CHAPTER 3

ARCHITECTURE OF M-MASIR

3.1 INTRODUCTION

Information retrieval is one of the activities of obtaining information relevant to one’s need, from a collection of information resources. Efficient information retrieval is one of the most important and active area of research and has been studied widely in the context of e-service applications. A variety of approaches have been proposed in literature for mobile agent based information retrieval. The mobile agent is an active entity that can migrate autonomously through a computer network and resume execution at a remote site. By doing this, it can access the required resources and perform a task on behalf of its user. The literature survey reveals that most of the works in this area have used only certain aspects of the mobile agent system for information retrieval. However, work related to the various issues the mobile agent faces during its travel is rather limited or not sufficient. Many of the techniques have not been tuned to provide a system that reacts to the possible blockages of the mobile agent at different stages of its travel. Hence, in order to strengthen such a general mobile agent system with specific features that are required for successful and efficient information retrieval, an architecture called M-MASIR (Multifaceted Mobile Agent System for Information Retrieval) is proposed. The architectural diagram of the M-MASIR is shown in Figure 3.1.
Figure 3.1 Architecture of a Multifaceted Mobile Agent System for Information Retrieval (M-MASIR)
The overall functioning of the M-MASIR is as follows. Initially, the client sends a request to the agent service center for an information retrieval service. The agent service center creates a mobile agent based on the client’s request and dispatches it to the agent monitor of a region. Multiple agents are deployed if the agents are to be operated in a multi-region environment. The mobile agent visits the nodes either in a sequence that is identified by the agent service center or dynamically in that region. At each node, initially the mobile agent requests accommodation. Once accommodated the mobile agent is subjected to a security check. On clearing the security check, the actual information retrieval process is done. After retrieving the required information from a particular node the mobile agent migrates to the next node, and continues the same process. On task completion, the mobile agent migrates back to the agent service center with the retrieved data. Simultaneously, the same process is performed in other regions also. After receiving the data from each region, the agent service center consolidates it and sends it back to the client in the required format. The following sections explain the functionalities of each module of the architecture of M-MASIR.

This architecture consists of two phases, namely, the pre-processing phase, and the execution phase. The objective of the pre-processing phase is to create mobile agents that are intended to fulfil the requirements of the client. The pre-processing phase includes a travel plan for the initial migration (in the case of a dynamic itinerary) of the mobile agent. The output of the pre-processing phase is a mobile agent that is equipped with an initial travel plan and the security requirements. The execution phase focuses mainly on the successful task completion of the mobile agent at each node, even in the face of possible blockages. During its arrival at each node, the mobile agent is provided with the necessary resources that meet its requirements, in order to perform its function.
3.2 PRE-PROCESSING PHASE

Developing an appropriate mobile agent is one of the critical tasks in the process of information retrieval. In the case of a static itinerary, this development involves generating the code based on the user’s requirements, determining the nodes that hold the relevant information, and generating a travel plan to visit these nodes. On the other hand, for a dynamic itinerary, this phase determines the identity of a node for the initial migration only. The goal is to design an adaptable mobile agent that is successful in its task completion with higher benefits and minimum overhead. The following sections discuss the functional components that are involved in the development of a mobile agent.

3.2.1 Client Request

In the pre-processing phase, a client can get a service from the number of available service providers. A service provider that provides agent related services is referred to as an Agent Service Center (ASC) in M-MASIR. A client can forward a service request query to more than one of these agent service centers and select the one that satisfies the requirements. This research work considers two types of information retrieval service requests. One is to retrieve the documents, that are related to the research topic of a person, from the members of a research group. The target documents should contain the words that match the keyword given by the user. Another task is in a medical environment, to retrieve the related images that match the image given by the user. In both the cases, the nodes that hold the related documents or images are distributed geographically in the distance ranges between 1 kilometer and 620 kilometers.
The request is expected to match the form that is required by the agent service center. In the same way, the client shall record the format in which the output is expected. For example, a client wants to collect the documents that are related to the keyword “mobile agents”. In order to register this request, the client should be available with a suitable query format. The format should also include convenient ways for the user to represent other constraints, such as task completion time, amount of the required output, and coverage area. Once the client submits the request, the connection between the client and the agent service enter is terminated.

