Chapter 3

Proposed Resource Provisioning and Scheduling Framework

The previous chapter discussed the research work accomplished in the area of Grid resource provisioning and resource scheduling. The study depicted that resource provisioning and scheduling challenges in Grid computing have not been addressed to a large extent. To provide a solution to resource provisioning and scheduling problems, a resource provisioning and scheduling framework has been proposed and designed in this chapter.

The proposed framework offers QoS parameter(s) based resource provisioning policies and a resource scheduling algorithm for Grid environment. A resource provisioning and scheduling framework provisions and schedules the resources along with achieving the practical constraints. QoS parameter(s) based resource provisioning policies have been designed and the policy rules have been specified in XML schema.

This chapter firstly discusses the requirements, architecture and components of the proposed framework. Then, it also discusses the objectives and commitments of QoS parameter(s) based resource provisioning policies. Finally, it illustrates the methods for expressing and exposing these policies in the Grid environment.

3.1 Goals of the Proposed Framework

The laid down research objectives have been accomplished through a proposed resource provisioning and scheduling framework where resource providers give the facility of resource provisioning to the user for optimum results, better services and avoid violations of the service level guarantees. Major Goals of the proposed framework are as follows:

- Providing QoS parameter based Resource provisioning policies and resource provisioning based scheduling.
- The implementation of this framework enables user to analyze customer requirements and define processes that contribute to the achievement of a product or service that is acceptable to the consumers.
- Framework assists organizations in enhancing customer satisfaction and contributes directly to the company's growth and institutional progress.

3.1.1 Framework Requirements

The detailed requirements for designing a system, which furnishes the solution for the desired problems are yet to be gathered. Unified Modelling Language (UML) [188] has been used to study, analyze and validate the functional requirements of the framework from different perspectives .
Requirement Analysis:

The key functions of the proposed resource provisioning and scheduling framework are:

- User Authentication
- SLA form for resource provisioning
- Searching Policy Repository
- Specification of the desired resources
- Execution of Resource Provisioning for Job Execution

These functional requirements of the Framework have been analyzed with the help of Use Case Diagrams and Sequence Diagrams as follows:

a. User Authentication- This use case shows the functionality of the system from the prospective of each user of the system. A use case is a coherent unit of functionality expressed as a transaction among actors and the system. In first use case, the successful registration of the user is demonstrated. User tries to login to access the resources and then he is asked to submit the basic information in registration page and password. Administrator confirms the registration of the user as shown in Figure 3.1.

