CHAPTER 9

IMPLEMENTATION

The proposed MAS based SCM system is being designed with aim of tackling the bullwhip effect successfully along with product availability and maximized customer satisfaction. The causes of the bullwhip effect are being controlled by predefined strategies. These strategies are being designed with the help of the case-based reasoning approach executed by the intelligent agent. The CBR-BDI intelligent agents are capable of learning from the working environment. There exist multiple tools which support the BDI architecture of the intelligent agents like ADE, Aglets, JACK, JADE, Jatlite etc. The JADE tool is totally written in Java language which supports the object-oriented features.

9.1 JAVA AGENT DEVELOPMENT ENVIRONMENT

The proposed MAS based SCM system consists the intelligent agents which having the BDI architecture along with the case-based reasoning. The selection of the intelligent agent development tool should be based on the following factors as given below:

- Easy to operate
- Easy to program
- Support BDI architecture
- FIPA-ACL compliant
- Support agent life cycle
- Flexibility
- Robust
- Modular approach

The JADE tool supports all these features in the agent-oriented programming (AOP) and agent-oriented software engineering (AOSE). JADE assists the programmers to develop the multi-agent systems. It consists following components as given below:
• **Runtime environment**  
  It enables the developer in which the agents can be constructed and operated on the specified host.

• **Library of classes**  
  The JADE tool contains the programmers to use the library of the classes to develop their agents.

• **Collection of graphical tools**  
  It facilities the collection of the graphical tools which allows administrating and monitoring the movements of running agents.

These components indicate the powers of the JADE tool which generate the efficient mechanism to work with intelligent agents.

### 9.1.1 Containers and platforms

In JADE tool, the active running instance of the JADE runtime environment may be known as the container. The container may consist several agents. The collection of running active containers is known as the platform.

![JADE containers and platform](image)
There may be specific environment running in the initial phase may be known as the main container. The main container remains the active in a JADE tool. Every agent is registered with a unique agent identifier. An AID object includes a globally unique name plus a number of addresses. The predefined format of the agent name is \texttt{<nickname>@<platform-name>} so that an agent called Demand agent running on a platform called SCM will have \textit{Demand@SCM} as the globally unique name. The addresses included in the AID are the addresses which are used during the communication between those agents of the other platforms.

9.1.2 Main packages

The JADE tool is composed of the following main packages as given below:

- \texttt{jade.core} implements the mechanism which acts as the kernel of the system. This package consists the Agent class that which is required to extend to construct the intelligent agents. The agent behavior is being defined by the Behaviour class contained in \texttt{jade.core.behaviours} sub-package.

- \texttt{jade.lang.acl} supports the mechanism to process Agent Communication Language according to FIPA standard specifications.

- \texttt{jade.content} package consists of multiple classes to support user-defined ontologies and content-languages.

- \texttt{jade.domain} package has the Java classes having the capability of representing the Agent Management entities such as AMS and DF etc.

- \texttt{jade.gui} package contains a collection of the classes which are used to generate the Graphical User interfaces to define the Agent-Identifiers, Agent Descriptions and ACLMessages, etc.

- \texttt{jade.mtp} package contains the Java interface which implements the Message Transport Protocol for the message passing.

- \texttt{jade.proto} package support the implementation of various interaction protocols such as fipa-request, fipa-query, fipa-contract-net and fipa-subscribe etc.

- \texttt{jade.wrapper} package provides binding of the JADE higher-level functionalities which support external Java applications to launch JADE agents and agent containers.
These packages are being imported during the process of the constructing the intelligent agent based on the BDI architecture.

9.2 JCOLLIBRI TOOL

The case-based reasoning approach is mainly used for enhancing the learning capabilities in the intelligent agents. There are multiple tools that can be used to use the case-based reasoning approach in the intelligent agent based application. The jCollibri is one of the tools which support the mechanism to enable users to take the decision with the help of the case-based reasoning feature. The major purpose of using jCOLIBRI is to afford an indication stage for designing the CBR applications. This tool facilitates the programmers to follow the whole CBR cycle. The jCOLIBRI support the designing of large scale commercial applications. It is efficient in quick prototyping of CBR systems and budding appliances for authentic situations.

