Engineering Letters, 13:3, EL_13_3_21 (Advance online publication: 4 November 2006)

Bridging Aglets and Web Services to Create a Common Platform for Mobile Agents

Jon Tong-Seng Quah

Abstract— **This project proposes and implements a system for seamless integration of HP Web Services E-services and IBM Aglet mobile agent applications onto a common platform. The developed software enables all applications written in Aglets and HP Web Services to create joint services across the two platforms. It establishes an environment for collaboration, intercommunication, translation, and bridging. The whole design is thereby highly reusable and portable. This paper also analyses the requirements of E-Commerce and elaborates how effectively a combination of Aglets and HP Web Services can provide the necessary resources. We conclude by suggesting a common model for Aglets, HP Web Services, and traditional Internet components that enhances performance through synergies. The integration combines the advantages of each technology in its respective domains.**

Index Terms— **HP Web Services, Aglets, E-Commerce, Mobile Agent.**

I. Introduction

Agent-based systems have gained prominence over the last few years. One of the most interesting categories of agents is mobile agents [11]. Unlike static agents, which are restricted to operate within a single machine or address space, mobile agents have the ability to migrate over the network, execute tasks at each location and potentially interact with other agents that cross their paths. Such a feature is potentially very useful in e-commerce applications.

Hewlett Packard HP Web Services is an open-source software platform for creating, composing, mediating, managing, and accessing Internet-based E-services [5]. These E-services can interact with each other to advertise capabilities, discover other E-services, and ally with each other to offer new functionality. E-services can even negotiate to broker, bill, manage, and monitor each other. This is all accomplished in a dynamic, platform-independent, ad hoc, yet secure manner. HP Web Services' goals for the services it provides to users are very congruent with those of mobile agent systems.

As new Internet technologies continue to emerge, the process of defining how to implement multi-enterprise E-business models becomes increasingly complex and difficult to manage. Many activities to solve these problems have been undertaken with the help of mobile software agents. Until now, however, most of these projects have been stand-alone implementations, unable to interact with each other. Consequently, guidelines on how to build future projects, and integrate existing applications into these projects are in high demand. This prompted the need of a bridging mechanism for linking mobile agent platforms so that agents may be able to roam freely between hosts, thus enabling cross-platforms agent-based e-commerce applications.

Jon Tong-Seng Quah is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Republic of Singapore. (Phone: 65-6790-5871; fax: 65-6270-1556; e-mail: itsquah@ ntu.edu.sg).

II. Overview of Aglets and HP Web Services

A. Aglets Features and Architecture

An Aglet is a Java-based autonomous mobile software agent. As used here, a software agent is a program that can halt itself, ship itself to another computer on the network, and continue execution at the new computer. An agent does not restart execution from the beginning at the new computer; it continues where it halted on the last host. The key feature of this kind of software agent is that both its code and state are mobile.

Aglets are autonomous because once they are started they independently decide where they will go and what they will do. They control their own lifecycle. They can receive requests from external sources, but each individual Aglet decides whether or not to comply with external requests. Also, Aglets can decide to perform actions, such as travel across a network to a new computer, independent of any external request. [10].

Mobile agents are defined as objects that have **behavior**, **state**, and **location**. A subset of behaviors of every agent is inherited from the model, notably those behaviors that define the means by which agents move from place to place. Mobile agent models almost always define a method of inter-agent messaging as well. Finally, a mobile agent model is not complete without defining a set of events that are of interest to the agent during its lifecycle. For example, the event of arriving at a new location is a momentous one in the life of an agent and generally leads to the invocation of one or more of its behaviors [11].

B. HP Web Services Features and Architecture

HP Web Services is an open software platform designed specifically for the development, deployment, interaction and management of globally distributed E-services. In HP Web Services, services seek out needed resources and communicate with all kinds of machines, from large data center servers to portable handheld devices. HP Web Services-enabled services can communicate through public networks, past network firewalls, and to non-networked devices through low-level serial protocols [3].