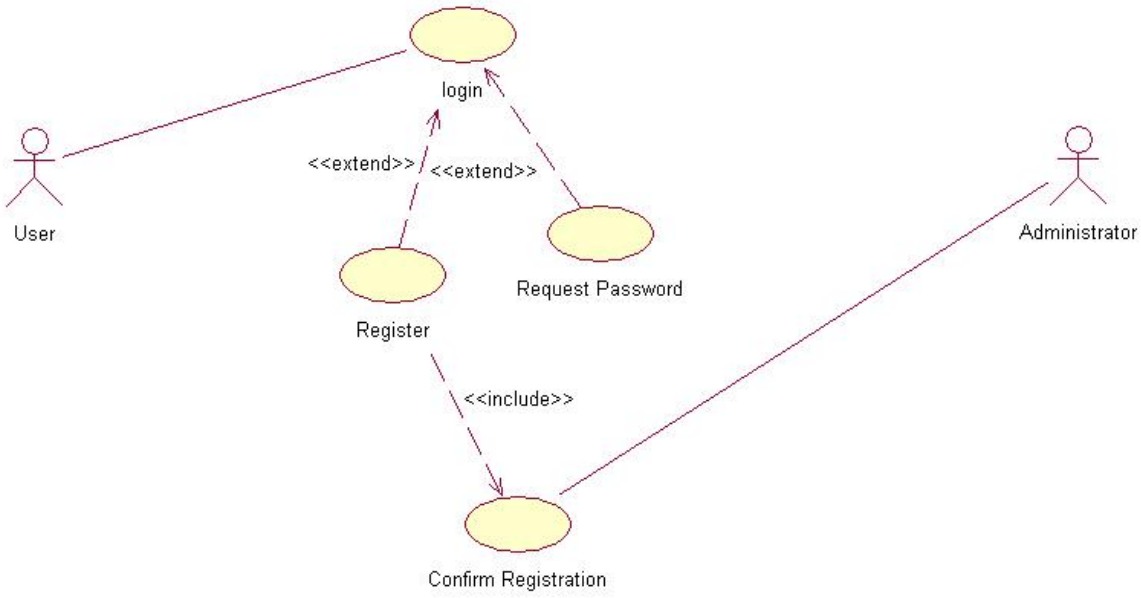

Figure 3.1: Use Case Diagram for User Authentication

After the authentication, user demands the resources for application's execution. Resource Information Center (RIC) will pass information about the availability of the resources to the Resource Provisioning Manager (RPM). RPM will provision resources to the user and then he will be able to access the resources as shown in Figure 3.2.

b. Sequence diagram of the successful execution of resource provisioning- In UML, sequence diagrams are useful design tools because they provide a dynamic view of the system behavior, which can be difficult to extract from static diagrams or specifications. In this case, the successful execution of resource provisioning has been shown using a sequence diagram in Figure 3.3. After login through the portal, user tries to access the resources. The user has to fill his requirements in the form of SLA. After going through

