# Content

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The High Level View of Layered Architecture of Generic Framework</td>
<td>59</td>
</tr>
<tr>
<td>3.2</td>
<td>The Generic Framework for Integration of Multi-agent Knowledge-based System and Distributed Database Grid</td>
<td>62</td>
</tr>
<tr>
<td>3.3</td>
<td>Conclusion</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>75</td>
</tr>
</tbody>
</table>
Chapter 3

The Generic Framework for Integration of Multi-agent Knowledge-based System and Distributed Database Grid

This chapter presents the integrated generic framework of multi-agent knowledge-based system accessing distributed database grid. The framework shows how heterogeneous and distributed data resources are integrated via grid middleware and how knowledge extraction, presentation and delivery services are provided on top of the distributed database grid environment by integrating multiple collaborative agents. It allows combining different types of information as a single entity to gain a more complete scenario and to integrate the similar types of information about different entities. Through integration of knowledge-based component, content is turned into something more than just a collection of data. The demonstrated generic framework provides a standardized interface which allows data resources, such as relational, XML databases and file-based data, to be federated, accessed and integrated across the Internet via OGSA-DAI (Open Grid Services Architecture - Data Access and Integration) - a data grid middleware.

Section 3.1 of the chapter presents the high level view of a layered architecture of the generic framework. The layered architecture contains five layers: Data Resource Layer, Communication Layer, Information Layer, Knowledge Layer and Application Layer. It also specifies the detailing about activities and services offered by each layer. Section 3.2 of the chapter presents the detailed view of the generic framework for integration of multi-agent knowledge-based system and distributed database grid. This section also provides the detailed description about the set of activities and services offered by each layer.
3.1 The High Level View of Layered Architecture of Generic Framework

Data sources used in today’s business and scientific enterprises are often heterogeneous and scattered across the enterprise or even across the world. The data storage format is also varies such as structured form, semi-structured form or even unstructured form like XML and flat files. Enterprises are generally striving for real-time and light-weight data integration approach which minimizes data latency and data duplication. Also, there is a need for a multi-agent system that can manage databases of scientific and commercial applications in more generalized and implicit way.

The high level view of layered architecture designed for integration of multi-agent knowledge-based system and distributed database grid is demonstrated in Figure 3.1. The architecture consists of five layers: Data Resource Layer, Communication Layer, Information Layer, Knowledge Layer and Application Layer. The transformation of data into knowledge is accomplished with the level of layer increases. Also, each layer is following some security and control policies. Each layer builds upon the standards, activities and services of the layer below it. Each layer also provides a standard set of activities and services to the above layer. The architecture provides various functionalities which we call services and activities. Some examples of these services and activities are shown in the Figure 3.1. The architecture presented here can also effectively used by various data intensive scientific and business applications like Scientific Research applications, E-business applications, Educational applications, Healthcare applications, Genomics applications, Physics applications, Satellite Imaging applications and many more. The following are the details about the services and activities offered by each layer.

3.1.1 Data Resource Layer

This is the lowest layer of a high level architecture. The data resource layer is the fabric layer which is directly connected to the several data resources. The data resources are heterogeneous & geographically distributed by nature as shown in the Figure 3.1. The data resource layer fetches the data from heterogeneous and geographically distributed databases mounted on data grid. Also, it can contain communication network system including LAN (local area network), WAN (wide area network) and wireless network.
Chapter 3: The Generic Framework for Integration of Multi-agent Knowledge-based System and Distributed Database Grid

3.1.2 Communication Layer

The communication layer is built upon the data layer. This layer includes the communication and authentication protocols required for data grid transactions such as exchange of data between data resources and verification of the identity of users and several resources. This layer also contains connectivity protocols used to accomplish the activities and services such as enabling secure initiation, resource monitoring, control of resource-sharing operations and many more.

3.1.3 Information Layer

The information layer rests above the communication layer. It provides the data access and integration services. It is also responsible for data delivery and data transformation. Apart from this, it offers some core and user level services like metadata management, security, grid information service and so forth. It allows the sharing and management of data...
coming from distributed datasets in the data grid environment and facilitates the multiple database federation by imparting the standards and services incorporated into the data grid. For that, it uses various APIs (Application Programming Interface) to implement transactions among data resources which are enabling applications to federate multiple databases at a single time.