HP Web Services aims at enabling services over the network – making existing resources (e.g., files, printers, Java objects and legacy applications) available as services, as well as lowering the barriers to providers of new services. The infrastructure's goal is to provide the basic building blocks for service creation, including [5]:

- Secure access to resources and services across local networks and the Internet
- Usage monitoring, billing, and access control
- Advertising and discovery of new services
- Mechanisms for negotiation to find the "best matched" service
- Introspection of the interface for interacting with services
- Independence of operating system, language, and device
- Ability to support large enterprise-wide, intra-enterprise, and global deployments

E-services that are capable of interaction should be able to dynamically discover and negotiate with each other, mediate on behalf of their users, and compose themselves into more complex services [5].

III. Architectural overview for an integration of Aglets and HP Web Services

The system consists of three sub-systems: HP Web Services client, Aglet client and Bridge Manager. The HP Web Services client operates in a pure HP Web Services environment without any additional configurations and it provides access for external HP Web Services legacy applications to the collaboration functionalities. The Aglet client is an Aglet agent hosted in pure Aglet environment (i.e. on a Tahiti server) and serves as a gateway for other Aglets to invoke the Bridge Manager's collaboration features. The Bridge Manager is the heart of the system. As combined unit of the HP Web Services's and Aglets' environments, it communicates and interacts with both technologies and constantly translates and bridges between the two systems. This entity allows collaboration between every HP Web Services and Aglet application.

Bridge Manager Software:

The Bridge Manager consists of both, a Bridge HP Web Services Core and a Bridge Tahiti Server. This combined unit together allows collaboration between any HP Web Services and any Aglet environment. The operating system hosting this entity needs to have both, the configuration required for the HP Web Services environment and the configuration required for the Tahiti Server and Aglets environment. The HP Web Services side of the Bridge Manager then mainly handles intercommunication and collaboration with HP Web Services legacy applications, whereas the Tahiti Server side attends to Aglets legacy applications. All use cases supported by the Bridge Manager are shown in Figure 1.

IV. Design principles for an integration of Aglets and HP Web Services

The following paragraphs elaborate on the methodologies and general design principles, which are deployed throughout the course of the design. These mainly describe fundamental design techniques necessary for developing HP Web Services software as well as Aglet software.

All the supporting use cases will be invoked at least twice from at least two of the three parts of the system: HP Web Services Client, Aglet Client and Bridge Manager. Their description can therefore sometimes be somehow generic and can contain conditions.

Figures 2 through 5 show the design class diagrams for the implementation of our Bridge System. The Bridge Manager class diagram (Figure 2) is one of the two components of the Bridging System. The other, the Event Distributor class diagram, is shown in Figure 3. This implements the HP Web Services Event Distribution use case and is running in an independent operating system shell. The HP Web Services Client class diagram is shown in Figure 4 and Figure 5 shows the Aglet Client class diagram.

A. 'Forward Object' Mechanism

The Bridge HP Web Services Core or the Bridge Tahiti Server invoke the use case after they have received either an HP Web Services Event Notification or an Aglet Event Notification, and provided the payload object to be forwarded. The notification type will be examined and the payload is extracted from the message and classified according to its object-type. For each single Aglet Client User the Bridge Manager will invoke the Transport Object use case. Each time, that use case will create a new Messenger Aglet, append the message with its payload and dispatch it to the respective remote Tahiti server.

B. 'Transport Object' Mechanism

The Bridge Manager software or the Aglet Client software invokes this use case when they want to transport an object from one Tahiti Server to another one. A new Messenger Aglet and a new proxy stub are created and the Messenger Aglet is assigned to that proxy stub. A new message is created and the given object is assigned as its payload. Then, the Messenger Aglet is dispatched through its proxy stub towards the destination Tahiti server. After having reached the destination Tahiti server, an Aglet event notification for the given event type is sent to the Tahiti server and thereby invokes its Notify Aglet Event use case. Together with the event notification, the message and its payload are delivered to the destination. The Messenger Aglet has thus physically transported the respective object to the destination Tahiti server and has sent an Aglet event notification to the receiver Aglet.