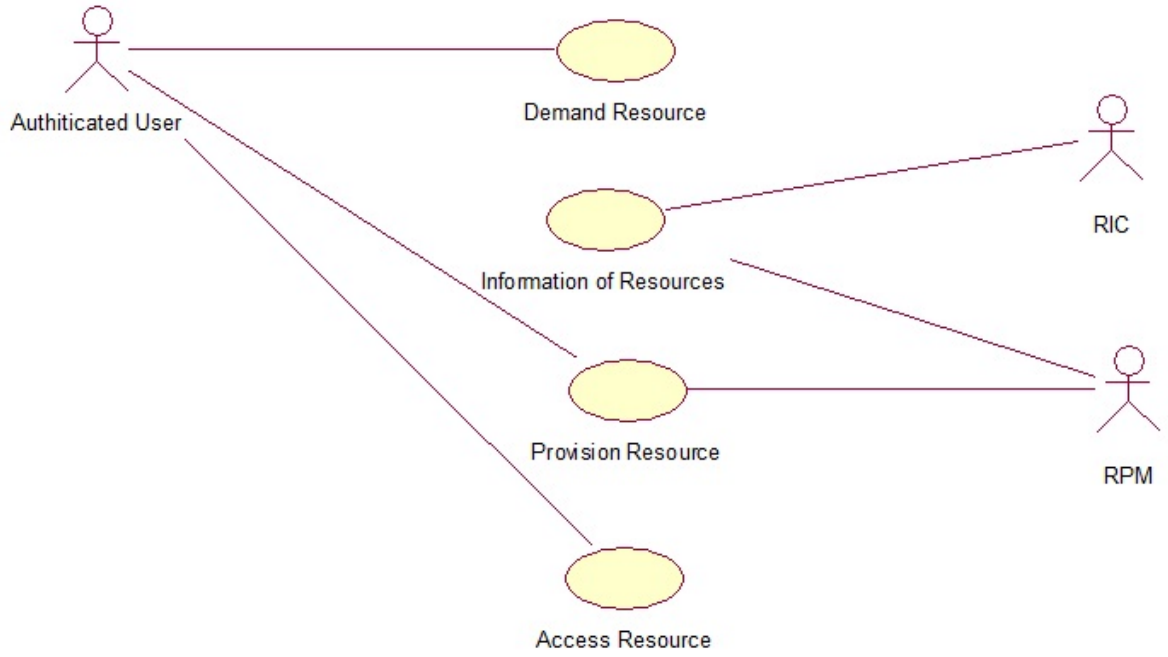

Figure 3.2: Use Case Diagram for Resource Provisioning

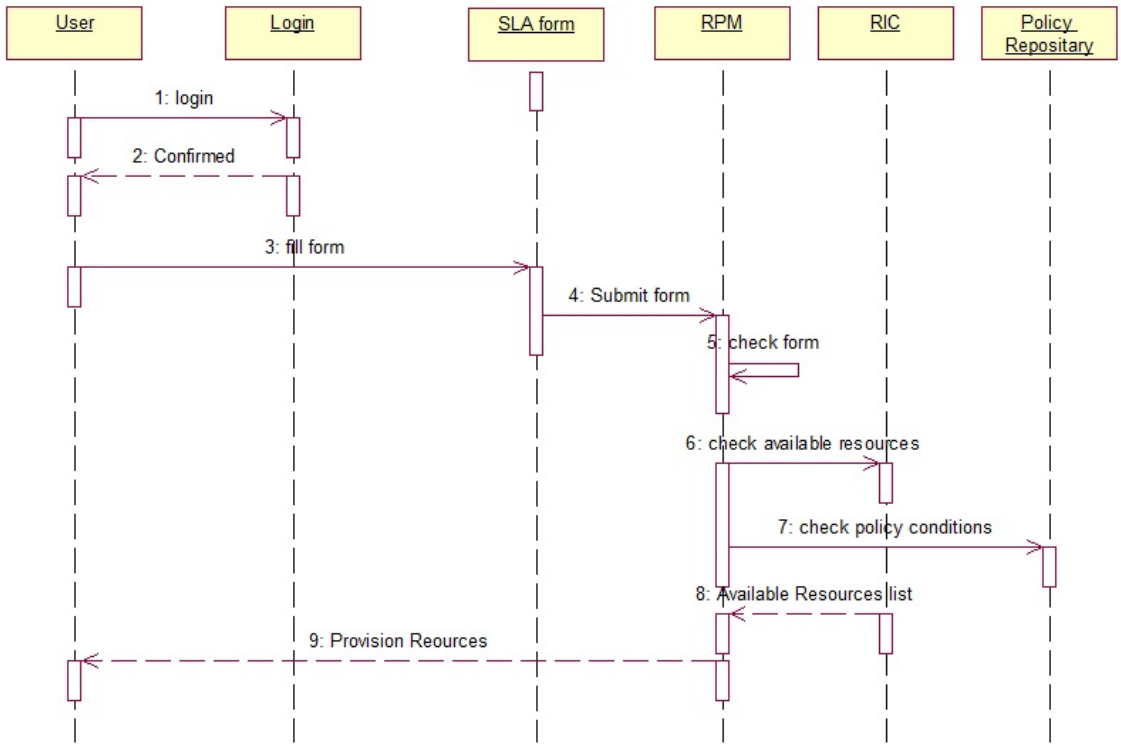

Figure 3.3: Sequence Diagram of Successful Execution of Resource Provisioning

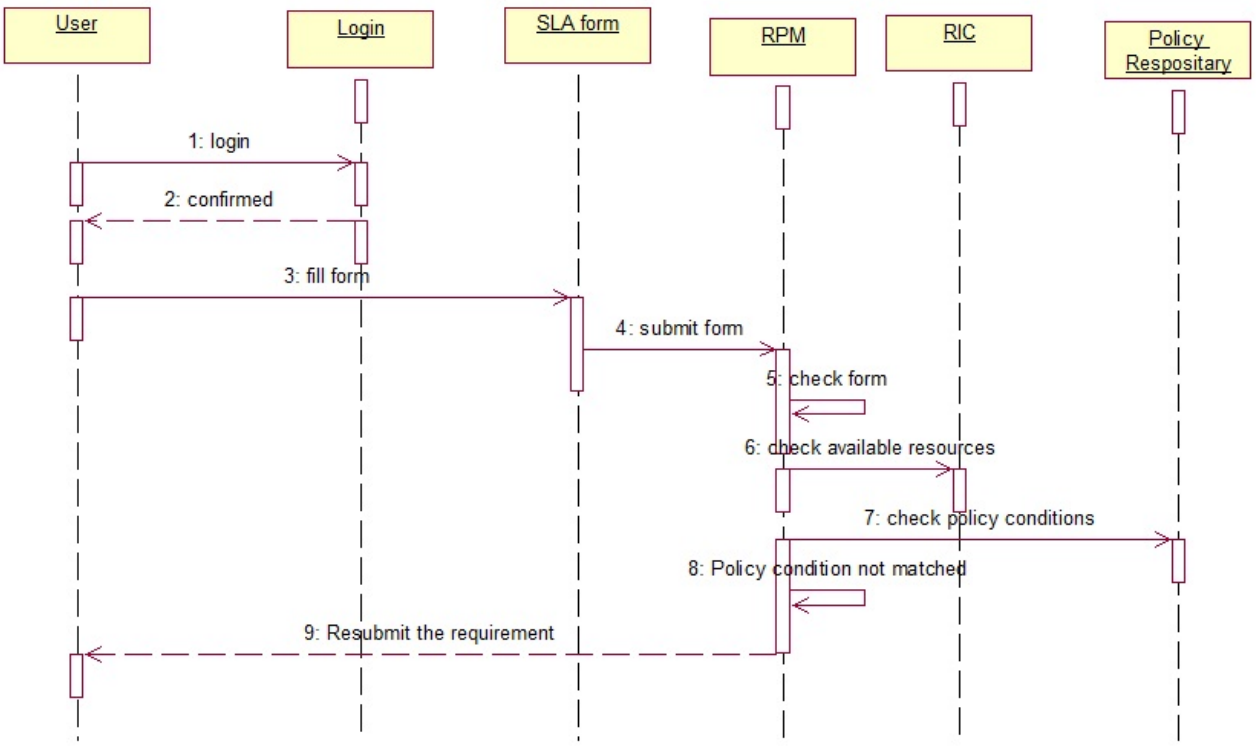

Figure 3.4: Sequence Diagram for resubmission of user's requirements