### 3.2.2 Agent Service Center

Agent based information retrieval is one of the services provided by the agent service center. An ASC accepts the keywords given by the user, which are meant to match with the information present in various nodes of the network. The agent service center is responsible for designing a mobile agent by assigning a task and providing the itinerary plan. The agent service center must include all the client’s requirements at the time of designing the code. In addition, it dispatches the mobile agent and receives it back on task completion. Multiple mobile agents are created in order to deploy them in parallel, if they are to operate in a multi-region environment. The components used by the agent service center in order to perform these functions are, the task allocator, itinerary planner and mobile agent dispatcher.

The task allocator focuses mainly on the efficient coding of the mobile agent. The code for information retrieval takes one pattern from the given query at a time and matches it with the information available at various nodes that are distributed geographically. It includes the format for the user input and output, the processing steps to be executed at each node and a
container to carry the output. The code for the mobile agent is backed with a travel plan that is provided by the itinerary planner.

The itinerary planner makes use of one of the two possible travel plans, namely, static and dynamic. For a static travel plan, it is the responsibility of the agent service center to figure out the nodes, from which it has to retrieve the related information. This process must be done before the start of the migration. In the case of a dynamic travel plan, initially the agent service center requests the monitoring agent residing at the location server. The objective of this request is to select the next optimal node to visit. The task of the itinerary planner is complete, once the mobile agent is assigned the address of the next node (dynamic) or the addresses of the node list (static). The travel plan is embedded with the code developed by the task allocator to form a mobile agent that is ready for migration.

The mobile agent dispatcher receives the mobile agent with the code and travel plan, and dispatches it to the first node as per the travel plan. Once the mobile agent is dispatched, the link between the mobile agent and the agent service center is terminated. This connection is reestablished, when the mobile agent migrates back to the mobile agent dispatcher after its task is successfully completed. The mobile agent dispatcher receives the successful mobile agent, collects the retrieved data, and this data is dispatched to the client in the required format.

3.3 EXECUTION PHASE

In the execution phase, the mobile agent is supported by a node to perform its function, by providing the necessary resources to accomplish the goal. During its travel, the execution of a mobile agent is viewed at two levels: region-based agent platform and node-based agent platform. The main function of the node-based agent platform is to provide the execution
environment for the mobile agent at the node level. The region-based agent
platform focuses mainly on the execution of the mobile agent at the multi-
region level.

3.4 REGION-BASED AGENT PLATFORM

In the event of an agent deployment, for a time constrained task, multiple agents are created. In order to properly organize the dispatch of these agents to the nodes, the nodes that are distributed geographically are grouped under regions. This will ensure that the nodes are not visited repeatedly by more than one agent. A region consists of a set of nodes that have the same authority, and separated from the network based on its location. The environments in which the multiple mobile agents operate are referred to as a multi-region environment. The region-based agent platform finds its functionalities based on the way in which the mobile agent operates in a multi-region environment. The architectural view of the region-based agent platform is shown in Figure 3.2.

3.4.1 Agent Monitor

In the multi-region environment considered for this work, each region is assigned with a dedicated server referred to as the Agent Monitor (AM). The responsibility of the AM includes, data storage, location management and fault management. By data storage this architecture means the logging of data that is retrieved from each node and sent by the mobile agent. After completing its task at a node, the mobile agent is required to send the data collected from that node, for example, a set of documents, to AM. In turn, the AM logs this data as replication. The agent monitor uses this replica to recover a failed mobile agent.
Figure 3.2 Architectural Diagram of M-MASIR: Region-based View
The location management deals with the current location of a mobile agent in its region and is managed by the location updater. During its roaming, a mobile agent is required to update its current location with the location updater, after every migration. The location updater records the current location update sent by a mobile agent during its travel. The fault manager governs the fault management that deals with the recovery process of the failed mobile agent. As the mobile agents are operated in an open environment, they are subject to failure due to link, application or node failure. The replication is achieved by clone creation that plays a vital role in the agent failure recovery process, and is handled by a clone creator. The agent monitors of each region are connected together through the Internet, and are capable of communicating and interacting among themselves. This feature permits the agent monitor to share the information it holds with the agent monitors of other regions. As a result of this information sharing process, the failure of the AM does not result in agent blocking or information loss.

