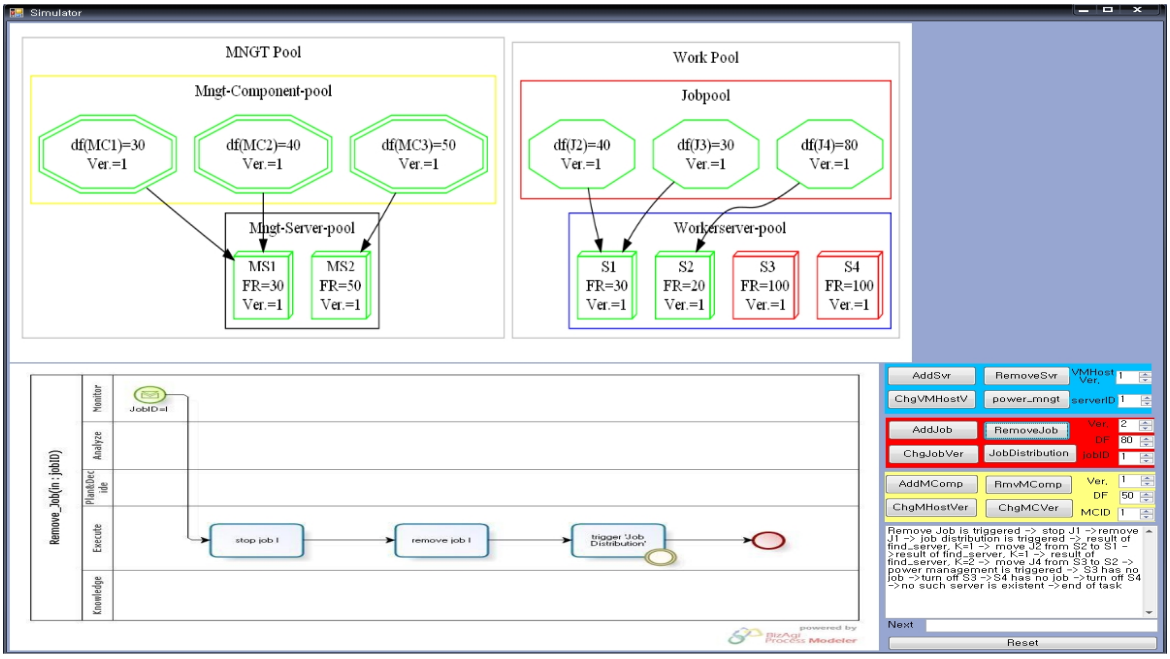


Figure 5.6: J1 is removed, and all jobs is redistributed, and free server S3 and S4 are turned off.

Remove MC2

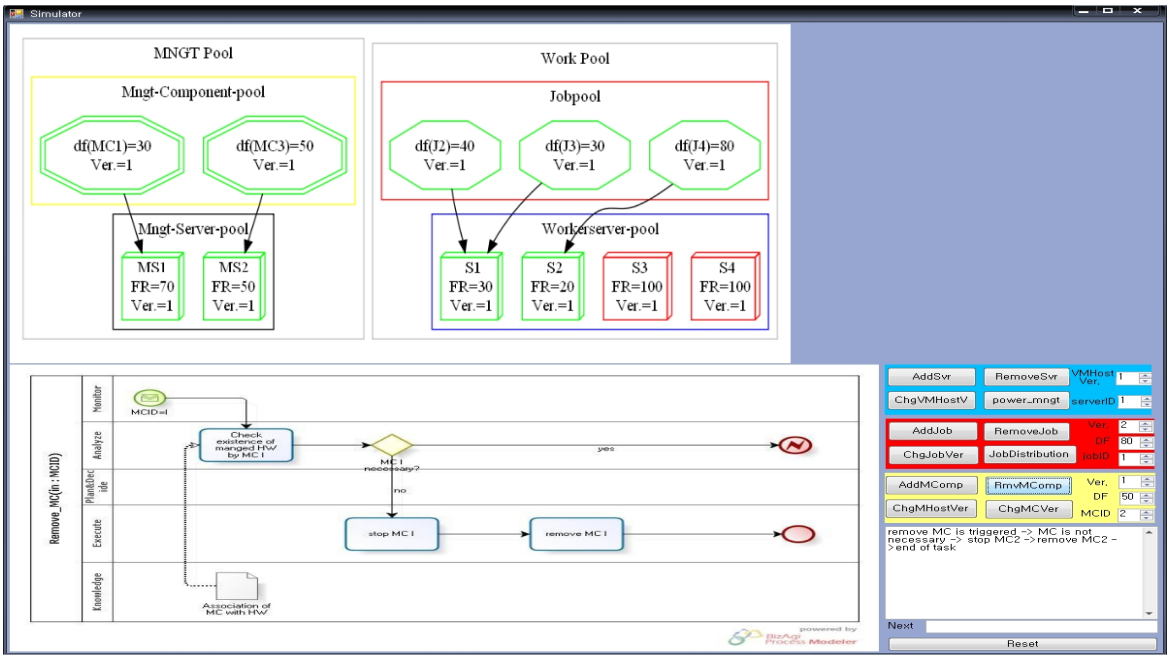


Figure 5.7: MC2 is removed.