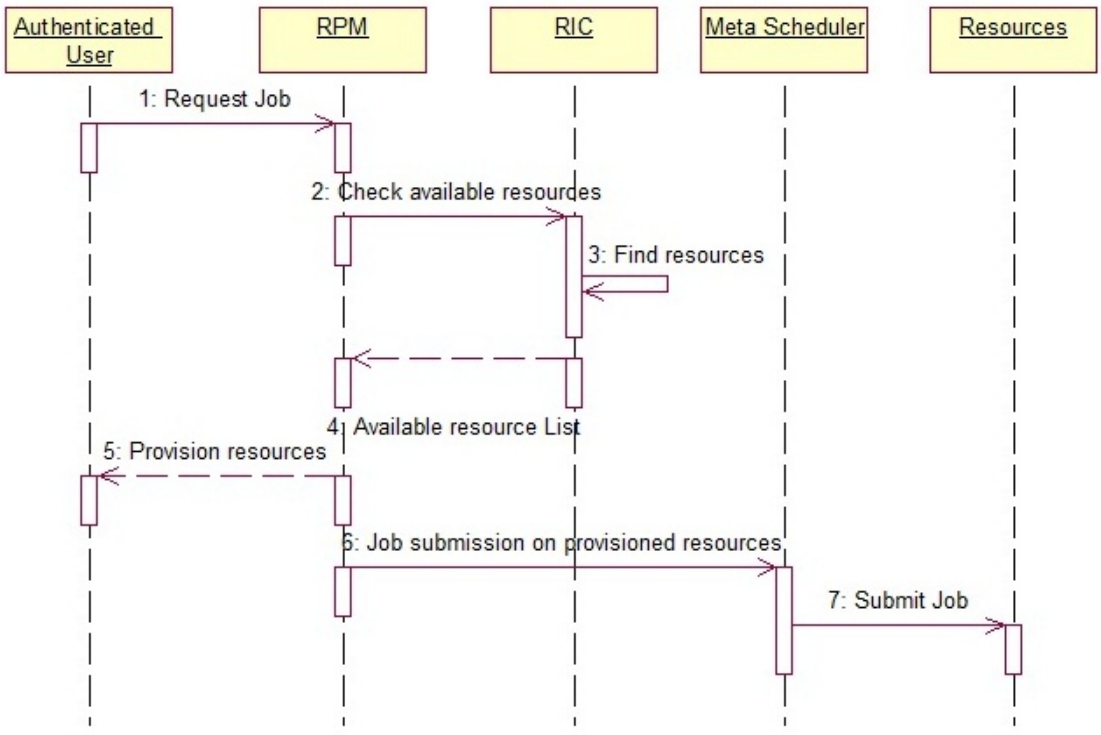

Figure 3.5: Sequence Diagram for job submission

the SLA form, RPM tries to take the information of available resources from the resource information center and simultaneously checks the policy conditions. If the policy conditions match then the RPM provisions the resources to users.