9.2.1 Main features

The case-based reasoning approach provides the mechanism for utilizing the past experiences during the problem solving phenomena. The jCOLIBRI includes the following features which power the developer during the designing process of the case-based reasoning as given below:

- **Persistence layer**
  This tool provides a comprehensible boundary between the medium containing the cases and the in-memory organization of the cases.

- **In-memory organization of the cases**
  This tool support to organize the cases in case-based using various indexing structures such as linear lists, trees, maps, etc.

- **Retrieval methods**
  This tool consists multiple efficient retrieval methods like Nearest Neighbor method. These methods compute the global similarity functions to evaluate composite attributes and local similarity functions to evaluate simple attributes.

- **Reuse and revision methods**
  This tool includes uncomplicated techniques to facsimile the explanation and information from stored cases and their cases.
• **Efficient case maintenance**

It allows the developers to diminish the dimension of the case base for efficient case maintenance. The evaluation methods measure the performance of a CBR application.

• **Visualization**

This tool arranges the distance between cases and debugs the robustness of the similarity measure.

For assisting the developer in developing the CBR applications, the jCOLIBRI offers different retrieval strategies with more similarity methods. It consists various adaptation and maintenance components plus visualization of the case base. These methods provide efficient usage of the past experiences of solving the problems faced in the real-time scenario.

### 9.3 PROPOSED CBR-BDI INTELLIGENT AGENT TOOL

The CBR-BDI intelligent agents are made of two components- intelligent agent’s BDI architecture and CBR approaches. There are not any single tools available which are capable of constructing the CBR-BDI agents. Hence to construct the CBR-BDI agents for running the supply chain activities there are following two components are imported into the Eclipse IDE as given below:

#### 9.3.1 EJADE

The EJADE is considered as the Eclipse plug-in that which import the JADE into the Eclipse IDE to launch agent platform and to deploy agents. The plug-in manages the Java classpaths and associates JADE Java agents’ specifications to Eclipse debugging framework. It helps in the process of making debugging agents. It performs following tasks as given below:

• **Deploy JADE agents**

The developers deploy the JADE agents through choosing the intelligent agent files anywhere in your project and deploy them. There is no necessity to worry about classpath, user lib etc. The successful agent deployment provides the environment for creating the intelligent agents and supports the all activities in the environment.
• **Debug JADE agents**
  The developer can deploy JADE agents by just selecting the option of the EJADE debug.

• **Launch & shutdown JADE platform**
  This plug-in provides the facility of lunching and shut down the JADE in standalone mode.

• **Indicate modified platform parameters**
  The programmers can set the EJADE preferences setting to define the intelligent agents’ properties.

These features make the developers to import the EJADE plug-in into the Eclipse IDE for designing the CBR-BDI intelligent agent based real time applications for various domains like business, teaching and medical science etc.

### 9.3.2 COLIBRI studio

It is the accomplishment of the top level of the COLIBRI platform. The main purpose of using this studio is to afford the visual builder tools required to generate CBR systems without dealing unswervingly with the source code. It consists of the jCOLIBRI framework and enables the composition of its CBR components.

The COLIBRI Studio can be easily integrated into the Eclipse IDE. With the help of this integration, the CBR projects may be managed easily as required libraries are automatically configured. There are following components of COLIBRI Studio which are mostly used in designing CBR applications as given below:

• **Case designer**
  The Case Designer component is used to specify the structure of the cases. The case structures contain the fields like Description, Solution, Result and Justification etc. Each field is collected of numerous attributes, which can be also single or multiple. These fields have great impact on whole system output during its working. It provides the mechanism to generate the case for particular intelligent agent in the proposed MAS based SCM system. This component defines all details of the particular case structure such as linear and cached linear etc.
- **Case base selector**
  This component specifies the in-memory organization of the case base for loading the cases into persistence media. It can create the source code after selecting the case base selector among multiple case base organization.

- **Plain text connector configuration**
  This component indicates the text file which consists the cases in the case base. Every line indicates the particular case and each case field are being separated by the comma.

- **Similarity**
  It defines the methods for computing the similarities between the cases stored in the case base. These similarity functions are being applied to compute the local and global similarity between the cases on the basis of case attributes.

These components are core of COLIBRI Studio which provides the greater support during the process of designing CBR application.

