An Architecture and Framework for Agent-Based Web Applications

Alexander Pokahr and Lars Braubach

Distributed Systems and Information Systems
Computer Science Department, University of Hamburg
{pokahr | braubach}@informatik.uni-hamburg.de

Abstract. The construction of web applications is a complex task as different kinds of technologies need to be integrated. To ease the task of developing web applications many different web frameworks have been conceived. These frameworks aim at providing support for recurring and tedious development tasks and address complexity by separating the basic concerns of applications. Most of the currently available web frameworks adhere to the widely accepted Model 2 design pattern that targets a clean separation of model, view and controller parts of an application in the sense of the model view controller (MVC) pattern. In this paper it is shown how the basic Model 2 archiecture can be adapted for enabling its usage in combination with business logic realized with agent technology. Besides the architecture itself additionally its realization within the Jadex Webbridge framework is sketched.

1 Introduction

The popularity of web applications is steadily increasing for which one important reason is that they can be accessed via browsers in a standardized way. In this regard they facilitate the execution of arbitrary applications without the need for installing or updating software components. These characteristics make web applications desirable even for more advanced and complex business tasks. Agent technology has already been used in many research and industrial projects for building enterprise scale applications [4,2], but only few works exist that aim at a systematic integration of agent and web technology allowing to easily build agent-based web applications. Hence, in the following an architecture and a corresponding framework for the efficient development of agent-based web applications are presented.

2 Architecture

The main aim of the approach consists in separating the agent-specific parts of an application from the web-specific parts allowing web and business logic developers to focus on their individual competency respectively. Foundation of the proposed architecture is the widely used and accepted Model 2 design pattern [3]. The pattern proposes a separation of concerns, whereby each of the three

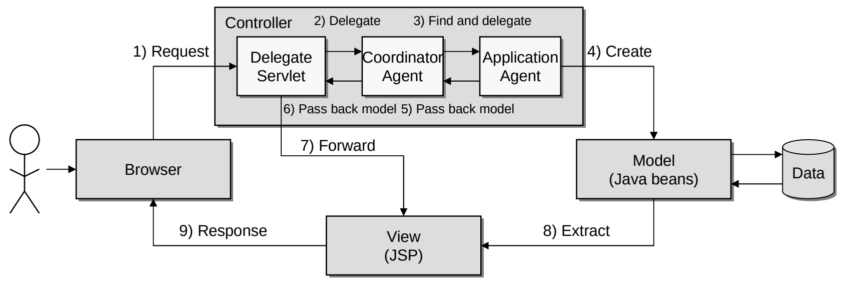

Fig. 1. Agent-based Model 2 architecture

proposed aspects plays a fundamentally different role. The *model* represents the domain-specific representation of the data on which the application operates. It is used by the *view* which has the purpose to render the data in a user-friendly manner. In between the *controller* serves as a connector that translates interactions with the view into actions to be performed on the model.

In an agent-based web application, the agents are responsible for the execution of the application logic. In the traditional Model 2 architecture, the application logic is executed by the controller, which is realized as a Java servlet. To achieve the seamless integration of agents with the web, the Model 2 architecture is extended as shown in Fig. 1 to allow for the execution of agent behavior inside the controller. For this purpose the controller is split into three different entities. The *delegate* servlet forwards the browser request to the agent layer and renders the result by redirecting to a chosen JSP. On the agent side a dedicated *coordinator* agent is responsible for finding or creating a suitable *application* agent that is capable of handling the request. The result is then passed back from the application agent to the coordinator which communicates it back to the delegate servlet.

3 Framework

The agentified Model 2 architecture presented above has been realized in a generic software framework based on the Jadex BDI (belief-desire-intention) agent system [1].[1] This new framework, called *Jadex Webbridge*, enables application developers focussing on the three core aspects of an application, i.e. the application logic using agents, visualization via JSP pages, and the domain data utilizing Java objects.

In the Webbridge framework the delegate servlet and the coordinator agent are responsible for mediating between these elements. They have been realized as reusable components within the framework and offer clear configuration and extension points. From the perspective of an application developer the main task consists in configuring the application settings via the normal `web.xml` file and additionally in developing the application agents for handling the web requests. The latter task is simplified by a generic agent module, called web interaction

[1] `http://jadex.sourceforge.net`

capability, which can be included by the developer into application agents and handles all communication aspects with the coordinator behind the scenes.

The generic coordinator automatically forwards Web requests as agent messages to an application agent. The *web interaction capability* is included into an application agent and automatically handles request messages sent by the coordinator. For each message a goal of type `web_request` is created, which has to be handled by custom application plans. The result of the goal processing is automatically communicated back to the coordinator. From the viewpoint of application developers, the web interaction capability converts web requests into goals that belong to the application agent. Therefore, the details of the web request handling are abstracted away from agent programmers allowing them to focus on the behavior of the application agent. This behavior can be built taking the intentional stance and applying the BDI-specific mentalistic notions such as goals and plans. Request processing can of course involve interactions with other agents if appropriate.

4 Conclusion

In this paper, an architecture and a corresponding framework have been presented that allow combining agent and web technology. The proposed architecture extends the well known Model 2 design pattern for web applications and introduces a decomposition of the controller part into a delegate servlet, a coordinator agent and application agents. Supporting the separation of concerns established by Model 2, the architecture further separates the web layer from the agent layer, thereby providing a solid foundation for combining agent-based application functionality and web-based user interaction.

To ease the development of applications following the proposed architecture, the Jadex Webbridge framework has been presented, which provides ready-to-use and extensible functionalities realizing the delegate servlet and the coordinator agent. Additionally, a web interaction module (capability) is provided that encapsulates the generic functionalities needed by application agents. It abstracts away details of message-based communication and reduces the task of the application agent developer to writing plans for the domain logic of pursuing goals, which correspond to web requests.

References

1. L. Braubach, A. Pokahr, and W. Lamersdorf. Jadex: A BDI Agent System Combining Middleware and Reasoning. In *Software Agent-Based Applications, Platforms and Development Kits*, pages 143–168. Birkhäuser, 2005.
2. J. Castro, M. Kolp, and J. Mylopoulos. Developing agent-oriented information systems for the enterprise. In *Proc. of the 2nd Int. Conf. on Enterprise Information Systems (ICEIS 2000)*, pages 9–24. ICEIS Secretariat, 2000.
3. N. Ford. *Art of Java Web development: Struts, Tapestry, Commons, Velocity, JUnit, Axis, Cocoon, InternetBeans, WebWorks.* Manning Publications, 2003.
4. N. R. Jennings and M. J. Wooldridge. *Agent Technology - Foundations, Applications and Markets.* Springer, 1998.