Interoperability for Mobile Agents by Incarnation Agents

Tetsuo Hasegawa  
Kenta Cho  
Fumihiro Kumenon  
Shin Nakajima  
Akihiko Ohsuga  
Shin'ichi Honiden

Toshiba Corporation  
Toshiba Corporation  
Mitsubishi Research Institute Inc.  
House University  
Toshiba Corporation  
National Institute of Informatics / Tokyo University

cho.kenta@toshiba.co.jp  
kumenon@mri.co.jp  
nkm@i.hosei.ac.jp  
akihiko.ohsuga@toshiba.co.jp  
honiden@ni.ac.jp

ABSTRACT

Many different kinds of mobile agent platforms have been developed. However, migration to a different type of mobile agent platform is actually impossible. As a solution to this problem, we propose an interoperability concept using incarnation agents. This concept realizes logical mobility between different kinds of agent platforms. The incarnation agent extracts an agent’s procedures and status, compiles them into an agent platform-independent format, and then re-compiles them to the target agent platform format, thus enabling the process to continue. The incarnation agent also has autonomy for interoperability. It manages complex migration for interoperability so that the agent’s procedures can be described in a simple mobile model. Moreover, the incarnation agent modifies the agent’s procedures so that they are applicable to the facilities of the target agent platform. Feasibility experiments using interoperability middleware have been conducted as a basis for realizing incarnation agents.

Categories and Subject Descriptors

D.2.12 [Software Engineering]: Interoperability

General Terms

Design, Experimentation

Keywords

Interoperability, Mobile agent

1. INTRODUCTION

A mobile agent is a process that may migrate to/from a mobile agent platform in another computer through a computer network. It would be worthwhile if it could migrate anywhere in the network. However, there are many types of mobile agent platforms, and migration to a different type of mobile agent platform is actually impossible.

FIPA [1] and OMG/Masif [2] are standardization activities related to mobile agents. However, neither refers to migration between different kinds of mobile agent platforms. Migration methods between different Java-based mobile agent platforms have been proposed as described in references [3] and [4]. However, there are also non-Java-based mobile agent platforms which have useful features that are not available in Java-based platforms. For example, script-type mobile agents have the feature that end-users can describe the agent’s procedures and execute them without compiling. Rule-based-type mobile agents have the feature that processing unsuitable for control structure description can be treated easily. Moreover, a major objective of interoperability is the integration of existing application systems, and its targets may be added or changed later. It is therefore desirable to permit migration to any kind of mobile agent platform.

We propose an interoperability method and interoperability middleware which can serve as a basic framework for realizing the method that has been developed in order to evaluate feasibility.

2. REQUIREMENTS

The requirements for interoperability and our approaches to solve them are described below.

Mobile agent model programming:

In a mobile agent model, a procedure continues to be processed at the remote host rather than invoking a remote procedure arranged previous to the execution. This model has several advantages. For example, the execution results can be observed and reflected in the consequent process. In addition, programs can be made more simple and straightforward by eliminating the communication protocols for invoking and replying to remote procedures. In order to preserve these advantages, when a mobile agent migrates to another type of mobile agent platform, its procedures and execution status are translated and incorporated into a substitute mobile agent body for the destination mobile agent platform. The mobile agent immediately before migration is maintained and its execution status is updated automatically according to how the execution status of the substitution agent is modified.

Transparency of mobile agent platform types:

In order to make the type of destination agent platform transparent, the procedures and execution status are not translated directly into the form of the destination mobile agent platform, but first into a common representation and then into the target form. As a result, when a new type of mobile agent platform is introduced, it is only necessary to formulate the translation rules between it and the common representation.

Adaptation to the destination environment:

There is the possibility that the destination mobile agent platform may have a more desirable facility to achieve the same result, compared with the facility used in the original procedure. In other cases, the original procedure may not
be suitable for the destination host environment, for example, in terms of memory occupancy size. In these cases, it is better to modify the procedure. In order to generate a procedure that has been adapted to the destination environment, the mediation procedure can be a highly abstract expression for translation into a suitable procedure.