This layer consists of services which interact with knowledge layer and application layer. Some of the examples of such services are data source service (to find the best available data source available in data grid), information service (to manage and monitor the current active services available in the data grid), and metadata service (to find the physical address of the logical data). The services of the information layer provide a holistic view of data to the higher level layers i.e. knowledge and application layers.

3.1.4 Knowledge Layer

The knowledge layer is directly accessible by application layer and it is built upon the information layer. It may implement the artificial intelligence techniques like fuzzy set theory & fuzzy logic to convert the data into its meaningful form called knowledge. It fetches the data from information layer and applies a specific set of services and activities which extracts the knowledge from the fetched data. The extracted knowledge is then after presented and delivered to application layer which may use for analysis and decision making process by client applications and users.

3.1.5 Application Layer

This is the highest layer of an architecture and accesses all other layers for resources where and when required. It also allows access to remote data resources and digital libraries, and provides overall management of all applications running. This layer can be customized according domain specific applications and used by several data intensive research and business applications. The users are directly accessing the application layer in order to access the lower layers to fulfill their requests. The application layer then passes the requests to activities and services reside in the lower layers and send the response back to the users about their status of the request.
3.2 The Generic Framework for Integration of Multi-agent Knowledge-based System and Distributed Database Grid

The grid community has historically focused on tools, applications and infrastructure for reliable and secure resource sharing within dynamic, heterogeneous and geographically distributed virtual organizations. On the contrary, the agent community has focused on autonomous problem solvers that can act flexibly & automatically in uncertain and dynamic environments as stated earlier. As the scale and ambition of grid and agent deployments increase, agent systems requiring robust infrastructure and grid systems require autonomous, flexible behaviors [3, 4]. This research work aims to provide the generic framework which provides an integrated approach to leverage the power of both grid and agent technologies. Moreover, it integrates the knowledge-based component to provide intelligent decisions with justification for real time data for analysis and decision making that can be directly accessible by client applications and users. The heterogeneous and dynamic nature of the data grid environment makes it a challenging work for flexible, large-scale, coordinated service sharing among dynamic and disparate collections of individuals, institutions and resources. Agents are autonomous, adaptive and cooperative by nature and emerging as dynamic, flexible and extensible mediators for facilitating grid services in a data grid environment. The generic framework follows agent-oriented approach for modeling, designing and developing agent mediated grid services.

In a distributed computing environment, multi-agent system can be used for a variety of reasons. Having multiple agents could speed up a system's operation by providing a method for parallel computation. Another benefit of multi-agent systems is their scalability. Since they are naturally modular, it should be easier to add new agents to an existing system to add new capabilities. Systems whose capabilities and functionalities are likely to change over time or across agents can benefit from this advantage [8].

Apart from this, there is an emerging need for multi-agent systems that can manage databases of scientific and commercial applications in more generalized and implicit way. The generic framework hides the heterogeneity and provides transparent and single point access of heterogeneous & geographically distributed databases to client applications and users and handles the data access and integration through multiple collaborative agents.
Nowadays, data within the research and business organizations generally resides in multiple and heterogeneous databases. These databases may own by different administrative domains. Therefore, both research and business organizations require these heterogeneous and geographically separated data on a real time basis for analysis and decision making. Data integration challenges have also been addressed by ETL (Extraction, Transformation and Loading) approach earlier. ETL approach comprises with three steps: extract, transform and loading. ETL approach extracts data transforms it to fit operational needs and loads it into the end target. However, the ETL approach fails to integrate the real time data due to the latency involved in cleaning, transforming and moving the data. ETL approach is having its own limitations as they fail to integrate the data in real time for which organizations generally strive. The generic framework explained in following section overcome the limitations of ETL as it accesses and integrates the data on a real time basis from geographically distributed database grid. The framework also shows how different heterogeneous data resources are integrated via grid middleware and how knowledge extraction, presentation and delivery process is performed on distributed data grid environment by using multiple task agents. It allows combining different types of information as a single entity to gain a more complete scenario and to integrate the similar types of information about different entities. The detailed view of the architecture of the generic framework for integration of multi-agent knowledge-based system and distributed database grid is presented in Figure 3.2.