C. *'Deploy Connection' Mechanism*

This use case will be invoked by the HP Web Services Client software or the Bridge Manager software from the Send Object use case whenever an HP Web Services User Client or an Aglet User Client wants to send an object to all other HP Web Services Client Users and/or Aglet Client Users. A new HP Web Services event is created and the given object is appended as payload. In its function as an HP Web Services event publisher, the respective HP Web Services Core will send an event notification to the Bridge HP Web Services Core. The Distribute Event use case will take care of forwarding the HP Web Services event to all HP Web Services Client Users. Additionally, the Notify HP Web Services Event use case at the Bridge Manager software will trigger the forwarding of the event and its payload to all Aglet Client Users. The HP Web Services core connection is therefore being used to send the object to the Bridge HP Web Services core, where it will raise an HP Web Services event notification.

D. *'Receive HP Web Services Object' Mechanism*

The HP Web Services Event Notification use case in the HP Web Services Client software invokes this use case after it has received an HP Web Services event notification from the Bridge HP Web Services Core and has identified the payload. If the object is of type String, the User Interface use case will be invoked to display the respective message content. If the object is not of type String, the User Interface use case will be invoked to display the type of object that was received. Finally, the User Interface Use Case will be invoked to display the latest update on the system status.

E. *'User Interface' Mechanism*

The HP Web Services Client, the Aglet Client, and the Bridge Manager software invoke this use case automatically after their start-up. The user interface serves the purpose of visualizing the operation of the system and providing means of direct human interaction for the purpose of demonstration. If the human user presses the "Forward" button, a String will be captured from the respective text field and the Send Object use case of the corresponding system will be invoked. If the calling software requests an update on the status information, the provided textual information will be displayed in each respective text field. This includes the system's status information, the type of object sent and its textual content for the case it was a String message. If the human user presses the "Quit" button, the respective HP Web Services Client, Aglet Client, or Bridge Manager system will orderly release its resources and shuts down.

F. *'Error Message' Mechanism*

This use case can be invoked by every other use case when an unexpected condition has occurred and an operation could not be performed. It will handle the exception as intended for the specific cause. Additionally it will raise an error message with information on the system status, the type of exception, potential causes of the exception and the position of operation where the exception occurred.

G. *'Receive Aglets Object' Mechanism*

The Aglet Event Notification use case invokes this use case after it has received an Aglet event notification from the Client Tahiti Server and has identified the payload. If the object is of type String, the User Interface use case will be invoked to display the respective message content. If the object is not of type String, the User Interface use case will be invoked to display the type of object that was received. Finally, the User Interface Use Case will be invoked to display the latest update on the system status.

V. Benefits derived from aglets and HP web services

A. *Service Architecture*

The currently dominant E-Commerce architecture is the one based on existing Web technologies. These generally follow the client-server paradigm, with Web browsers as clients accessing the E-Commerce servers at the merchants end. The servers allow requests for goods and services and manage their billing securely. The primary technologies involved are server-side technologies, such as scripts, Java Servlets, Active Server Pages, and technologies to manage issues such as security (Secure Socket Layer (SSL) among others).

Our approach inherits the strengths of HP Web Services implementation of a service brokering architecture, which provides a dynamic ecosystem of E-Commerce entities offering Internet-based services. At the core of HP Web Services is a service broker, which mediates between service providers and the customers. A typical request for a particular service would be broken down into simpler requests, for each of which the set of service providers is dynamically discovered. Bids are requested from each of them and the one best matching the customer's preferences is selected. Eventually, a final service package is created, whose delivery and billing is negotiated with the customer and then is initiated. This E-broker discovers, composes, brokers, and completes a serious of transactions 'on-the-fly'.

One of these services could for example be an airline, e.g. providing a flight to New York. Another E-service could provide a hotel in Manhattan, while another one provides a limousine service in the area and yet another one delivers flowers. Then, a hard-working IT professional with limited time may want to spend a week-end with his wife, visiting Manhattan, having a limousine picking them up at the airport and driving to the hotel, where he wants a room higher than the eighths floor and view to Miss Liberty while flowers for his wife are delivered to the room right after they arrived. Now, this hard-working professional could simply enter this highly complex request to the travel agency E-service. This, in turn, could lead to a dynamic discovery of the airline, limousine, hotel, and flower E-service individually and 'on-the-fly' combine them into the complex service request. Then, the travel agent service would bill the businessman for the whole bundle and cover the costs for its other requested services.