Change VM host version of S1 to 2

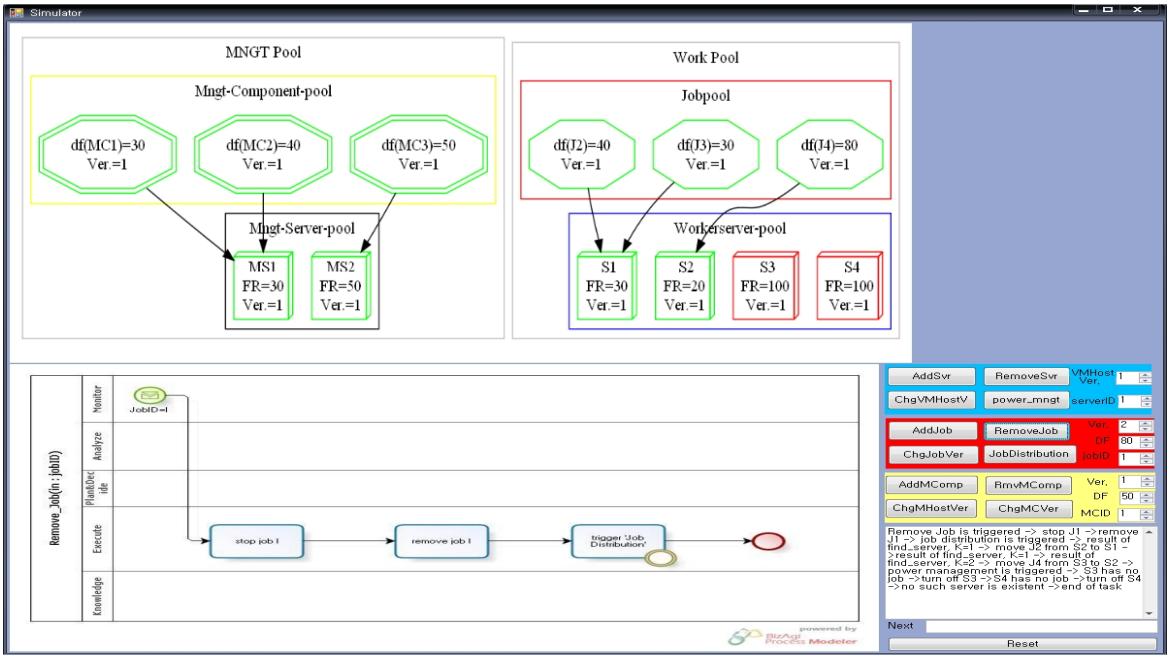


Figure 5.8: All jobs on S1 are redistributed and VM host version of S1 is changed to 2.