The Figure 3.2 demonstrates the generic framework for integration of multi-agent knowledge-based system and distributed database grid. The framework consists of four layers: Grid Fabric Layer, Middleware Layer, Knowledge Layer and Application Layer. Each layer is separated by a dotted line. The layers are connected in a bidirectional way. The grid fabric layer is the lowest layer of the generic framework. Middleware layer is built upon the grid fabric layer. The middleware layer provides the core and user level services which are wrapped up in agents. Different collaborative agents are contained by middleware layer which are responsible for realization of requests made by the client applications and users or upper level layer. The knowledge layer is built upon the middleware layer. The knowledge layer consists of knowledge-based component which may implement through artificial intelligence techniques. This will turn the data into something more than just a collection of data. The services offered by the knowledge layer used for analysis and decision making. Knowledge-based systems are providing a new
dimension to the advancement of information systems. The analysis which can be performed by the knowledge-based system is providing optimal result with minimum human intervention.

Figure 3.2: The Generic Framework for Integration of Multi-agent Knowledge-based System and Distributed Database Grid
Fuzzy logic are highly suitable and applicable techniques for developing knowledge-based systems that can handle uncertainty and partial information. Fuzzy logic is a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth. Truth values exist between ‘entirely true’ and ‘entirely false’. The significance of fuzzy logic derives from the fact that most modes of human reasoning and especially common sense reasoning are approximate by nature. Therefore, fuzzy logic is the logic of underlying modes of reasoning which are approximate rather than exact. [11]. On top of the framework, there is an application layer which may comprise with various client applications and users. It contains the interface facilitator agent which provides an interface to access core, data, user and knowledge-based services offered by the underlying layers. The following sections provide the detailing of activities and services offered by each layer of the generic framework.

### 3.2.1 Grid Fabric Layer

The grid fabric layer is the lowest layer resides in the framework. It consists of data and network resources used by the data grid. The data grid may use the network resources like switches, routers, operating systems etc. It also consists of various relational heterogeneous database resources which may be geographically distributed. These data resources are to be exposed via middleware layer. Currently these include:

- Relational data resources, e.g. Oracle, MySQL, SQL Server etc.
- XML data resources, e.g. Xindice.
- Files data resources, e.g. files and directories

The data resources located in grid fabric layer are accessible by middleware layer through an interface exist between grid fabric layer and middleware layer. The interface contains various classes which invoke JDBC drivers, XMLDB drivers, or other classes to manage communications to and from heterogeneous & geographically distributed data resources.

### 3.2.2 Middleware Layer

The middleware layer builds on grid fabric layer. It mainly provides the core, data and user level services. It is a key component of the generic framework. Through middleware services, client applications and users are enabled to access and manage various
heterogeneous databases through secure and transparent grid system and implement the grid interoperability characteristics, while hiding the underlying heterogeneity and dynamics of such databases.

The core, data and user level services offered by middleware layer are encapsulated in various task agents. These agents are part of a multi-agent system and they are collaborating with each other in order to provide above mentioned services. These agents generally receive a request for a specific service from higher level layer. In order to provide the requested service, agents may also need to access the services offered by lower level layers. These agents are triggered when a request is sent by a user or the state of the data grid is changed. To provide the services offered by middleware layer, the agents need to access the data resources which are located in grid fabric layer. Figure 3.3 shows the abstract scenario where multiple collaborative agents are working together within a distributed database grid environment in order to provide the data grid services to the client applications and users. As shown in the Figure 3.3, agents need to collaborate with each other in order to provide core, data and user level services.

The services offered by middleware layer provide support for data storage, data retrieval and data discovery processes through the data management infrastructure and services on the top of grid fabric layer where the actual data resources are located. The services offered by multiple collaborative agents of the middleware helps in designing and development of complex data and knowledge intensive applications.

Figure 3.3: Multiple Collaborative Agents in Distributed Database Grid Environment
The following are the core, data and user level services offered by different collaborative agents reside in middleware layer. Each service may be provided by using one or more collaborative agents.

- **Data Integration**

The data integration service mainly focuses on publishing and discovering of the data stored in databases of grid environment and allows flexible representation of a data resource. It collects data from multiple different data sources and used to expose heterogeneous data resources. It has to interact with other services to accomplish the data integration tasks such as multiple database federation service, data delivery service and data transformation service. It provides APIs (Application Programming Interface) and interfaces to the client application and users for accessing and integrating the data stored in heterogeneous and scattered relational databases (MySQL, Oracle, DB2, SQL Server etc.), XML (Xindice etc.) and files.