One major advantage of HP Web Services as opposed to traditional E-Commerce is that all these E-services can be discovered and combined on the fly and then delivered as an entire composition. That is enabled through the brokering architecture.

B. Information Exchange

Many of today's E-Commerce applications include complex business processes with a large number of concurrent tasks. Business tasks are modeled as services and can be composed through other lower-level nested services. A typical complex request is broken down into simpler requests. The set of service providers for each of these simple requests is then dynamically discovered. Subsequently, the best match is invoked, and its execution mediated.

C. Bulk Data Transfer

The bridge architecture leverages on HP Web Services ability to provide asynchronous and synchronous communication in the same environment. Bulk data transfer is an easy task for HP Web Services just like for other distributed computing environments, like CORBA and RMI. It fits closely into distributed computing and is a direct extension from Networking Transport Protocols (like TCP/IP).

D. XML as Joint Communication Language

Aglets communication through messages is adapted to the XML format for its message content. The bridge developed in this project enables communication between Aglets and HP Web Services. The software can receive and send Aglet messages as well as deploy HP Web Services services. And it exports all these functionalities in the form of handy modules, to be configured together to fit individual needs. Furthermore, reuse was one of the major design considerations for this project. So, next the software could be easily extended with additional modules to implement a proxy between the Aglet world, HP Web Services and the Internet. A DTD-based interpreter should closely fulfill these requirements. This would enable document-driven Aglet cooperation. Moreover, it would allow Aglets to share ontology for multiple or even dynamic domains. In this way, the cooperation of dynamic Aglets would support *plug-and-play commerce*; mediating businesses that are built on one another's service. Aglets thus acquired some of the key functionalities of HP Web Services.

E. Firewalls

Internet-based E-Commerce involves multiple enterprises separated by firewalls. *Intra-enterprise* process management differs from *inter-enterprise* process management significantly. Different enterprises are not only separated by firewalls, but also have self-interests and individual data sharing scopes. When they are involved in a business process, they are unlikely to trust and rely on a centralized workflow server. Rather, they need support for peer-to-peer interactions. This has become the major impendence for using the conventional centralized workflow systems for inter-enterprise E-Commerce automation.

On the other hand, the bridge architecture utilizes HP Web Services Firewall Traversal Standard Services to overcome the difficulty. Since HP Web Services has its roots in distributed operating systems research, it also has an integrated support for fine-grained access control. The HP Web Services Engine can be inserted at multiple points in the chain between clients and remote services. These remote services will act and look just like a local service, since the HP Web Services Engine acts like a kernel. Thus, the administrator can see and control access to services inside his network and firewall traversal is supported.

F. Collaboration Services

An E-Commerce scenario typically involves the following activities: identifying requirements, brokering products, brokering vendors, negotiating deals, or making purchase and payment transactions. Today, these activities are initiated and executed by humans.

Using Aglets, or in general Mobile Agents technology to support E-Commerce automation is a promising direction. Aglets could be personalized, continuously running and semi-autonomous, driven by a set of beliefs, desires and intentions (BDI). They could be used to *mediate* users and servers to automate a number of the most time-consuming tasks in E-Commerce, with enhanced parallelism.

HP Web Services then provide the engine to extend the services into inter-enterprise business model, as it was primarily designed for enabling the creation of dynamic, Internet-based business relationships through the ad hoc discovery and interaction of E-services. E-services include applications, computing resources, business processes, and information, delivered securely over the Internet. The HP Web Services Service Framework Specification (SFS) defines standard business interactions and conventions as XML documents that allow E-services to dynamically discover and negotiate with each other and compose themselves into more complex services.

G. Dynamics in E-commerce, Aglets, and HP Web Services

E-Commerce applications operate in a distributed computing environment involving multiple parties with dynamic availability and a large number of heterogeneous information sources with evolving contents. Dynamic relationships among a large number of autonomous service requesters, brokers and providers is common. A business partnership (e.g. between suppliers, resellers, brokers, and customers) is often created dynamically and maintained only for the required duration such as a single transaction. E-Commerce activities typically rely on distributed and autonomous tasks for dealing with such operational dynamics. Thus, E-Commerce is a *plug and play* environment. Services need to be provided on demand. To support such dynamics, and E-Commerce infrastructure must support the cooperation of loosely coupled E-Business systems.

