Fault-Tolerant Mobile Agent based on Group Communication

Naohiro Hayashibara

School of Information Science,
Japan Advanced Institute of Science and Technology

15th Feb. 2001

Keywords: Mobile Agent, Fault-Tolerance, Group Communication, Majority Voting, Two Phase Commit.

1 Background

In this research, we propose a group management layer for the design and implementation of fault-Tolerant mobile agents. Mobile agents are autonomous software programs which are executed on a runtime environment called "agent system" and/or "agent execution environment". The mobile agent can carry its own internal state and migrate to other nodes connected to the communication network. Mobile agents can offer several advantages in a dynamic computing environment including flexibility and higher performance. Mobile agents can be easily implemented even by programmers with limited experience.

Recently, the ubiquitous use of the computer and the Internet as well as the increasing use of mobile devices such as PDAs resulted in an increasing interest in using the mobile agent paradigm for mobile and remote execution.

Many mobile agent applications has already been implemented such as a search engine for WWW. However, mobile agents do not provide any guarantees in case of failures such as network failures, network partitions and node failures. Such problems along with the security issues limits the acceptance and the usage of mobile agents in complex and critical application domains.

To apply mobile agents and agent systems to critical application or business applications, we must be able to build reliable and robust systems using the mobile agent paradigm.

2 Purpose

The purpose of this study is to provide system support to facilitate the design and implementation of fault-tolerant mobile agents.

We propose to introduce fault tolerance concepts and techniques in building fault-tolerant mobile agent systems. To build such fault-tolerant systems, we have to introduce some form of redundancy in the system such as using mobile agents as backups. The introduced redundancy, if it is not well designed, might decrease the overall reliability of the system. For example, if there is one original(primary) active agent and a number of copies of this agent (backup agents) which are passive. In some cases and after a failure we might have situations where two agents become active and might then introduce inconsistencies in the system.

We propose to apply the *Group Communication* approach to implement fault tolerant mobile agent systems and to ensure system consistency despite failures and recoveries. The group communication approach supports the group concept where processes working together to achieve a common goal such as fault tolerance are joined into a group and it also provides communication primitives to allow group members to exchange messages following a certain set of rules. The group communication approach proved to be very efficient in implementing traditional fault tolerant distributed systems. Group communication provides two services to groups members that is a membership service to monitor the membership changes within a group and a communication service that provide atomic multicast primitives to groups members. These services can be used to build fault tolerant and highly reliable systems.

In our approach we create groups of agents where the primary agent and all its back-ups are joined into the same group. Group communication services are used to maintain consistency among group members and to ensure that at any time there is only one primary agent that is active, other agents are back-ups.

3 Our Approach

The proposed approach is based on the group communication approach. However, the traditional group communication approach cannot be used for mobile agents in a straightforward way. This is because traditional group communication services are build on the assumption that processes that are member of a group are static entities that can not move. If we apply the traditional group communication approach to mobile agents in a straightforward way, when an agent moves from a node it will be considered as faulty and removed from the group membership, after migrating to a new host the agent should then rejoin the group as a new member. This result in a large overhead due to the number of group configuration changes and might produces inconsistencies in the systems, we might even not be able to achieve any progress.

To solve this problem, we introduce new concepts and techniques to introduce and support agent migration in the traditional group communication approach.

We propose a system architecture for building fault tolerant mobile agent systems

based on group communication. The proposed architecture can be used to build mobile agent application using different types of agents that is mobile agents created using different agent systems such as Aglets and AgentSpace. The proposed architecture is described in (figure1.1).

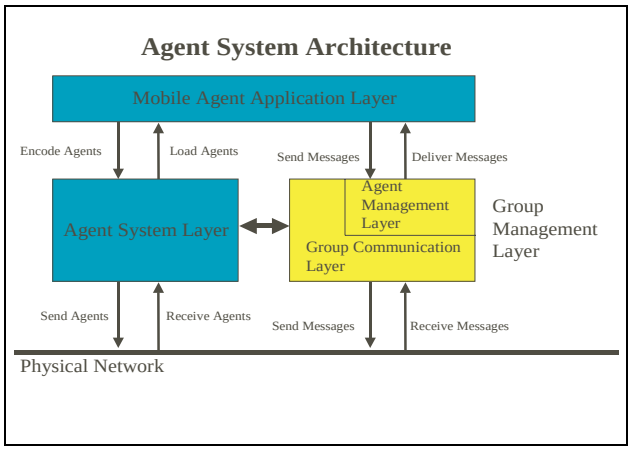

Figure 1: System Architecture

Typically, an agent system is constructed from *Agent Application Layer** and *Agent System Layer*. We propose to include a *Group Management Layer*.

The Group management layer is constructed from a *Group Communication Layer* and an *Agent Management Layer*.

Group communication layer maintains agents as groups, provides multicast primitives for group members and detects faulty agents.

The Agent management layer implements several fault tolerance protocols such as the majority voting protocol in order to ensure the at most once property for primary agent. In case of network partition where a group A can be divided into two groups A1 and A2, to ensure the at most once property we allow progress at only one group which is the majority group either group A1 or A2 . The majority group is determined by the number of agents joined to the group say group A1, if this number is equal or exceed the majoirty of the number of members joined to the previous group that is group A, group A1 is then called a majority group. In such a case, all members of group A2 are discarded. Therefore our system can ensure the at most once property for primary agent.

The Agent management layer provides also a two-phase commit protocol service to commit or abort transactions. The two-phase commit protocol is used to commit the execution of a transaction by the primary agent with all other group members that is with back-up agents. In doing so we can ensure the exactly once property required for electronic commerce applications.

*The space for running agents

4 Prototype Implementation

We implement the prototype of our system on *AgentSpace*^{TM}[†]. The purpose of this implementation is to evaluate that the proposed approach can be easily applied for agent systems and to test its effectiveness and flexibility.

Our system is constructed from the server process *Group Manager(GM)* and the package *fta* which can only be used on AgentSpace. Each component is implemented by java.

As a result of this implementation, we could confirm that we could easily apply and adapt the group communication approach to agent systems. For our system to support several agent system, we will only have to add a new package fta for this agent system.

5 Concluding Remarks

In this research, we proposed a group management layer to facilitate the design and implementation of fault tolerant mobile agents. We also introduced some fault tolerant techniques and protocols to ensure the at most once property of primary agents and the exactly once property of electronic commerce applications. We implementation a prototype of our system and we succeded in using the system with the AgentSpace agent system. We also proved that our system is modular and flexible in that it can support different types of agent systems.

Our system can provide fault tolerance and allow us to implement robust and highly reliable mobile agent applications. In this research we do not address security issues, we are planning to address this issue in our future work and to build further packages to support other agent systems such as Aglets, Bee-gent, and Concordia.

As future work, we are also planning to explore the following topics:

- support cooperative works
  We will support cooperative works for agents which belong to different groups.
- describe the behavior about our system formally
  We think to describe the behaveior in our system. It is very important to analysis our system rigorously.
- apply wide area networks
  We think to apply and implement our system on a wide area network.

[†]AgentSpace is a agent system based on Java. It was made by Dr. Ichiro Sato