c. Sequence Diagram for resubmission of user's requirements - In this sequence diagram, the request of the resources for the execution of application through resource provisioning is shown. After login through the portal, user will try to access the resources. The user has to fill his requirements in the form of SLA. After going through the SLA form, resource provisioning manager tries to take the information of available resources from the resource information center and simultaneously will check the policy conditions. If the policy conditions do not match then the RPP will not provision the resources to users as shown in Figure 3.4.

d. Sequence Diagram for job submission - In this sequence diagram, job submission after resource provisioning is done. Authenticated user requests for job submission to resource provisioning manager. After resource provisioning, RPM submits the list of provisioned resources and applications to meta-scheduler. Then, meta-scheduler performs his task for optimal job's execution on the provisioned resources as shown in Figure 3.5.

3.1.2 QoS Requirements of the Framework

A Grid application is able to negotiate an SLA with the Resource Management System (RMS) on the basis of QoS [59]. The idea of QoS or SLA can be extended to placing bounds on the resource discovery or other Grid provided services such as data migration, scheduling, etc. So, to provide provisioning and scheduling, first of all, QoS parameters should be defined.

As per the literature survey, following QoS parameters have been identified for designing the resource provisioning and scheduling framework:-

a. Cost: Cost is identified as per unit of resources that are consumed by the users for execution of their applications.

b. Time: Time is calculated by subtracting the start time from the deadline time of user's application.

c. Security: Security is based on trust values of nodes and the trust of the node is identified on the basis of the past transactions/interactions and present

environmental characteristics.

d. Reliability: Reliability can be defined as the fault tolerance of the node. The fault tolerance of the node can be checked for data storage and other tasks. A performance criteria (like data transfer and computation capacity) is used to measure the reliability of any node.

The next section discusses the mode of operation of the proposed framework in detail.