The bridge architecture started from the beginning with the vision of dynamic brokering. HP Web Services provide Services Framework Specification (SFS) that allows E-services to dynamically discover and negotiate with each other and compose themselves into more complex services. This creates an open service model, allowing all kinds of digital functionality to be delivered through a common set of APIs.

H. A Combined Model for Aglets and HP Web Services

Many E-Commerce applications include complex business processes, and involve multiple enterprises. These goals could be achieved through the use of HP Web Services and XML in combination with Aglets. The use of HP Web Services could allow Aglets to communicate across enterprise boundaries, with fine-grained access control, firewall traversal and other infrastructure services. HP Web Services could provide a messaging service to each Aglet domain coordinator within one enterprise. This service would then become the single gateway to the Aglet domain, and could be made standard for all Aglet domains. Within a domain, the domain coordinator could forward messages to other Aglets using intra-domain Aglet messaging.

The bridge architecture implemented, among other functionalities, the communication in the publish/subscribe mode between any number of HP Web Services cores and Aglets. By using this feature, when an Aglet intends to buy some electronic parts, for example instead of checking the vendor Aglet one by one, it can publish an availability-check message, and HP Web Services can broadcast this message to the entire mass of vendor Aglets who subscribe to this message. All these work easily across enterprise boundaries and across firewalls.

Finally, XML is used as *lingua franca* to enable compatibility between Aglets, HP Web Services and other Internet-based services. Aglets could deploy the XML format as a standardized way to represent their message payload, while HP Web Services provides build-in support in form of an XML API, implemented through HP Web Services's WebAccess. This might lead to compatibility even with many other technologies, as XML is expected to become a universal standard for exchanging data in E-commerce, other loosely coupled, extensible and dynamic applications, or generally for Web-based data exchange.

VI. Conclusion and future work

During the implementation phase of our project, a software collection of utility methods necessary to perform collaboration between Aglets and HP Web Services was successfully developed. The provided modules, methods and functionalities can be used for every existing application written in HP Web Services or Aglets to create common services together with any number of other Aglets and/or HP Web Services applications. The whole software is designed with a strong emphasis for reuse and portability and is, therefore, not only implemented platform-independently without any hardware-specific properties, but also designed in a highly modular way, so that individual functionalities can be invoked separately.

The software consists of three sub-systems: HP Web Services Client, Aglet Client and Bridge Manager. The HP Web Services Client operates in a pure HP Web Services environment without any additional configurations and it provides access for external HP Web Services applications to the collaboration functionalities. The Aglet Client is an Aglet agent hosted in a pure Aglet environment (i.e. on a Tahiti server) and serves as a gateway for other Aglets to invoke the Bridge Manager's collaboration features. The Bridge Manager is the heart of the system. As combined unit of HP Web Services's and Aglets' environments, it communicates and interacts with both technologies and constantly translates and bridges between the two systems. This entity allows collaboration between every HP Web Services and Aglet application.

The previous sections also highlighted the benefits of the bridge architecture. Various aspects for the specific nature and persistent characteristics of E-Commerce applications are analyzed and how they can be supported by the combined system discussed.

XML is deployed as the common language for information exchange in *'plug-and-play commerce'*.

In future projects, Aglets, using XML format as a standardized way to represent their message payload, would lead to XML being used to enable compatibility between Aglets, HP Web Services and other Internet-based services. A proxy server could transform these Aglet messages into the HTTP format and transmit them via TCP/IP. The software developed during our project already enables cooperation, communication, translation, and bridging between Aglets and HP Web Services. HP Web Services already provides build-in support for XML in the form of a special API, implemented through HP Web Services's WebAccess. All the collaboration features developed in our project are exported in the form of handy modules. Following this, the software system could be extended with these proxy features, eventually becoming an E-service and therefore available to all Aglets and HP Web Services clients.