- **Data Delivery**

Server system such as JDBC uses a traditional client which sends a request (e.g. a query) to the database, and receives the result in return, is called synchronous request model. This synchronous request model will not always meet the extreme performance and capacity requirements of some grid applications. Therefore, it is important to support a range of data delivery alternatives. For example, directly deliver the data to a third party, which is consuming the result of the query, to hold the result in the data service until it is required by another service and so forth [6].

The data delivery service of the middleware layer provides two options to the client applications and users: synchronous and asynchronous. In synchronous delivery, the consumer sends a request document and in return receives a message containing the results. In asynchronous delivery, the result is not returned to the consumer immediately. It may remain in the service until it is accessed by a consumer. It may also send the result directly to another service, if required.
• **Data Transformation**

Data transformation service offers a set of utilities and objects which allow the automation of extract, transform and load operations to or from a database. It exposes data in schema P to client applications or users as data in schema Q. Through data transformation services, data can be transformed into suitable formats near the data resource, thus avoiding unnecessary data movement. For example, we can use the TupleToCSV service provided by OGSA-DAI (Open Grid Services Architecture – Data Access and Integration) – a grid middleware, which converts a list of tuples (i.e. database records) into Comma Separated Values (CSV), we can change the tuples output by relational activities (e.g. SQLQuery) to the XML WebRowSet format and so forth.

• **Uniform Data Access**

A uniform data access service is a typical service offered by the generic framework, which allows data to be accessed through a uniform and standardized interface. It hides diversity of data resource types, vendors, middleware, schema, metadata and other details to the end users. It provides homogeneous access to heterogeneous distributed data or information, which are coming from several data resources. The service wraps data resources to provide a uniform data access interface to disparate heterogeneous data resources. This abstraction allows the services to be used as the base primitives for higher-level services offering more sophisticated data access patterns such as data federation and query processing.

• **Multiple Database Federation**

One of the aims of the data grid environment is to promote the systematic sharing of scientific and commercial data [6]. A recent study of the requirements of some early grid applications concluded that a large amount of data is gathered from billions of data resources. For example, Large Hadron Collider and telescopes generates petabytes of data. Even, portable devices like medical scanners, networks of environmental sensors also generate a significant amount of data every year. These data resources are being accessible in a grid environment [2]. This scenario motivates researchers to combine different types of information as a single entity to gain a more complete and concrete picture. At a same
time, it is needed to aggregate the same types of information about different entities. These require support for integrating data from multiple data sources.

In the data grid environment, the constituent databases are interconnected via a computer network and they may be geographically decentralized. The multiple database federation service of the middleware layer provides a uniform user interface, enabling grid applications and users to store and retrieve data in multiple, distributed and heterogeneous databases with a single query. To achieve this, the multiple database federation service decomposes the query into sub-queries for submission to the relevant constituent DBMS's. After submission of sub-queries, the system must composite the result sets of the sub-queries. There are various database management systems exist and they all are employed different query languages. For that, the service can apply wrappers to execute the sub-queries to translate them into the appropriate query languages. The service uses a common query language used to contrast, combine, manipulate and analyze the data which may come from several heterogeneous and distributed data resources.

- **Metadata Management**

Metadata management service is one of the key components of the middleware layer. It is responsible for collecting and managing the metadata to provide information about involved computing & data resources like data machines, data repositories, data schema, networks, programs, etc. As a result, metadata can represent a key element to effective resource discovery and utilization in the grid environment.

Metadata management service publishes metadata about an underlying data resource that is exposed, its capabilities, the service capabilities and the connection infrastructure that uses to communicate with the data resources. It provides metadata about the DBMS that is being exposed to the data grid. For data sources, the database schema may be extracted from the service. This may be helpful to higher level services such as distributed query execution. Metadata may be provided either statically or dynamically. The static metadata is extensible so that developers implement a framework to access databases within a grid context can provide corresponding metadata for the databases they expose to the grid.

