Multiagent Systems (MAS) are applicable to engineering evolvable systems. It is essential to be able to capture and communicate the varied experiences of researchers in developing MAS. Software patterns are potentially useful for sharing experiences of what is and is not successful for MAS development. This paper presents our Two Way Classification Scheme and pattern template structure that is based on the classification scheme, for the effective classification and description of agent oriented software patterns.

Keywords: Adaptive systems, Agent Technology, Multiagent Systems, Software Patterns, Classification Scheme, Pattern Template.

1. Introduction

There is increasing demand for software systems that are capable of capturing the changes in their usage and environment and adjusting their behaviour according to these captured changes. The challenge is to build software systems that are able to adjust adequately to these changes after deployment (on the fly) without redesign or reprogramming. Software systems that exhibit this behaviour are said to be evolvable or adaptive.

The demand for the ability to evolve and other non traditional features of software systems such as intelligence, autonomy, rational planning, context level interaction, and reactivity has fueled the advancement of software agent technology. Software agent technology is focused on domains of application that require systems with these non traditional features such as online auction, intelligent manufacturing, logistics management, etc.

A software agent is a software entity that is situated in an environment and is capable of autonomous, proactive, reactive and social behaviours for achieving a set of goals. A system of interacting software agents is referred to as a Multiagent System (MAS).

Researchers have invested efforts in building MAS generating different results according to their experiences. It is essential to bring these varied experiences together in a way that they can be effectively shared, in order to enhance the comprehension of the concepts of and facilitate the successful application of agent technology to the construction of large scale complex software systems. Software patterns are potentially useful for sharing these experiences.

A software pattern can be defined as a general solution to a recurring problem in context. Software patterns provide a universal way to communicate important experience in software development and have been used with success in the object oriented programming community. Patterns have been suggested as potentially useful within the agent-oriented programming community for improving the communication and comprehension of concepts of agency [1].

Although a number of agent oriented patterns have been written, there is yet to be a comprehensive framework for describing and classifying these patterns. Hence the ability of most agent oriented patterns to communicate the concepts they describe are limited.

This paper presents our work in designing a comprehensive framework for describing and classifying agent oriented patterns. We present a two way classification scheme for classifying agent patterns with category attributes and a template structure for describing agent oriented patterns.

This paper is organized as follows. Section 2 is the motivation for this work. Section 3 presents the classification scheme and the description template design. Section 4 concludes the paper.

2. Motivation

Agent oriented pattern writers propose classification schemes for the patterns they write. [2] categorizes agent design patterns into traveling, task and interaction strictly in line with the patterns they have so far
identified. A view oriented approach for classifying agent design patterns is presented in [3]. It is however not clear where each of these views fall in the agent design levels and what aspects of the design they relate to. The issue with present agent oriented pattern classifications is that they either categorize at only one particular level of abstraction [2] or the classification schemes are not structured to correspond to do the different levels of abstraction in agent based development.

The pattern template (also known as pattern form or pattern description) is the vehicle for conveying the experiences that patterns are meant to share. Different pattern templates have been used in describing agent oriented patterns [4, 5]. These works use different pattern templates. However, they do not give a rationale for the chosen template or for adapting existing templates. The lack of rationale for defining these templates makes it difficult to compare the patterns; combine the patterns into pattern languages; or integrate the patterns into existing designs.

Agent oriented patterns templates require specific features for capturing the notions of agency effectively [6, 3] and one generic template is not adequate for describing all categories of agent patterns [3]. We need a comprehensive framework for classifying and describing agent oriented patterns. A framework that will allow patterns to be discovered, written and classified, searched for and found, adapted, integrated and combined. This comprehensive framework requires a classification scheme that will reflect the of levels abstraction, the peculiarities of individual levels of abstraction in terms of relevant concepts of agency, and the different aspects of development at each level. In addition, the classification scheme should define the basis for designing the description template for agent oriented patterns.

3. Agent Pattern Classification and Description
In this section, we describe the design of the Two Way classification scheme and category attributes and the template structure. We also describe the analysis and classification of the InteRRaP [7] pattern.

3.1 The Two Way Classification Scheme
In designing the Two Way classification scheme, we analyzed the levels of abstraction in agent oriented development as proposed by AOSE methodologies [8, 9]. Then we analyzed some MAS implementations to define prevalent and crucial issues and notions of agency in engineering MAS at the different levels of abstraction. The classification scheme is made up of the horizontal classification and the vertical classification. The horizontal classification is made up of four levels based on the definition of the levels of abstraction in agent oriented software development. These are Agent Analysis, MAS Architecture, Agent Architecture and MAS implementation levels. The vertical classification is designed on a level by level basis according to the analysis of the software development tasks and notions of agency that are relevant to each of the levels. The vertical classifications for each of the levels as shown in table 1.