Secondly, our project has already implemented the major entities for a combined model of HP Web Services and Aglets. Such a combination could result in the beneficial use of both and would provide significantly more value than simply the sum of both. Meaning to say, the combination itself could improve the performance. HP Web Services could serve as the communication backbone of an Aglet domain (e.g. an organization) and connect easily through firewalls to any number of other Aglet domains. HP Web Services would add fine-grained access control, advertising, discovery, composition, metering, billing, event aggregation, and firewall traversal to the combined model. The provided software from our project could become the gateway between Aglet domains, HP Web Services domains and ordinary Web servers. Furthermore, HP Web Services could take care of all the bulk data transfer to ensure the required throughput and privacy.

Aglets could enfold their functionalities within their domain (organization) as autonomous distributed components, offering greater flexibility and acting on behalf of their clients to fulfill many types of tasks.

Eventually, the combination and interaction of HP Web Services and Aglets could enhance the probability of both systems being propelled into widespread usage.

References

[1] A. Chavez and P. Maes, *Kasbah: An Agent Marketplace for Buying and Selling Goods*, Proc. First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, 1996.

[2] William R Cockayne and Michael Zyda, *Mobile Agents*, Manning Publications Co., 1998.

[3] Hewlett Packard, *Ten Ways to Think HP Web Services*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/ThinkEspeak.pdf.

[4] Hewlett Packard, *System Management*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/SysMgmt.pdf.

[5] Hewlett Packard, *Architecture Specification*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/HP Web ServicesArch.pdf.

[6] Hewlett Packard, *JESI Programmers Guide*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/Jesi-PgmGuide.pdf.

[7] Hewlett Packard, *Service Framework Guide*, HP Web Services Documentation, 2005.

[8] Hewlett Packard, *Contributed Services*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/ContributedServices.pdf.

[9] Hewlett Packard, *WebAccess Programmers Guide*, HP Web Services Documentation, 2005, Available: http://www.HP Web Services.net/library/pdfs/Webaccess-PgmGuide.pdf.

[10] IBM, *Aglets Software Development Kit*, Aglets Documentation, 2005, Available: http://www.trl.ibm.co.jp/aglets/.

[11] Danny B. Lange and Mitsuru Oshima, *Programming and Deploying Java Mobile Agents with Aglets*, Addison-Wesley, 1998.

[12] Alexandros Moukas, Robert Guttman, Giorgos Zacharia, and Pattie Maes, *Agent-Mediated Electronic Commerce*, MIT Media Laboratories, 2005, Available: http://ecommerce.media.mit.edu.

[13] Object Management Group, *Component Object Request Broker Architecture*, 2005, Available: http://www.omg.org

[14] M. Rusinikiewicz, W. Klas, T. Tesch, J. Wasch, and P. Muth, *Towards a Cooperative Transaction Model - The Cooperative Activity Model*, VLDB'95, 1995.

[15] Douglas C. Schmidt, *Distributed Object Computing with COBRA Middleware*, Washington University, 2005, Available: http://www.cs.wustl.edu/~schmidt/cobra.html.

[16] Sun Microsystems, *Java Remote Method Invocation (RMI)*, Java Development Kit, 2005, Available: http://java.sun.com/products/jdk/rmi/.

[17] Sun Microsystems, *Jini Connection Technology*, 2005, Available: http://www.sun.com/jini/

[18] World Wide Web Consortium (W3C), *Extensible Mark-up Language (XML)*, 2005, Available: http://www.w3.org/XML/

