Chapter 2.15 Social Structure Based Design Patterns for Agent-Oriented Software Engineering

Manuel Kolp
Université catholique de Louvain, Belgium

Stéphane Faulkner
University of Namur, Belgium

Yves Wautelet
Université catholique de Louvain, Belgium

ABSTRACT

Multi-agent systems (MAS) architectures are gaining popularity over traditional ones for building open, distributed, and evolving software required by today's corporate IT applications such as e-business systems, Web services, or enterprise knowledge bases. Since the fundamental concepts of multi-agent systems are social and intentional rather than object, functional, or implementation-oriented, the design of MAS architectures can be eased by using social patterns. They are detailed agent-oriented design idioms to describe MAS architectures composed of autonomous agents that interact and coordinate to achieve their intentions, like actors in human organizations. This article presents social patterns and focuses on a framework aimed to gain insight into these patterns. The framework can be integrated into agent-oriented software engineering methodologies used to build MAS. We consider the Broker social pattern to illustrate the framework. An overview of the mapping from system architectural design (through organizational architectural styles), to system detailed design (through social patterns), is presented with a data integration case study. The automation of creating design patterns is also discussed.

INTRODUCTION

This section introduces and motivates the research. In Section 1.1, we describe the advantages of using multi-agent systems over traditional systems. Section 1.2 presents the importance of *patterns* for designing information systems. We formulate our research proposal in Section 1.3. The section 1.4 introduces elements for work validation. The context of the research and an overview of the state of the art are given in Section 1.5. Finally, Section 1.6 presents the organization of the article.

Advantages of Multi-Agent Systems

The meteoric rise of Internet and World Wide Web technologies has created new application areas for enterprise software, including e-business, Web services, ubiquitous computing, knowledge management and peer-to-peer networks. These areas demand software that is robust, can operate within a wide range of environments, and evolve over time to cope with changing requirements. Moreover, such software has to be highly customizable to meet the needs of a wide range of users, and sufficiently secure to protect personal data and other assets on behalf of its stakeholders.

Not surprisingly, researchers are looking for new software designs that can cope with such requirements. One promising source for designing such business software is the area of multi-agent systems. Multi-agent system architectures appear to be more flexible, modular, and robust than traditional architectures including object-oriented ones. They tend to be open and dynamic in the sense they exist in a changing organizational and operational environment where new components can be added, modified or removed at any time.

Multi-agent systems are based on the concept of agent which is defined as "a software component situated in some environment that is capable of flexible autonomous action in order to meet its design objective" (Aridor & Lange, 1998). An agent exhibits the following characteristics:

- **Autonomy:** an agent has its own internal thread of execution, typically oriented to the achievement of a specific task, and it decides for itself what actions it should perform at what time.
- **Situateness:** agents perform their actions in the context of being situated in a particular environment. This environment may be a computational one (e.g., a Web site) or a physical one (e.g., a manufacturing pipeline). The agent can sense and affect some portion of that environment.
- **Flexibility:** in order to accomplish its design objectives in a dynamic and unpredictable environment, the agent may need to act to ensure that its goals are achieved (by realizing alternative plan). This property is enabled by the fact that the agent is autonomous in its problem solving.

An agent can be useful as a stand-alone entity that delegates particular tasks on behalf of a user (e.g., a personal digital assistant and e-mail filter (Bauer, Muller & Odell, 2001), or a goal-driven office delivery mobile device (Castro, Kolp & Mylopoulos, 2002)). However, in the overwhelming majority of cases, agents exist in an environment that contains other agents. Such environment is a multi-agent system (MAS).

In MAS, the global behavior derives from the interaction among the constituent agents: they cooperate, coordinate or negotiate with one another. A multi-agent system is then conceived as a society of autonomous, collaborative, and goal-driven software components (agents), much like a social organization. Each role an agent can play has a well defined set of responsibilities (goals) achieved by means of an agent's own abilities, as well as its interaction capabilities.