3.4.2 Location Server

The location server is a centralized server that monitors the movement of the mobile agent globally, in the absence of a AM. The services of the location server are common (centralized) to all the nodes in the entire network, whereas, those of the AM are restricted (distributed) to the nodes of a particular region. The foremost services like, failure detection and recovery, location update and information sharing will be done globally. A prominent feature of the location server is that it supports mobile agent tracking. A mobile agent that is roaming in one region can utilize the service of the location server, in order to track the mobile agents of other regions. The request for intra-region tracking is handled by AM and that for inter-region is handled by the location server. This provision encourages the mobile agents to communicate and share the information they are accommodated with. This
information sharing process ensures that the data collected from one region is available as replication in all regions. Through this, failure recovery in the event of a mobile agent failure is ensured. That is, the failure of a mobile agent of one region does not result in information loss or mobile agent blockage. The mobile agent can be recovered from the replication that is available at the nodes of other regions. Another specific feature of the location server is the directory service support. The directory server maintains the details required by a mobile agent regarding the nodes that are associated with this service. The current status of a node, the type of information a node possesses, and the cryptographic keys that are associated with a node’s security are some of the services required by the mobile agent. More importantly, these services are utilized by a mobile agent in order to select dynamically the next node for migration.

3.5 NODE-BASED AGENT PLATFORM

The node-based agent platform deals with the operation of a mobile agent within a region. This module manages the accommodation process, security requirements, failure recovery and obviously the information retrieval process of a mobile agent within a node. After selecting an optimal node using the directory services of AM, the mobile agent migrates to that node. A mobile agent may not be able to perform its function at a node or migrate to the next node successfully for various reasons. The reasons identified for the mobile agent system presented in this thesis are, a node running short of memory to accommodate a mobile agent; the mobile agent being attacked by other agents; and mobile agent failing due to node failure. A node is a physical machine connected to other nodes through a network, and is responsible for providing resources through an execution environment to the visiting mobile agent. The initial resource required by a mobile agent at a node is memory for accommodation. The architectural view of the node-based agent platform is shown in Figure 3.3.
Figure 3.3 Architectural Diagram of M-MASIR: Node-based View
3.5.1 Memory Allocator

The principal function of the memory allocator is to offer a memory allocation algorithm. This algorithm is invoked in a situation where the mobile agent find difficulties in its accommodation. If the problem occurs due to the excessive accommodation of mobile agents, then this algorithm is invoked. This algorithm identifies and removes the mobile agents that are not in use, but still occupy the memory in large numbers. As a result, it is possible for a node to accommodate more number of mobile agents for execution. By doing this, the blockage ratio of the mobile agent due to non availability of memory is reduced.

3.5.2 Security Manager

Once a mobile agent is accommodated, the next requirement is the execution environment, in which the mobile agent executes. However, a mobile agent is allowed to enter the execution environment only after getting permission from the security manager. The security manager of the presented architecture M-MASIR, intensifies its focus on an attack that is performed by a mobile agent against the others. This is achieved by imposing a three level security check on an incoming mobile agent. Based on this architecture, a secure mobile agent comes with the required information about the agent, which is encrypted by the sending node (sender). The information includes the identity of the mobile agent, the identity of the receiving node (receiver) and the size of the mobile agent.

The receiver of the mobile agent determines the required details by decrypting them. The authentication verifier compares the identity of the mobile agent; the agent size verifier cross checks the size of the mobile agent; and the itinerary checker verifies the identification of the receiver that is selected by the sender.
The directory service is requested for cryptographic key management that plays a vital role in all these encryption, decryption and verification processes. Only a mobile agent that clears this security check is allowed to enter the agent execution environment of a node. Security failure or mismatch at any level results in the immediate termination of the mobile agent.