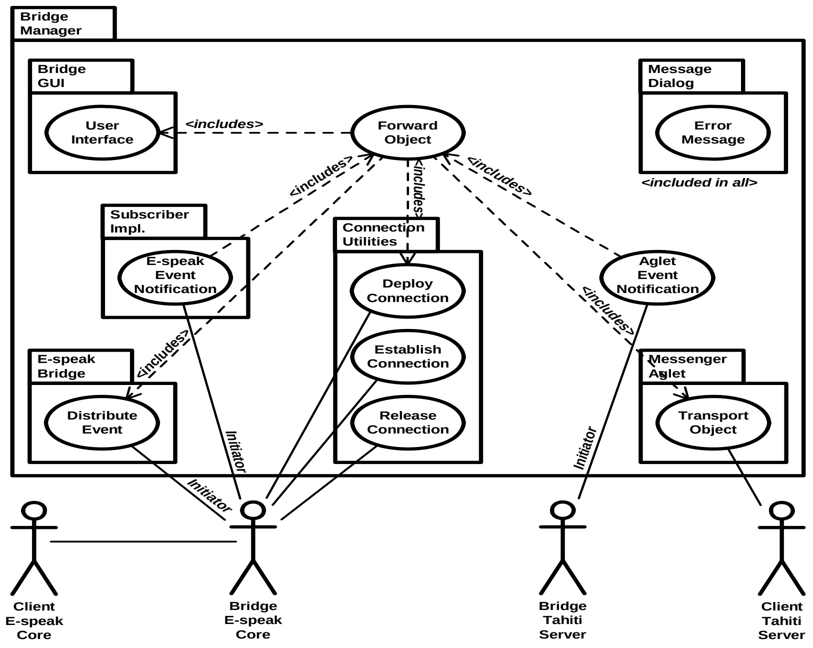

Figure 1: Bridge Manager Use Case Diagram

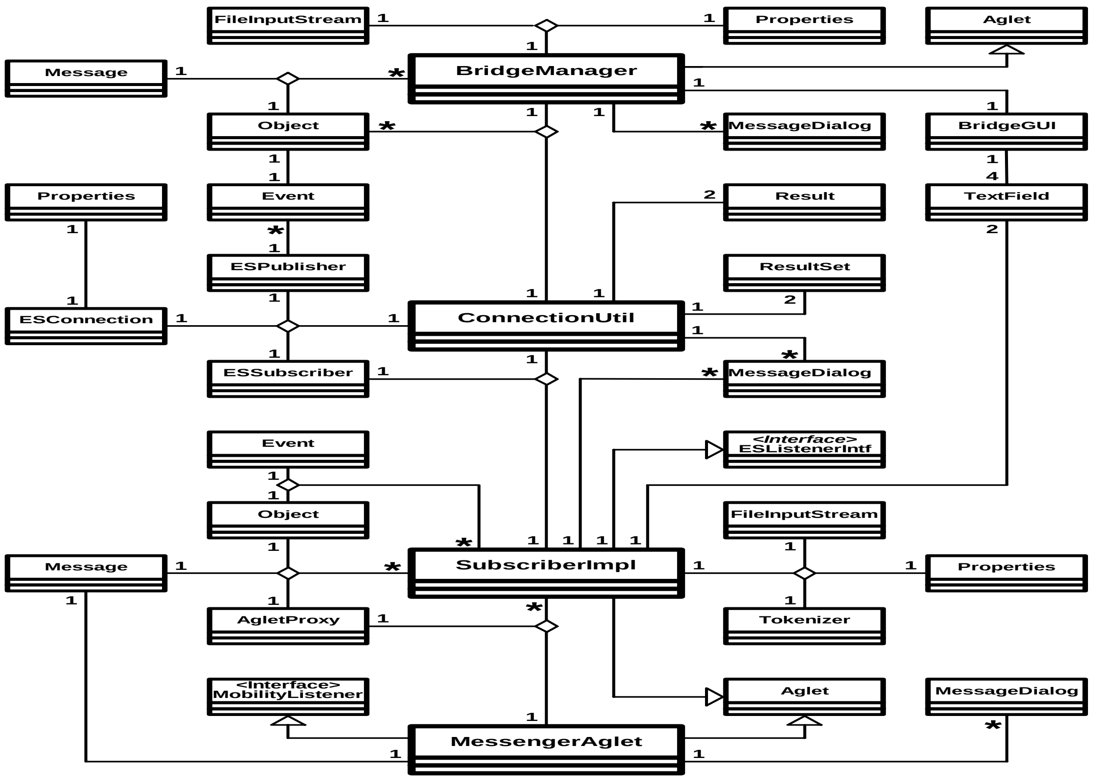

Figure 2: Bridge Manager Class Diagram

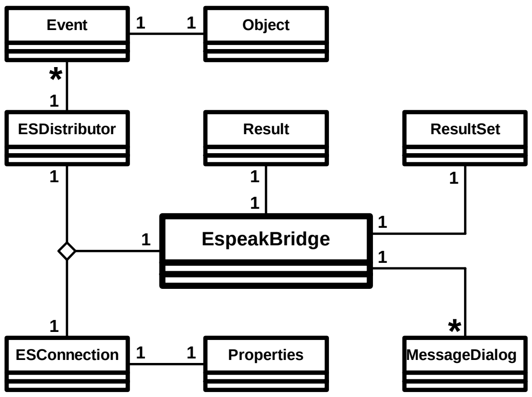