3.2 The Proposed Framework: Mode of Operation

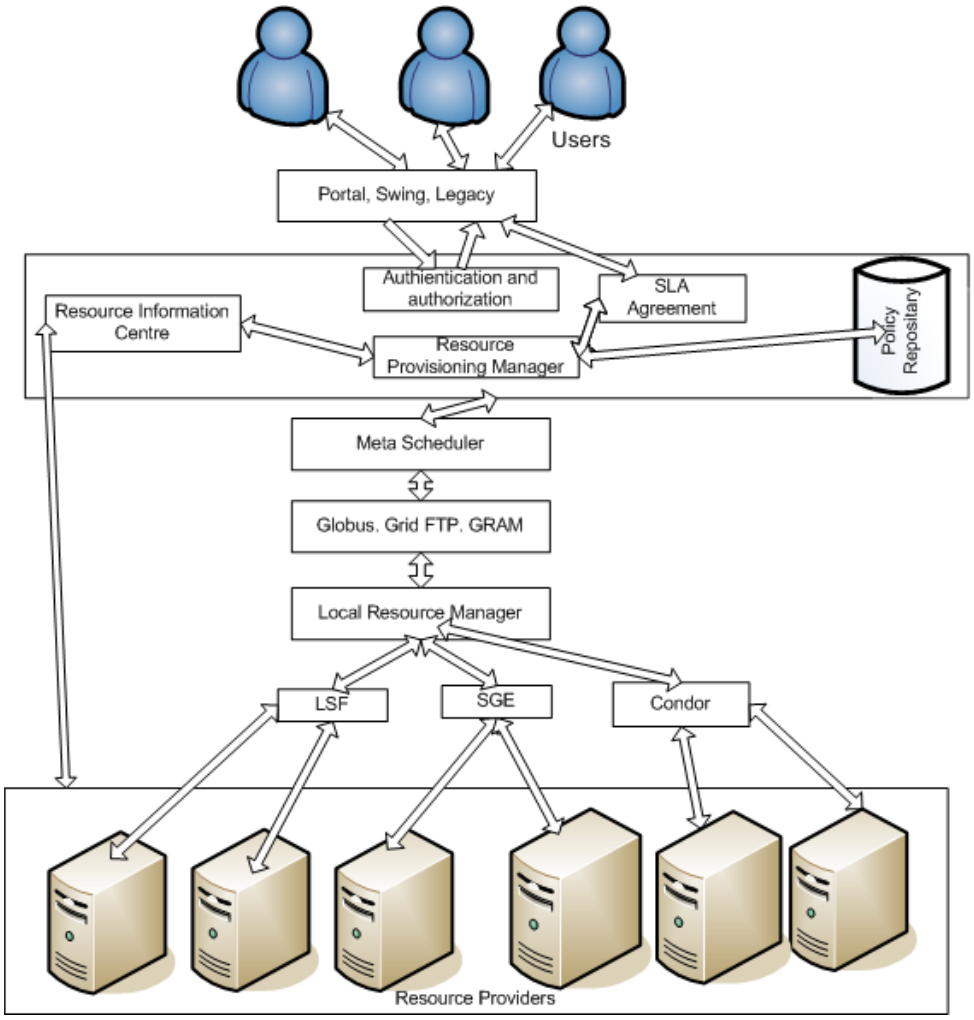

Figure 3.6: Resource Provisioning and Scheduling Framework

In the proposed framework (as shown in Figure 3.6), Resource Provisioning Manager (RPM) performs the tasks of managing the SLAs, maintaining information about the resources and dynamically updating the resources' status. In this process, RPM keeps a check over the provisioned resources and the proper execution of the jobs, information about SLAs and manages the resources' status and it also fills the policy repository. The XML formats of policies are stored in XML database that is depicted in the policy repository as shown in Figure 3.6. XML has been chosen as it is designed to be self-descriptive and it is a W3C recommendation [189]. The main aim of the resource provisioning and scheduling framework is to provision the resources to the Grid user with QoS and then schedule the application on the ingredient resources. The framework executes the requests in the following manner:

- In Resource Provisioning and Scheduling Framework, first of all user tries to access the resources through portal, swing or legacy. After that, the task of user's authentication and authorization is performed.

- After authentication, resource provisioning manager gives a SLA form to an authorized user and the user fills it to send the request for the availability of particular resources with proper specification for the execution of their application.

- RPM takes the information from the appropriate SLA. After studying the various parameters which the user has demanded, manager checks for the available resources. The selection of resources is made on the basis of QoS parameters defined in the SLA form.

- RPM then collects the information of available resources from the Resource Information Center (RIC). RIC contains all the information about the available resources.

- RPM provisions the requested resources to the user for the execution of application in the Grid environment only if the requested resources are available with the manager, according to the policy conditions stored in the policy repository.

- If the requested resources are not available according to the QoS based resource provisioning policies (discussed in the next section) to satisfy the QoS constraints then the resource provisioning manger asks to resubmit the QoS requirements again in the form of SLA.

- After provisioning of the resources, job is submitted through the Grid middleware.
- In the next step, the meta-scheduler via middleware communicates with the local resource manager for job submission. Local resource manager corresponds to different resource providers and the job is submitted to the provisioned resources.
- After getting the result, local resource manager gives it to the Meta scheduler through local scheduler. The result is then sent back to the RPM.
- Grid application gets the information and finally, the user collects the result.

Thus, this framework exhibits how QoS based resource provisioning can be done in the Grid environment. Resource provisioning policies have been identified as a part of this framework. QoS parameters based resource provisioning policies for the Grid computing environment are required to minimize the complexity of provisioning for job execution in Grid computing. With the use of a uniform policy, the number of migrations and complexity of policies can be less.

Next section discusses the detailed description of the QoS parameter based Resource Provisioning Policies.

3.3 QoS based Resource Provisioning Policies

Resource provisioning and scheduling framework provisions the resources on the basis of QoS based resource provisioning policies. The terminology associated with the policy is as follows:

- Resource Consumer (RC): Resource Consumer is an entity that demands resources for the execution of its applications.
- Resource Information Center (RIC): Resource Information Center collects all the information about the available resources.
- Resource Provisioning Manager (RPM): Resource Provisioning Manager is an entity that provides the resources to resource consumers as per the requirement of user's applications and keeps a track of provisioned resources.
- Resources: Resources can be clusters of computers, network latency, memory space, storage capacity, files and attached peripheral devices etc.