Remove S2

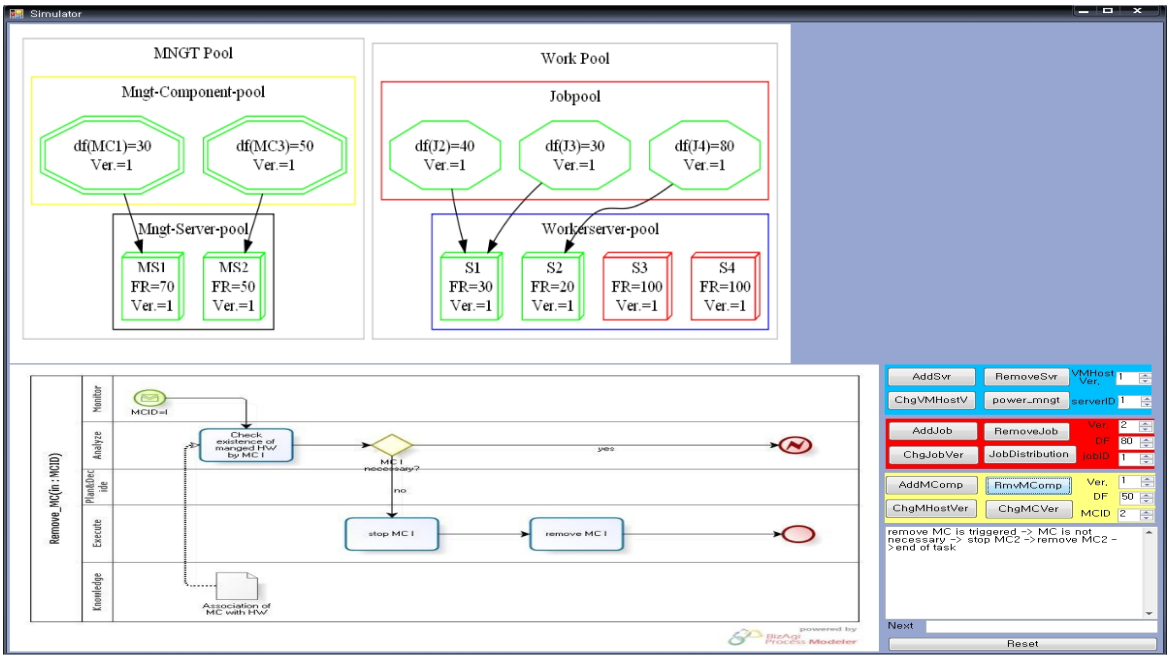


Figure 5.9: All jobs on S2 are redistributed and S2 is removed.

1 : Change VM host version of MS to 2 (rolling update)

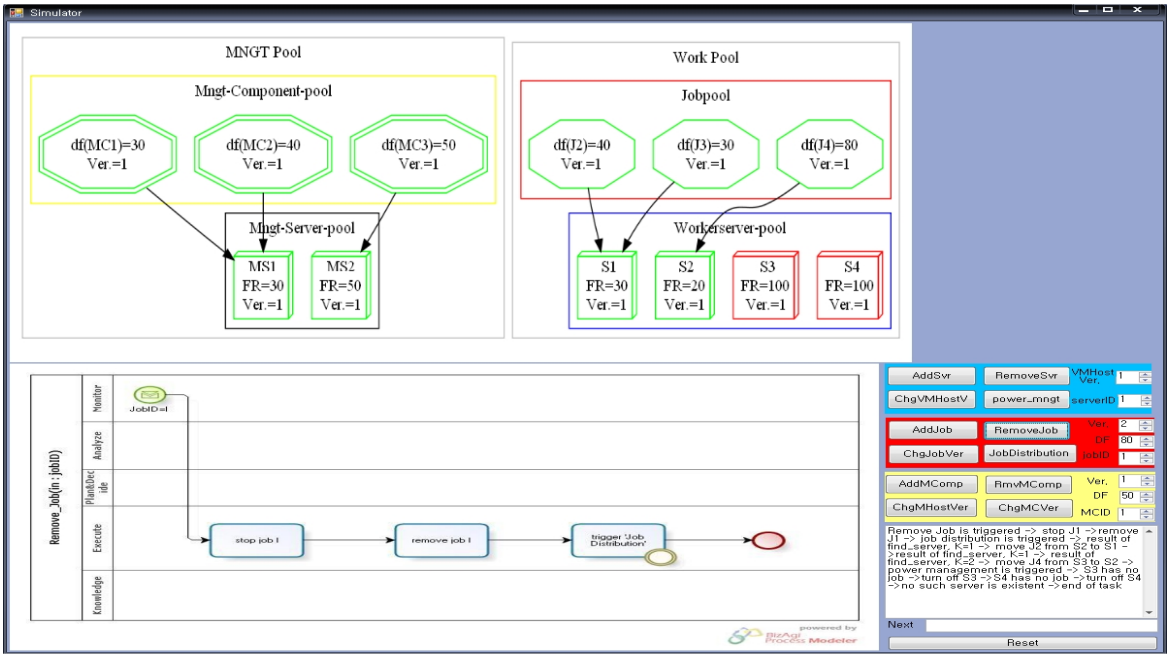


Figure 5.10: All MCs on MS1 are moved to MS2 and VM host version of MS1 is changed to 2.

2 : Change VM host version of MS to 2 (rolling update)