Metadata management service is also offered the metadata to other services such as authentication & authorization service and security service. The metadata used for access mechanism are users IDs, password, logged in information, role of user, access rights &
privileges etc. Metadata management service also provides the metadata about grid node, its location, IP address, storage capacity etc. The users of the data grid belong from different categories like registered users, anonymous users etc. Registered users are allowed to browse and modify the databases of the data grid by using certain criteria. The unregistered or anonymous users are only allowed to browse the information about no. of grid nodes, type of databases etc. This service can be customized according to the requirement of client applications or users of the specific application domain.

The metadata management service is application specific. Many disciplines are developing extensive bodies of practice concerning the organization and a description of their data. Applications like portals, workflows or personalized code, first examine metadata according to user requirements. They use this metadata to locate the data, describe which data are accessed, and determine what transformations are necessary to steer analyses and visualizations. It is also used to carry forward information into automatically generated metadata associated with result sets [6]. It is, therefore, widely recognized that metadata will be very important for many grid applications. At present, the use of metadata in grid applications is used mostly for mapping the logical names for datasets into the physical locations where they can be accessed [9, 5]. As users and developers develop more sophisticated applications, and also the grid expands into new application areas such as engineering, physical sciences, life sciences, earth sciences, more advanced and sophisticated metadata systems and tools will be required. The consequence is likely to be a Semantic Grid [1] that is analogous to the Semantic Web [10].

- **Authentication & Authorization**

As security is a very crucial and complicated issue especially in a grid environment, it is important to set up the method to identity authentication and access authorization so that the user’s request can be executed with the appropriate privileges. The authentication & authorization service of a middleware layer avoids direct access to data resources by client applications and users and presents the data grid resources from unauthorized access.

Any user, who wants to access the data grid, will first query metadata management service that contains corresponding metadata information like logging information. According to the logging information, different users may map into different roles that are granted with
different access permission levels based on the corresponding user metadata. After determining the proper access authorization, the request is either accepted or rejected.

- **Security**

  The security service of data grid middleware implements a simple authentication & authorization mechanism that consists of a Role Map file and an implementation class that does the role mapping between the grid credentials used to access the system. Therefore, the grid middleware security service uses a specific type of metadata provided by the metadata management service and authentication and authorization service. As based on current system security policies, an access control and authorization mechanism will be implemented with fine granularity to ensure the access security in a more strict way to prevent any false or malicious manipulations on each database in the data grid.

- **Grid Information Services**

  The grid information service of the grid middleware provides the information about the grid nodes which are registered in the data grid environment. It provides the information about registered nodes, live nodes and active nodes of the data grid environment and also provides the facility to manage and manipulate a specific node in the data grid. If the new node comes, first it has to register itself within the data grid through grid information service and then the system starts the communication with the newly registered data grid node in order to provide a specific set of services.

- **Data Backup & Replication**

  Backup services are used for data safety. It keeps a valid copy of data and restores them on master site when the data is lost. Data replication is the user level service generally built upon the core services offered by the data grid environment. Replication is important for all DBMS as it can improve system availability, data consolidation and data distribution. However, a data grid environment is often heterogeneous and highly scattered and large in scale, data replication service of the middleware distributes data resources into remote sites and keeps them synchronized with the original site on a particular level. Distributed data are usually processed by these remote sites, and sometimes the changes need to be propagated back to the original site.
• **Communication Services**

Communication services consist of protocols used to query resources in the grid fabric layer in order to conduct data transfers between data resources. These protocols are built on core communication protocols such as TCP/IP and authentication protocols such as PKI (Public Key Infrastructure), passwords, or SSL (Secure Sockets Layer) etc. Moreover, file transfer protocols (such as FTP, GridFTP etc.) provide services for efficient transfer of data between two data resources which reside in the data grid environment.

**3.2.3 Knowledge Layer**

The data grid environment provides a most promising framework for future implementations of high-performance data intensive distributed applications. Although today the grid is mainly used for scientific applications, in the near future it will be used for industrial and commercial applications. In these areas, knowledge extraction, presentation and delivery services are very important and critical. Furthermore, the Internet is shifting from an information and communication infrastructure to a knowledge delivery infrastructure. The extraction, presentation and delivery of knowledge from geographically distributed sources will be increasingly important in many typical operational and strategic activities [7]. A vast amount of data and/or information is currently stored in digital data repositories, however it is often difficult to understand what are the important and useful information in these massive data sets.