### 9.3.3 Eclipse IDE

The Eclipse is one of mainly used integrated development environments which is used to design the application in various programming languages. It helps the programmers to build its application smoothly and rapidly. This IDE provides all functionalities to the developers for constructing the real-time applications. This IDE is composed of multiple components. It consists the Client Platform (RCP) components for integrations of other platform with this IDE. This tool is capable of integrating with other tools and applications. In Eclipse Platform, the programmer can write the Java code through Java development components (JDT) and C/C++ code through C/C++ development components. Combination of JDT and CDT make this IDE to provide the mechanism to write the Java and C++ code.

### 9.3.4 Running the proposed tool

To construct the CBR-BDI intelligent agent, we are going to embed the JADE and jColibri tool at one place. The JADE tool is used for developing the intelligent agents and jColibri tool is used to integrate the case-based reasoning approach in the working of the intelligent agents to learn from external environments.
On running this proposed tool, this tool provides the mechanism of using the JADE and jColibri tool. As shown in figure 9.2, there are symbols which highlight the way to switching the simple Java application into CBR-BDI intelligent agent based applications.

![Figure 9.2 Eclipse IDE](image)

There are following modes to work with this proposed tool as given below:

- **Simple java application**
  In this mode, simple Java applications can be created without including the jColibri and JADE tools during the process of designing the Java applications

- **JADE based application**
  In this mode, the developers can create the intelligent agent based applications. In this mode, the CBR components are being toggled off.

- **CBR based project**
  In this mode, the developers can create the case-based reasoning based applications. In this mode, the JADE components are being toggled off.
• **CBR-BDI applications**

In this mode, the developer can create the case-based reasoning based intelligent agent applications. In this mode, the CBR-BDI intelligent agents can be developed. These modes indicate the working of the proposed tools for creating the CBR-BDI intelligent agents involved in the supply chain management system with the aim of controlling the bullwhip effect.

### 9.4 CONSTRUCTING CBR-BDI INTELLIGENT AGENT

The CBR-BDI intelligent agent plays very important role in the proposed MAS based SCM system for tackling the bullwhip effect occurred in the SCM activities. The working of the CBR-BDI intelligent agent is being divided into three phases in figure 9.3 as shown below.

![Figure 9.3 Life cycle of CBR-BDI agent](image)

There are mainly following three phases of the CBR-BDI agents working as given below:

- PreCycle
- Cycle
• Post Cycle

Every phase of CBR-BDI intelligent agent consists some predefined actions to be performed during that particular phase.

9.4.1 Precycle phase

This phase of the CBR-BDI intelligent agent initializes the case base associated with its working. For this purpose, it defines the case base connector whose task is to connect the case base with CBR application. The main tasks of the precycle phase are being shown in figure 9.4 given below.

![Figure 9.4 Precycle phase of CBR-BDI Intelligent agent](Image)

9.4.2 Cycle phase

This phase of the CBR-BDI intelligent agent performs the actions related to case base after defining the case base connector and loading of the case base. It performs following actions as given below:
- Specifying CBR Query
- Specifying Similarity functions
- Calculating the similarity
- Retrieving the cases
- Displaying the cases

The flow of the main tasks of the cycle phase is being shown in figure 9.5 given below.

![Figure 9.5 Cycle phase of CBR-BDI Intelligent agent](image)

### 9.4.3 PostCycle phase

This phase is the last phase of the working of the CBR-BDI intelligent agent’s working. This phase consists the task of retaining of the cases in the associated case base. It generates the efficient case indexing for fast retrieval. This phase directly decides the
performance of the CBR application. The flow of the main tasks of the cycle phase is being shown in figure 9.6 given below.

![Figure 9.6 PostCycle phase of CBR-BDI Intelligent agent](image)

These phases are the main phases of the CBR-BDI intelligent agent’s working. Each phase is performing set of the specific tasks during its execution. For each phase, the particular phase consists the different type of the variables such as the input, output and global variables. The input and output variables are being used to hold the user inputs and outputs during its operations. The global variables have unlimited scope and can be accessed through whole CBR application.