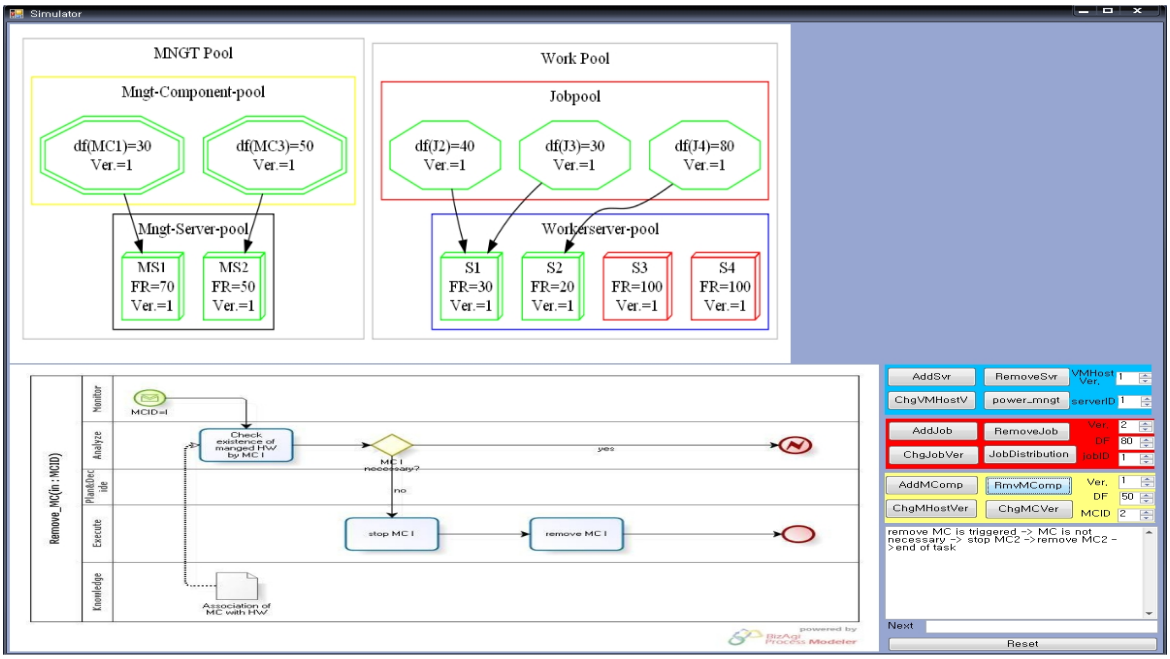


Figure 5.11: All MCs on MS2 are moved to MS1 and VM host version of MS2 is changed to 2.

3 : Change VM host version of MS to 2 (rolling update)

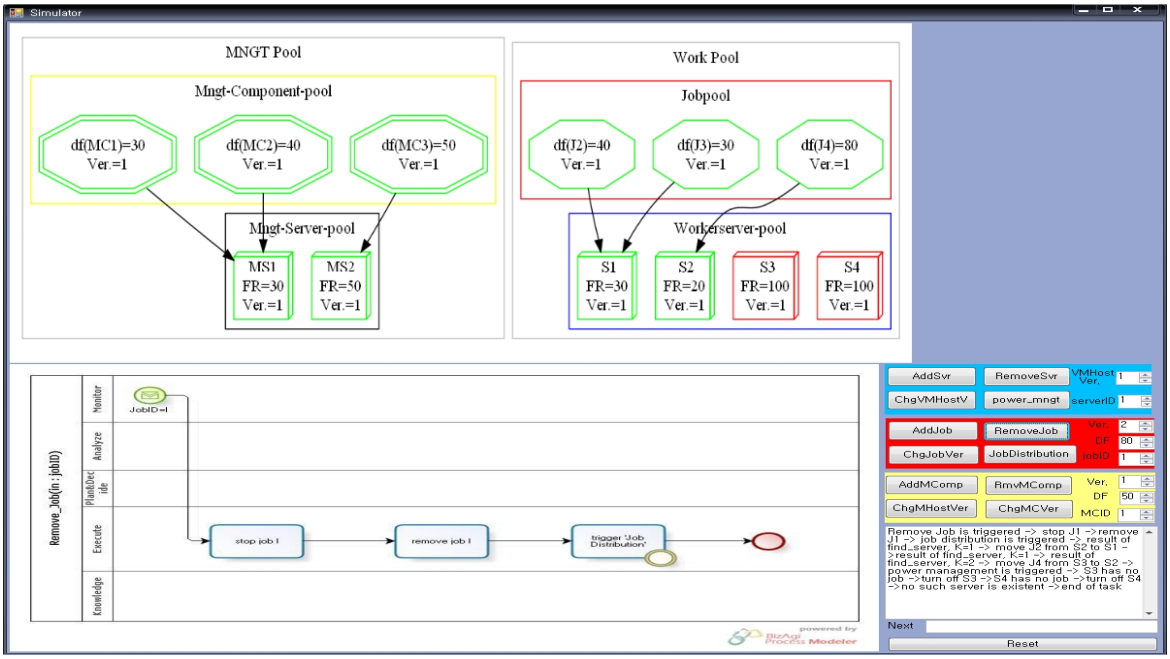


Figure 5.12: last distribution status of MCs is restored.