### 3.5.3 Mobile Agent Executor

The component in which the actual information retrieval process takes place is the mobile agent executor. All the remaining modules of this phase will cooperate with this component to complete the task. In order to perform its task, a mobile agent that enters the execution environment, confers with the stationary agent residing at that node. After checking the validity of the request, the stationary agent seeks the database concerning the required service. The stationary agent hands over the retrieved information, i.e., a set of documents or images according to this architecture, to the requested mobile agent. The mobile agent sends the retrieved information to AM, and migrates to the next node with the code but not the data. In the context of the region-based agent platform, the mobile agent creates its clone for future procurement, and is stored in a data log as replication. It is the employment of the fault manager to forward a mobile agent from the data log as the failure recovery process. At the same time, in a node-based agent platform, the recovery of the failed mobile agent is done through AM. The agent monitor forwards the clone of a mobile agent as recovery to the node from where the mobile agent actually failed.

### 3.5.4 Travel Planner

The aim of the travel planner is to select the next optimal node for migration among the possible nodes. A mobile agent can select the next optimal node in two ways. One is by sending the migration request to all its
immediate neighbor nodes. This request may include some constraints that the mobile agent is interested in for migration. After receiving and analyzing the response, the mobile agent will select the next optimal node and migrate to that node. Another way of selecting the optimal node is by requesting the monitoring agent that monitors the directory services available at the location server. The directory service holds the information with regard to all the nodes in a region. The monitoring agent determines a node with the most recent update, and the identity of that node is forwarded to the requested mobile agent. On receipt of this information, the mobile agent migrates to that node and continues its execution there. The communication between the node and the location server or AM is implemented by using the communication manager. The communication manager provides a reliable communication protocol in the event of mobile agent communication.

3.5.5 Output

The information, i.e., a set of documents, retrieved from each node the mobile agent visits in a particular region, is available at the corresponding AM. This is applicable to all the other regions also. On the other hand, as a result of the information sharing process, the data available at one AM is available as replicas in all other agent monitors. On task completion, the mobile agent of each region migrates back to the AM and then to the agent service center. The output the agent service center receives is the aggregated data that is retrieved from the various pre-defined nodes if the static travel plan is used. In the event of a dynamic travel plan, the output is a bundle of data retrieved from the remote nodes that are selected dynamically by the mobile agents on the fly. The data retrieved by each mobile agent is consolidated and formatted according to the user’s requirement and is delivered to the user.
3.6 SUMMARY

- M-MASIR encompasses two distinct phases, namely, i) the pre-processing phase and ii) the execution phase. The intended purpose of the pre-processing phase is to enumerate the preliminary actions desired by a mobile agent system in the creation of appropriate mobile agents.

- In order to retrieve the required information efficiently from the various sites that are distributed geographically, a multifaceted mobile agent system M-MASIR is proposed. The system M-MASIR with novel capabilities ensures efficient and reliable information retrieval.

- The execution phase of the M-MASR consists of four functional components such as, i) the memory allocator, ii) security manager, iii) agent executor and iv) optimal node selector.

- The memory allocation algorithm invoked by the memory allocator removes the mobile agents that occupy the memory unnecessarily, and provide more space for the incoming mobile agent’s accommodation.

- The novelty of the security manager is to secure the mobile agents from other mobile agents with the intention of tailgating attacks. The tailgating attack detection is achieved by the agent size verifier.

- The itinerary checker ensures the security of the dynamically selected itinerary by utilizing the services of the directory server that are exclusively allocated in M-MASIR.
• The fault manager module of the agent monitor, which is specifically introduced in this architecture M-MASIR, manages the recovery of the mobile agent in the event of its failure.

• The optimal node selector introduced in M-MASIR, selects the next optimal node dynamically for the efficient migration of the mobile agent with the help of a specially allocated location server.