### 9.4.4 Defining the case structure

To define the case structure is the first major task of the process of creating CBR-BDI intelligent agent. We have already discussed the case structure of various intelligent agents. Let’s take the example of creating the Demand CBR-BDI agent. The case structure of the Demand CBR-BDI intelligent agent is described in the table 8.1. The case designer component helps in defining the attribute names along with its type. The case structure is being stored in caseStructure.xml file. The figure 9.7 illustrates the process of defining the case structure of the Demand CBR-BDI intelligent agent as shown below.
9.4.5 Selecting the case base organization

The next step of this process is finalizing the case base organization between available compatible case base organizations. There are following compatible case base organizations as given below:

- Cache Linear Case base
- Indexed Linear case base
- Linear case base

The figure 9.8 illustrates the process of selecting the case base organization of the Demand CBR-BDI intelligent agent as shown below.
Figure 9.8 Selecting the case base organization

9.4.6 Configuring case base connector

After selecting the case base organization, this step is setting up following values as given below:

- Path of the case base
- Class consisting the case description fields
- Class consisting the case solution fields
- Delimiter between the case fields

All this information is being stored in the particular file named as plainTextConnectorConfig.xml file. This step specifies the path of this file and data file containing the case base associated with particular CBR-BDI intelligent agent. The figure 9.9 illustrates the process of selecting the case base organization of the Demand CBR-BDI intelligent agent as shown below.
9.4.7 Setting up the similarity

The major task of this process is defining the similarity measurement mechanism used in the case-retrieval phase. This phase consists following factors as given below:

- Similarity Function
- Weight

There exist various types of similarity functions which may be applied to particular attribute depending on its data types. These functions calculate both local and global similarity between the different cases stored in the case base. The weight may be defined the level of the importance in the similarity measurement phenomena. This information is being stored in the file named as similarityConfig.xml file. The figure 9.10 illustrates the process of selecting the case base organization of the Demand CBR-BDI intelligent agent as shown below.
The proposed MAS based SCM system consists multiple intelligent agents designed for specified purposes such as demand forecasting, retaining, sale and inventory control etc. So before implementing the proposed MAS based SCM system, it is required to draw its UML diagram to explain its internal working. To illustrate the working of the proposed system, following diagrams are being required to draw as given below:

**9.5.1 Class diagram**

This diagram shows the all classes designed for various CBR-BDI intelligent agents involved in MAS based SCM system. The class diagram represents the static view of the proposed MAS based SCM system. It is very useful in picturing, explaining and documenting various features of a system. This diagram builds the scenario of the executable code of the proposed application. It sculpts the motionless outlook of an application before designing of the proposed MAS based SCM system. The class diagrams can directly map in the object oriented languages and thus widely used at the
time of construction. So in a short, class diagrams are applied for following reasons as given below:

- Explaining the static view of the proposed system.
- Describing the collaboration among the elements of the proposed system.
- Explaining the functionalities to be performed by the proposed system.

The figure 9.11 shows the class diagram of proposed MAS based SCM system as given below.

Figure 9.11 Class Diagram of Proposed MAS based SCM System
9.5.2 Use case diagram

To capture the dynamic aspect of proposed MAS based SCM system, the use case diagram is being drawn. Mainly this type of the diagram is applied to assemble the necessities of a proposed system including interior and exterior controls. These necessities are regularly design requirements. It is designed for the purpose of analyzing functionalities. It uses the concept of the actor. There are following purpose of drawing the Use case diagrams as given below:

- To collect necessities of the proposed system.
- To generate an exterior analysis of the proposed system.
- To recognize exterior and interior aspects influencing the proposed system.
- To demonstrate the interacting between the necessities as the actors.

The figure 9.12 shows use case diagram of proposed MAS based SCM system as given below:

![Use Case Diagram of Proposed MAS based SCM System](image_url)
9.5.3 Activity diagram

The activity diagram is used to demonstrate the various activities occurring in the proposed system. It also consist the message flow from one activity to another. This type of the diagram represents the dynamic view of the system. It looks like as an algorithm but not same as it. The basic purposes of drawing the activity chart can be described as below:

- To illustrate the activity flow of the proposed system.
- To explain the series from individual activity to another.
- To depict the equivalent, divided and synchronized flow of the proposed system.

