High Level Design Document
Bin-packing VM Consolidation Algorithm

Atchutuni Bhavana 13MCMT01
Terli Venkatesh 13MCMT55
Surineni Sampath Kumar 13MCMT49

Contents

1 Introduction **3**

2 Modules in the system **3**

2.1 Data flow diagram 4

2.2 API Specification 4

2.2.1 Modules of the architecture 4

1 Introduction

The purpose of this document is to depict the high level design and the data flow diagram of bin packing vm consolidation algorithm project.

2 Modules in the system

Our system is divided into 4 modules. They are

1. **Main Module**

   This module provides standard interface for all modules to communicate with each other.

2. **User Interface**

   This module is the main interface to the user and is responsible for building, editing and updating the GUI.

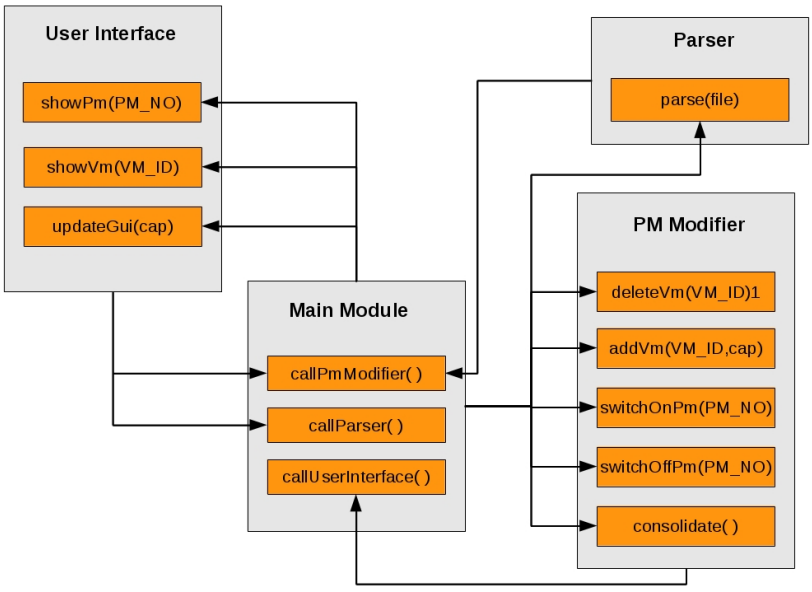

Figure 1: Interfaces between modules

3. **Parser** This module reads the input from file, parses it and initializes PMs and VMs as specified in it.

4. **PM modifier**

   This module is responsible for adding virtual machines(VM) and doing modifications to Physical Machines(PM)s. This is also responsible for consolidation operation.

2.1 Data flow diagram

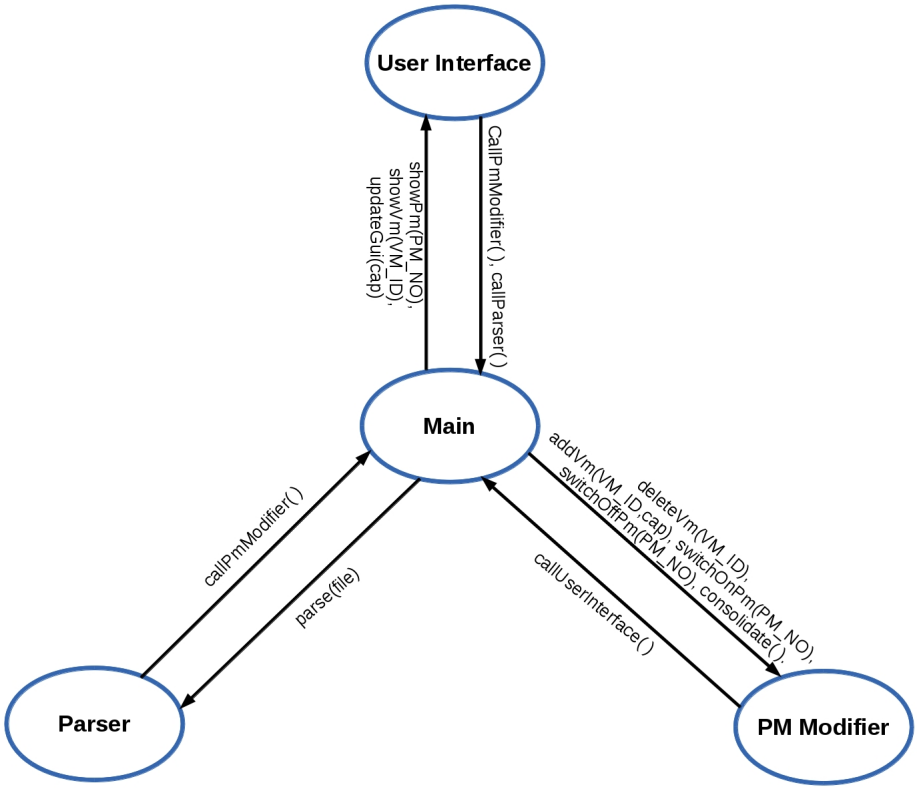

Figure 2: Data flow diagram

2.2 API Specification

2.2.1 Modules of the architecture

Main module

- **Functionality**

  This module is the central interface between all others modules. All the communications between the modules must pass through this module.