This policy standard is based on ISO:9000-2000, RFC 4745 [190].

3.3.1 Objectives and Commitments

The intent of QoS parameter(s) based Resource Provisioning Policy is to ensure that the policy will provision resources for the execution of job with QoS. It facilitates to:

i. Clearly understand the current and potential future requirements and expectations of the resource consumers.

ii. Deliver resources and services of the highest practicable quality, reliability and consistency that meet resource consumer's requirements.

iii. Establish and measure performance and customer satisfaction against appropriate objectives with security and reliability.

iv. Measure an appropriate level service performance and customer satisfaction by minimizing cost and time.

v. Enhance customer satisfaction by meeting all of its requirements.

Based on the four QoS parameters, four resource provisioning policies have been proposed and designed.

3.3.2 Cost based Resource Provisioning Policy (CRPP)

Under cost QoS parameter, a resource is used for the provisioning of job execution on consideration of cost of the resources. Cost is identified as per unit of resources that are consumed by the users for execution of their applications. It is an important aspect to be considered at the time of resource provisioning. After the minimization of cost of the resources, resources are provisioned. Thus, cost based QoS can be provided to the resource consumer. An XML schema of CRPP is as follows:

```
<?xmlversion="1.0"encoding="UTF-8"?><xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"attribute
Form Default="unqualified">
<xs:elementname="Cost">
<xs:annotation>
```

```
<xs:documentation>
Resource Provisioning Policy for Cost Operation
</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:elementname="application execution service">
<xs:complexType>
<xs:attributename="resource"type="xs:string"
use="required"/>
<xs:attributename="type"type="xs:string"
use="optional"/>
</xs:complexType>
</xs:element>
<xs:elementname="compute QoS parameter Cost"
minOccurs="0">
<xs:complexType>
<xs:attributename="capacity"type="xs:integer"
use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

3.3.3 Time based Resource Provisioning Policy (TRPP)

This resource provisioning policy has been made on the basis of time and takes static information as input. Time is calculated by the manager by subtracting the start time from the deadline time after the users have submitted their deadlines . The resource manager checks the resource providers' list to know about the resources that can satisfy the requirements of the users. This information is generally based on pre-execution analysis of the job or on historical data gathered after any previous execution. Suppose, if any node is available but no job is submitted to this node, then the performance of this node is predicted by its configuration like memory, processor's speed etc. If the configuration is high, then the response will be fast or vice versa. All the information is stored in RIC as shown in Figure 3.5. Then, the resource manager provisions the resources to users. The main aim of the TRPP is to minimize the time.

```
<?xmlversion="1.0"encoding="UTF-8"?><xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"attributeFormDefault="unqualified">
<xs:elementname="Time">
<xs:annotation>
<xs:documentation>
This is Time based Resource Provisioning Policy
</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:elementname="application execution service">
<xs:complexType>
<xs:attributename="resource"type="xs:string"
use="required"/>
<xs:attributename="type"type="xs:string"
use="optional"/>
```

```
</xs:complexType>
</xs:element>
<xs:elementname="compute QoS parameter Time" minOccurs="0">
<xs:complexType>
<xs:attributename="startTime"type="xs:dateTime"
use="required"/>
<xs:attributename="endTime"type="xs:dateTime"
use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

3.3.4 Security based Resource Provisioning Policy (SRPP)

Under security QoS parameter, a resource is used for provisioning the job execution on consideration of security of the node. To enable more effective and security aware resource provisioning, it is desirable to know the Security Demand (SD) from Grid users at the time of job submission and the Trust Level (TL) assured by a resource provider at the Grid site. The first step is to issue a SD to all the available resource sites which is done by the user. The trust model requires assessing the resource site's trustworthiness, called the TL of a node. This information is taken from RIC unit. It is calculated to avoid run-time failure. Trust is the firm belief in the competence of an entity to behave as expected such that this firm belief is a dynamic value associated with the entity and it is also subject to the entity's behavior and applies only within a specific context at a given time [191]. When an application is scheduled to execute on a resource, the trustworthiness of node also reflects the reliability of the node's services. The TL quantifies how much a user can trust a site for successfully executing a given job. A job is expected to be carried out successfully when SD and TL satisfy a security-assurance condition ($SD \leq TL$) during the job mapping process [192].