The grid has recently emerged as an integrated infrastructure for coordinating resource sharing and problem solving in distributed environments. Grid middleware targets technical challenges such as security, communication, fault detection, scheduling, information and data access [2]. Therefore, it is very useful to develop a knowledge layer on top of basic grid layers by deploying grid services for the extraction and delivery of knowledge from large distributed & heterogeneous data repositories.

The generic framework is a significant step in the process of studying the unification of knowledge management and grid technologies and defining an integrating architecture for knowledge extraction and delivery based on data grid services. The knowledge layer of the generic framework uses the basic data grid services and defines an additional layer to implement the services for knowledge extraction, presentation and delivery from disparate
and heterogeneous databases stored on globally connected computers where each node can be a sequential or a parallel machine.

The knowledge layer of the generic framework consists of knowledge-based component through which, content is turned into something more than just a collection of data. The knowledge-based component is implemented through an agent. A knowledge-based system is having two essential components: A knowledge base and an inference engine. The knowledge base is a repository of domain knowledge and meta knowledge. The inference engine is a software program which infers the knowledge available in the knowledge base.

Several mechanisms are available to model and realize knowledge-based system. One of these mechanisms is fuzzy logic. The agent resides in the knowledge layer incorporates fuzzy set theory and fuzzy logic. Fuzzy logic representations are more intuitively satisfying than classical Boolean logic. It is more precise and compact compared to classical rule based representations.

Fuzzy Logic Controller (FLC) implements the concepts of fuzzy set theory and fuzzy logic and is being widely and successfully applied in different areas. FLC can be considered as a knowledge-based system incorporating human knowledge in their knowledge base through fuzzy rules and fuzzy membership functions. It used to provide an intelligent assistance to decision makers for data analysis. At present, the framework applies fuzzy set theory and fuzzy logic to realize knowledge-based system. Agents provide open, dynamic and extensible environment and therefore able to accommodate new capabilities and features. Therefore, in the future, several other approaches can also be integrated to the knowledge layer, as the framework uses the agent paradigm to offer knowledge extraction, presentation and delivery services. New agents will be added in the knowledge layer which may follow the other artificial intelligence approaches to implement knowledge-based systems such as neural network, neuro-fuzzy architecture, genetic algorithms, data mining methodologies and many more.

**3.2.4 Application Layer**

The application layer is the highest layer of generic framework and accesses all other layers for resources where and when required. The application layer consists a specific
agent called an interface facilitator agent. The user first sends requests through the interface facilitator agent which provides a user interface to client applications and users. The interface facilitator agent enables users to send a request and receive a response to and from the data grid environment. The application layer may access the knowledge layer or middleware layer to use core, data and user level services.

The interface facilitator agent is used to connect end users or external systems together with the multi-agent system. It reacts towards the different requests made by the end user or external system and translates these commands into agent understandable requests, and sends them to the appropriate agents. The interface facilitator agent may also coordinate with different collaborative agents (resides in middleware layer) in order to access the data grid services. It may also coordinate with agent resides in knowledge layer to extract knowledge from data or information gathered from various data resources resides in a data grid environment. Furthermore, the interface facilitator agent sends back the response to the client application and users about the status of their request.

Basically, the application layer of the generic framework allows access to remote software and digital libraries, and provides overall management of all applications running. This layer consists of several data intensive applications like Scientific Research applications, E-business applications, Educational Applications, Healthcare Applications, Genomics Applications, High Energy Physics Applications, Earth Science Applications and many more.

### 3.3 Conclusion

This chapter covers the detailed aspects of generic framework developed for the integration of multi-agent knowledge-based system and distributed database grid. The generic framework is designed and developed as a layered architecture. Each layer in the framework offers some set of services and activities to higher level layer. It also uses the specific set of services and activities offered by lower level layer. The generic framework offers the various types of data grid services like metadata management, data access & integration, data delivery and many more. It also offers knowledge level services by realization of agents. Currently, to offer knowledge level services, fuzzy set theory and fuzzy logic are implemented. In the future, several other artificial intelligence techniques may be incorporated by the genetic framework as it follows agent oriented paradigm. This
chapter discusses about the overall architecture of the generic framework in detail. The next chapter will discuss the methodology used to implement the generic framework in detail. It will elaborate the implementation details of the several component resides in the framework.

References