This provides a standard interface for all modules to communicate with each other.

- **Interface Description**

  callParser()

Purpose	: All the calls to Parser from other modules pass through this function.
Input Parameter	: fuction to be called in parser module and required parameters.
Output Parameter	: none
Called by	: User Interface module
Calls	: all the fuctions in Parser module

  callUI()

Purpose	: All the calls to User Interface module from other modules pass through this function.
Input Parameter	: Fuction to be called in User Interface module and required parameters.
Output Parameter	: none
Called by	: PM modifier
Calls	: All the fuctions in Parser module

  callPmModifier()

Purpose	: All the calls to PM Modifier module from other modules pass through this function.
Input Parameter	: Fuction to be called in PM modifier module and required parameters.
Output Parameter	: none
Called by	: Parser moduler and User Interface module
Calls	: All the fuctions in PM Modifier module through Main module

User Interface Module

- **Functionality**

  The main purpose of this module is to take input from the user and reflect the system state to the user.

- **Interface Description**

showPm(PM_NO)

Purpose	: This function takes the input from the parse file and creates a UI element for PM and displays it.
Input Parameter	: PM number
Output Parameter	: none
Called by	: PM modifier module through Main module

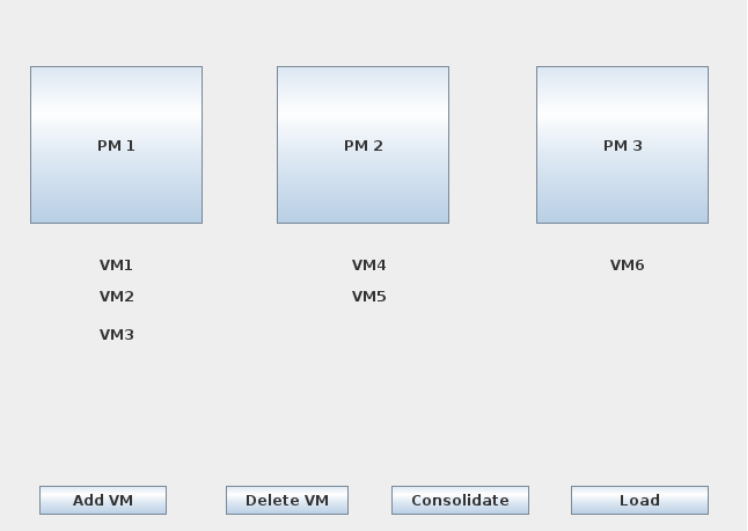

Figure 3: Main user interface

showVm(VM_ID)

Purpose	:This function takes the input from the parse file and creates a UI element for VM and displays it in corresponding PM.
Input Parameter	: VM ID
Output Parameter	: none
Called by	: PM modifier module through Main module

updateGui()

Purpose	:The purpose of this function is to update the GUI after a modification has been done to a PM like adding a VM or deleting a VM.
Input Parameter	: none
Output Parameter	: none
Called by	: PM modifier module

Parser

- **Functionality**

  The aim of this module is to parse a text file specified by the user, extract the information about PM's and VM's in it. It then initializes the PM's and adds VM's to them by the help of PM modifier module.

- **Interface Description**

  parse(filename)

Purpose	:The purpose of this function is to parse the file specified by the user and extract information about PM's and VM's.
Input Parameter	: file path specified by the user
Output Parameter	: none
Return Value	: If the file is not in the specified format it would display **wrong file format** message
Called by	: User Interface module through Main module
Calls	: createPM and addVM functions of PM modifier

PM modifier

- **Functionality**

  The operations of this module includes adding VM's to PM, calculating the residual capacity and consolidation.

- **Interface Description**

  addVm(VM_ID, cap)

Purpose	: The purpose of this function is to add VM's to the PM as specified by the parser.
Input Parameters	: VM ID and VM capacity.
Output Parameter	: none
Return Value	: If there is no enough space to add VM to a PM it outputs **No enough space** message
Called by	: Parser module through Main module
Calls	: showVm of User Interface Module through Main module.

deleteVm(VM_ID)

Purpose	: The purpose of this function is to delete a VM specified by the user.
Input Parameters	: PM number in which VM resides and VM ID.
Output Parameter	: none
Called by	: User Interface module
Calls	: none.

switchOffPm(PM _NO)

Purpose	: The purpose of this function is to switch off a PM specified by user.
Input Parameters	: PM number.
Output Parameter	: none
Return Value	: Displays **It is not possible to switch off this PM at this time** switching off was not successful.
Called by	: User Interface module through Main module
Calls	: updateGui() through Main module.

switchOnPm(PM_NO)

Purpose	: The purpose of this function is to switch on a PM specified by user.
Input Parameters	: PM number.
Output Parameters	: none
Return Value	:
Called by	: User Interface module through Main module
Calls	: updateGui() through Main module.

consolidate( )

Purpose	: The purpose of this function is to run Bin packing algorithm and consolidate all the VM's into minimum number of PM's.
Input Parameters	: none
Output Parameter	: none
Called by	: User Interface module through Main module
Calls	: none.