<table>
<thead>
<tr>
<th>Agent Analysis (Level 1)</th>
<th>MAS Architecture (Level 2)</th>
<th>Agent Architecture (Level 3)</th>
<th>MAS Implementation (Level 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definitional</td>
<td>Structural</td>
<td>Definitional</td>
<td>Definitional</td>
</tr>
<tr>
<td>Organizational</td>
<td>Interaction</td>
<td>Structural</td>
<td>Technology</td>
</tr>
<tr>
<td>Interaction</td>
<td></td>
<td>Interaction, Strategic</td>
<td>...</td>
</tr>
</tbody>
</table>

Following is a brief description of the categories at the levels.

Agent Analysis-Definitional: The patterns in this category describe different ways of combining or breaking roles up to create agents.

Agent Analysis-Organizational: The patterns in this category describe ways of modeling virtual organizations with agents.
Agent Analysis-Interational: The patterns in this category define interaction issues such as type of interaction and structure of interaction.

MAS-Structural: The patterns in this category describe how to translate virtual organizations of agents into software system architectural structures featuring the components of the system and their associations.

MAS-Interactional: The patterns in this category describe the communication mechanism and agent interaction protocols amongst the c interaction amongst components and agents in the system.

Agent Architecture-Definitional: The patterns in this category define the type (e.g. information agent, mobile agent, assistant agent, etc) of agent to use for modeling an agent in the system.

Agent Architecture-Structural: The patterns in this category define the design of the internal structure of a single agent e.g. BDI.

Agent Architecture-Interational: The patterns in this category focus on the components of the architecture that describe how the agent handles interaction/collaboration/negotiation with other agents.

Agent Architecture-Strategic: The patterns in this category focus on how specific notions of agency (autonomy, reactivity, etc) are modeled in an agent.

MAS Implementation-Definitional: The patterns in this category define the programming paradigm for implementing a particular MAS based on the domain, the type of agents, MAS design and other factors.

3.2 Using category attributes to analyze and classify patterns

In order to design a process for the use of the classification scheme, it is necessary to define attributes for each of the levels and categories in the classification scheme. The patterns that fall into each level and category will be characterized by these attributes. Hence, there are attributes for the levels as well as each of the categories at levels. We used a combination of the characteristic notions of agency and development issues to define the attributes of the levels and the attributes of each of the categories at the levels of abstraction.

For instance, the attributes that should characterize patterns that belong to the Agent Architecture level (horizontal classification level 3) are Single Agent, Internal Architecture Components, Arrangement of Components and Constraints. The Structural (vertical classification) category at this level has the following attributes: Knowledge Component, Strategy Component, Interaction Component, Environmental Interface Component and Geometry. Analyzing the InteRRaP agent architecture [7], we note that it describes the internal components of a single agent. It features the knowledge, strategy and environmental interface components. Therefore, we classify InteRRaP as a Structural pattern at the Agent Architecture level of abstraction. See figure 1 for a diagrammatic representation of the InteRRaP agent architecture.

3.3 Agent Pattern Template Design

Here we present our design for agent oriented pattern templates and a template for agent architecture-structural patterns based on the design.

The agent pattern template design that we propose is based on three premises which are: template elements should be structured to clearly define the problem, context, forces, solution, and resulting context of the pattern; template elements should be designed to ensure that the agent concepts are appropriately described in the pattern; and there should be different templates for the different categories of agent patterns.
Given the premises above, we design an agent pattern template to consist of major template elements and sub-elements. The sub-elements are combined with the major elements to generate a template structure for a particular category of agent patterns. The major template elements are the common features of a good pattern i.e. problem, context, forces, etc. The sub-elements are introduced to capture the different concepts of agency necessary for describing agent patterns for the different categories of patterns according to our two way classification scheme. Table 2 presents our design of the template structure for *Agent Architecture-Structural* patterns.

<table>
<thead>
<tr>
<th>Major Template Elements</th>
<th>Sub-elements for Agent Architecture-Structural patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>Classification</td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td></td>
</tr>
<tr>
<td>Context</td>
<td></td>
</tr>
<tr>
<td>Forces</td>
<td>Goal, Autonomy, Social Ability, Environment, Adaptive Behaviour, Intelligence, Decision and action</td>
</tr>
<tr>
<td>Solution</td>
<td>Control, Knowledge Management, Strategy, Interaction Management, Environmental Interface, Structure, Dynamics</td>
</tr>
<tr>
<td>Known Uses</td>
<td></td>
</tr>
<tr>
<td>Resulting context</td>
<td>Adaptation/Integration</td>
</tr>
<tr>
<td>Related Patterns</td>
<td></td>
</tr>
</tbody>
</table>

4. Conclusion

There is a need to have a comprehensive framework for describing and classifying agent oriented patterns. In this paper, we address this need by presenting a two way classification scheme with category attributes and a description template design that is based on the classification scheme.

Our classification scheme and template structure serves as a basis for a systematic process for analyzing agent oriented patterns for purposes of placing the patterns in a catalogue, assessing their applicability to development problems, searching for patterns in the catalogue, and describing agent patterns.

5. References