The figure 9.13 shows an activity diagram of MAS based SCM system as given below:

![Activity Diagram of Proposed MAS based SCM System](image-url)
9.6 EXECUTING PROPOSED MAS BASED SCM SYSTEM

The proposed MAS based SCM system is being designed for running the supply chain activities with aiming of tackling the bullwhip effect properly. At the first stage the intelligent agent manipulates its plan regarding the initial stage of the problem called as the beliefs. For example the Demand CBR-BDI agent has the responsibility of generating the more accurate product demand. With the help of the case-based reasoning approach, the individual CBR-BDI agent generates its plan regarding to encounter the problem. This plan is based on the CBR query. The CBR query specified the all attributes of the problem faced by the intelligent agent. The resulted plan of the CBR-BDI intelligent agent helps in deriving the actions taken during the problem-solving phenomena. The CBR query of the Demand CBR-BDI intelligent agent is shown in figure 9.14 displayed below.

![CBR Query of CBR-BDI Intelligent agents](image)

Figure 9.14 CBR Query of CBR-BDI Intelligent agents
The next step of working of the reassign the configuration of the weight value and associated similarity function along with different CBR query attributes. The figure 9.15 illustrates the reconfiguring the similarity of the Demand CBR-BDI intelligent agent as shown below.

![Configure Similarity](image)

Figure 9.15 Reconfiguring the similarity

This step provides one more option to change the similarity in the process of preparing the plan for solving the particular problem. It provides more flexibility in the problem solving phenomena. One more advantage of the case-based reasoning is that it provides the way of utilizing the past knowledge associated the cases stored in the case base. This facility is known as the case adaptation. In case of the Demand CBR-BDI intelligent agent, there exists direct the relationship between the seasonal patterns and demand forecasting values. There is also another relationship between the forecasting values and its associated method. The figure 9.16 illustrates the case adaptation of the Demand CBR-BDI intelligent agent as shown below.
After setting the weight values and its associated similarity function along with case adaptation facilities, the each CBR-BDI intelligent agent retrieve the similar cases for preparing its plans. It displays the value of the similarity between the new case and the stored case. The number of the retrieved cases is directly dependent on the values of K specified in the figure 9.15. The K-Nearest Neighbor algorithm is being applied in the process of retrieving the similar cases regarding the attribute values of the CBR query. For example the value of K is 3 in the given CBR query. The result shows one of 3 retrieved similar cases along with its similarity value. The similarity value is made of following two parameters as given as below:

- Local similarity
- Global similarity

The result of the CBR query shown in figure 9.14 is being shown in the figure 9.17 given below.
Figure 9.17 Case retrieval

The figure 9.17 illustrates the result of the CBR query. The case-based reasoning provides the facility of revising the generated result of the case retrieval phase. The case revision phase enables the users to modify the values during this phase. This factor prevents the error propagation from past experience to the future solution.

The case revision phase should be performed with more concentration. It is the last chance to stop the error propagation to the next phase. This phase helps the CBR-BDI intelligent agents to build their plans regarding their desires. The figure 9.16 illustrates the case adaptation of the Demand CBR-BDI intelligent agent as shown below.
After revising the proposed solution, it produces the final solution in the form of the plans. The plans should be stored in the case base for utilizing them in the future. The case indexing is one major issue associated with the case retrain phase. The preindexing process is being found very consuming process for applying the nearest neighbor algorithm in the case retrieval phase. The induction retrieval algorithm is not more suitable for this proposed MAS based SCM system. This algorithm is not efficient in handling the inadequate and inaccurate data in the business activities. The case indexing makes the case retrieval phase very quick in its processing. It helps the proposed MAS based SCM system to overcome the main problems faced in traditional MAS based SCM system. The CBR system goes down slow with growth in the size of case base.

After executing the CBR cycle the CBR-BDI agent has prepared its plan to perform the actions for solving the upcoming problems, the next phase of executing these actions is being performed along with the support of the case-based reasoning facility. The case retraining for the Demand CBR-BDI intelligent agent is shown below in the figure 9.19.
The Demand CBR-BDI intelligent agent takes the historical data and based on the selected method it calculate the product demand values. These steps are predestined steps of working of the every CBR-BDI intelligent agent in the proposed MAS based SCM system.