3. INCARNATION AGENT

We propose a mobile agent interoperability method using a virtual mobile agent known as an incarnation agent that migrates between different types of mobile agent platforms on behalf of a mobile agent. A mobile agent migrates between mobile agent platforms of the same type in the usual way. However, when it migrates to a platform of a different type, it is first transformed into an incarnation agent and then migrates to the destination. After that, it is transformed into a mobile agent of the agent platform and the procedure continues. The advantage of interoperability using an incarnation agent is that autonomy can be introduced in the behavior of the interoperable migration. For example, when an incarnation agent can, it generates parallel execution procedures to improve efficiency by utilizing a facility of the target platform that is not available in the original platform.

The following processes describe the functions of an incarnation agent:

[Information extraction] When a mobile agent invokes the migration function to a different type of mobile agent platform, an incarnation agent is created. It extracts the mobile agent’s procedures and execution status from the agent platform. Additional information concerning how to update the execution status of the original agent, named as the life cycle information as described below, is also extracted as described below.

[Translation] The extracted procedure and current execution status are translated into a common representation, named as the intention script as described below, by referring to a translation rule.

[Migration] An incarnation agent migrates to the destination agent platform by sending this information to it.

[Creation of a substitute agent] A substitute agent of the destination platform is generated by translation from the intention script referring to the translation rules of the destination platform.

[Updating the execution status] When the substitute agent migrates to a different type of mobile agent platform, the execution status of the original mobile agent is updated and the subsequent procedures are continued.

The intention script represents the procedures and execution status independent from any mobile agent platform in the form of a script. It is a sequence of capabilities those are executable operations on a mobile agent platform. With regard to the execution status, an incarnation agent treats the restarting points and the values of the explicit variables in the mobile agent’s program. A restarting point is defined as the first capability invocation in the intention script, and the values of variables are represented as fixed values in arguments of capability.

The classification structure of capabilities consists of mobile agent facilities such as migration, general application facilities such as input/output, and facilities peculiar to an application domain. This structure of capabilities ranges from primitive functions to integrated functions. In particular, application domain capabilities are composed of integrated functions as typical facilities in the application domain. Each capability in an intention script is translated into procedures suitable for the destination mobile agent platform and the characteristics of the destination host. Autonomy of an incarnation agent is achieved by constructing an intention script with integrated capabilities as far as possible.

The life cycle information is used to update the execution status of a mobile agent immediately before interoperable migration after the substitute agent works on other types of mobile agent platforms. It consists of mappings between variables and migration capabilities in the intention script and the original procedure, as well as methods for updating their values.

4. EXPERIMENTS

Interoperability middleware, which is the basic framework for the proposed interoperability method, has been developed, and experiments involving migration between Plangent [5] and Flage [6] were conducted using an information-gathering application. The response time of the application and network traffic were measured in the following cases: normal migration, migration between mobile agent platforms of the same type but through the interoperability middleware, and migration between a Plangent platform and a Flage platform. The number of computer hosts was varied from 2 to 5 for each case.

The results showed that the increase in response time was less than 1.9 times that in the case of normal migration and the increase in the number of packets was less than 17%. For the Plangent platform, the response time and network traffic were decreased by migration through the interoperability middleware compared to normal migration. This is due to the fact that the Plangent platform sends all of the Java objects of a mobile agent, whereas the interoperability middleware sends only the procedures in script format, thus reducing the amount of communication. For interoperability migration from the Plangent platform to the Flage platform, the response times in the case that the original sequential procedure was translated into parallel ones by utilizing the agent replication facility of Flage were faster than in the case of translation into the same sequential procedure when the number of hosts was greater than four. This is an adaptation to the destination environment in the incarnation agent’s concept.

5. REFERENCES