The failure probability of a resource/machine as a function, which is dependent on the difference $SD_i - TL_k$ has been defined. The formula (4.1) presented below expresses the failure probability of a resource r_k with trust level value TL_k , to a job j_i with a specific SD_i value [193]. In the resource provisioning and scheduling framework, a job could be delayed or dropped, if the site TL is lower than the job SD. The SD is a real fraction in the range [0 , 1] with 0 representing the lowest and 1 the highest security requirement. The TL is in the same range with 0 for the most risky resource site and 1 for a risk-free or fully trusted site. The negative exponent indicates that the failure probability of a scheduling grows with the difference $SD_i - TL_k$. The failure probability of executing a job, with a job SD on a site with TL, is modeled by an exponential distributed failure function as follows:

$$P_f(j_i, r_k) = \begin{cases} 0, & \text{if} SD_i \leq TL_k \\ 1 - \mathrm{e}^{-\alpha(SD_i - TL_k)} & \text{if } SD_i > TL_k \end{cases} \tag{3.1}$$

All resources in Grid computing are shared and distributed, therefore trust relationship is very crucial for provisioning of resources for job execution. The provisioning of resources is based on the accuracy of the feedback provided about the resources. After the calculation of failure probability of executing a job, resources are provisioned.

```
<?xmlversion="1.0"encoding="UTF-8"?><xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"attributeFormDefault="unqualified">
<xs:elementname="Security">
<xs:annotation>
<xs:documentation>
This is Security based Resource Provisioning Policy
</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:elementname="application execution service">
```

```
<xs:complexType>
<xs:attributename="resource"type="xs:string"
use="required"/>
<xs:attributename="type"type="xs:string"
use="optional"/>
</xs:complexType>
</xs:element>
<xs:elementname="compute Trust of node" maxOccurs="0">
<xs:complexType>
<xs:attributename="value"type="xs:integer"
use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema >
```

3.3.5 Reliability based Resource Provisioning Policy (RRPP)

Reliability based RPP is based on reliability of the node. Reliability of the node has to be checked before provisioning of the resources. With the help of reliability parameter, the fault tolerance of the node can be checked for data storage and other tasks. A performance criteria (like data transfer and computation capacity) is used to measure the reliability of any node. Node can be ranked according to its ability and it can be measured in the form of computation power and data transfer capacity.

By analyzing the updated RM's reports, RIC generates the reliability probabilities $P_{r_k}, k = 1, ..., n$ for each resource i. The execution of a job on the resource i can then be aborted with the probability defined as follows:

$$P_{rb}(r_k) = (1 - P_{r_k}) \tag{3.2}$$

The reliability probability (also referred to as prediction of node failure) P_{r_k} was introduced by Rood and Lewis in [194] as the real fraction in the range [0,1], which utilizes historical data to forecast the availability of Grid nodes. The higher P_{r_k} value means the smaller probability of the failure of node execution due to some additional network problems like a disconnection of power and node's failure etc. After calculation of reliability of the node, resources are provisioned.

```
<?xmlversion="1.0"encoding="UTF-8"?><xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"attributeFormDefault="unqualified">
<xs:elementname="Reliability">
<xs:annotation>
<xs:documentation>
This is Reliability based Resource Provisioning Policy
</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:elementname="application execution service">
<xs:complexType>
<xs:attributename="resource"type="xs:string"
use="required"/>
<xs:attributename="type"type="xs:string"
use="optional"/>
</xs:complexType>
</xs:element>
<xs:elementname="Compute Reliability of resource or node ">
<xs:complexType>
<xs:attributename="startTime"type="xs:dateTime"
use="required"/>
```

```
<xs:attributename="endTime"type="xs:dateTime"
use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema >
```

3.4 Conclusion

This chapter discussed the proposed Resource Provisioning & Scheduling Framework and QoS parameter based Resource Provisioning Policies as an outcome of the research work. The detailed requirements of Resource Provisioning & Scheduling Framework have been analyzed. Further, the mode of operation of QoS based resource provisioning and scheduling in the Framework has been discussed. In the next chapter, resource scheduling algorithm which caters to the scheduling aspects of the framework has been proposed.