Figure 3: Event Distributor Class Diagram

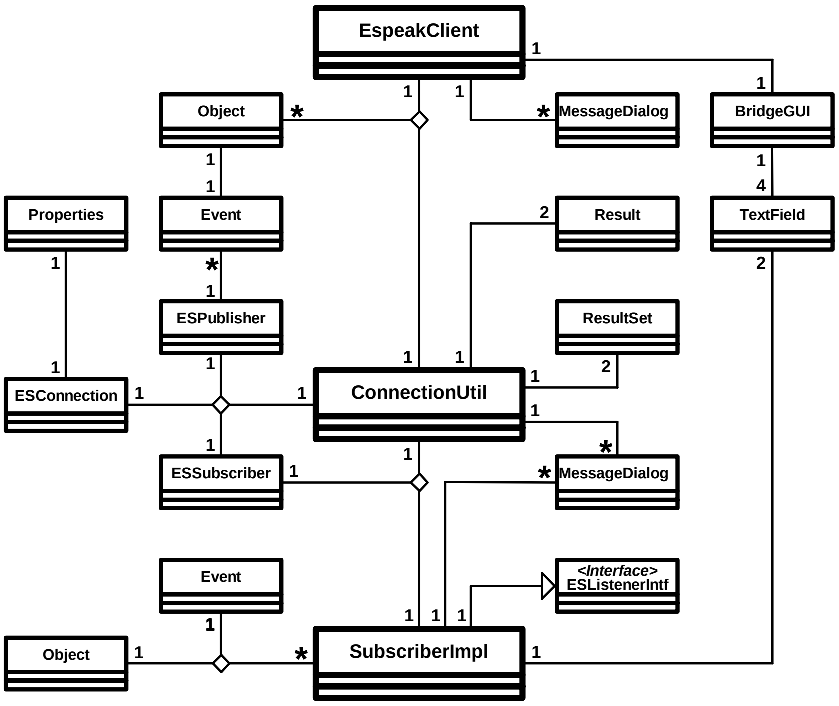

Figure 4: HP Web Services Client Class Diagram

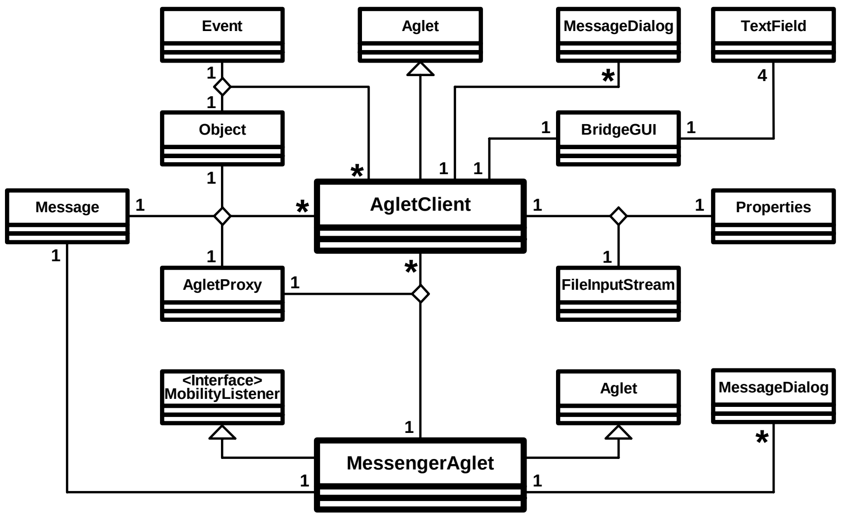

Figure 5: Aglet Client Class Diagram