7
Final Remarks

This work has analyzed an important issue that arises the software reuse in the agent-oriented software engineering. A study of the state of the art in the area has shown that only a few complete and well-grounded methodologies for the analysis and design of multi-agent systems have been proposed so far. We believe our approach will bring improvement to everyday software practice from the point of view of reusability. In this section we discuss the main contributions and limitations of this thesis, and pose the future work.

7.1. Contributions and Limitations

The goal of our work is to offer real promise for building repositories of reusable agent-oriented artifacts. The benefits of developing an effective agent-oriented artifact repository are allowing software engineers to build agent-based systems from pre-existing agent-oriented artifacts rather than laboriously develop each system from the ground up, and enormous time and energy can be saved in the development of new agent-based systems. The repository developed of agent components is based on semantics. It includes a search system to retrieve agents from the repository, with the capacity of detecting and discovering the agent components most relevant to concepts within the context of a domain. For that, we adapt the vector space model, based here on the semantics of the artifacts and a lexical database and using the weighting scheme *tf-idf*, to retrieve effectively the relevant agent components. By specifying characteristics of agent components by means of feature model and ontology, and classifying agents by means of taxonomies according their inherit characteristics as roles they perform within application domains, the users can reuse heterogeneous agents from previous systems or integrate existing agent components. In addition, a recommendation system and a subscription service were developed to support the user's search and user's knowledge of the existing agent-oriented artifacts.

The meta-model sets up all the characteristics about heterogeneous software agents, which includes their structure, functionalities and interfaces. The interfaces define the agent interdependencies; how a software agent can cooperate, coordinate and negotiate with others ones. The meta-model is implemented using an ontology, open for anyone to use and explore. The individuals of the ontology are the agent components stored in the repository.

Since agents are developed under dissimilar architectures, platforms and programming languages and agent components are different in their interfaces, internal architecture and functionalities, we created a feature model to describe and represent explicitly the agent variability. The feature model can be potentially adopted to represent agents attending specific characteristics, but search support is still not included. In addition, it can be used to derive and define axioms and rules in the ontology that describe agent components. For example in the feature model, agents necessarily have defined roles, and these roles may have explicit operations to carry out their objectives. These conditions can be represented in the ontology through axioms and rules.

As ontologies are generally considered complex and time consuming to be built, we structured the agent components by using taxonomies for classifying agents according their behaviors and application domains in a more realistic model for common users. The choice of using taxonomies to classify agents provides contextualized view of artifacts. We posit that such information may be helpful in aiding our recommendation system to automatically find knowledge on behalf of the system users. Nevertheless, it is not possible to represent the (semantic) intersection among application domains with the taxonomy.

We also created a recommendation system that allows software engineers to discover interrelated agents, and to learn new information as needed, improving user productivity, quality and promoting agent reuse.

The search system developed is based on the taxonomies and ontology. The semantic on the search methods fits applying stemming and/or lemmatization supported by a lexical database. This allows more relaxed matching retrieving interesting agent components since the variations of the query term. We implement various search methods based on the following types: keyword, facet, tag, platform, language and interface based searches. On the other hand, a limitation of our search system is the lack of a mechanism that allows software engineers to formulate high-level queries according to their needs. We did not include potentially advanced search methods in the repository, for example the composition of the search techniques. For instance, we could combine a keyword-based approach with the facet-based search or consider combinations of attributes as input data, e.g. fixed platform, programming language and an agent's behavior. The search parameters are reduced, improving the search performance.

Determining appropriate agent classification schemes makes the search more effective and efficient in the repository. The taxonomies and ontology are a promising solution for agent-based artifacts identification, classification and retrieval, taking into account the drawbacks associated with many of the existing encoding methods (e.g., text-based, lexical description-based) and the limitation of natural language's imprecision, and also provide a controlled vocabulary decreasing ambiguity at the linguistic and conceptual levels since they limit the number of concepts in a given model. In any case, it is indispensable to have an explicit description of the agent components, which depends on the users. It not only affects the description, but also affects the definition of the concepts in the taxonomies and the results of the search. This process of description and classification software artifacts is susceptible to the subjective rationale of the person who is responsible for these tasks. For example, in a facet-based scheme, two different people may designate various keywords to describe an artifact. Thus unfamiliar vocabulary to describe diverse information, frequently used in the software design process, prevents or makes it difficult for users to find proper artifacts.

The taxonomies are prepared to be extensible in whenever moment, for instance adding more facets in hierarchical form or adding relations between facets creating a graph like the ontology structure. In addition, we define a subscription service in the repository that allows users to be aware of the existing reusable agent-oriented artifacts in different application domains.

7.2. Future Research Directions

According to the limitations mentioned above, there are potential improvements and related projects that could be done. We could consider the use of the feature model to define the axioms and rules in the ontology that describes the agent components. It is possible as well to define an ontology to represent the intersection among application domains and implement some composition of the search techniques. We will analyze if both of these improve the relevance of the searches.

The literature has reported that the effort needed to manually classify artifacts of a software library is not manageable, since these tend to grow exponentially. Hence, it becomes necessary to have a tool that employs a machine learning approach that recommends or constrains classification of agents into several domain categories based on their characteristics. The user can perform the classification if the existing categories are in appropriate, but this has the drawback that it is based on users' conceptions of domain categories.

Users in the repository would have similar profiles, for example, they are developers of agent-oriented artifacts in the electronic e-commerce domain. This increases the consensus on user introduced domain categories. Among subscribers that belong to the same domain, future surveys can be used to increase consensus on domain categories introduced by users. Thus, users can be updated about the uploads of another user and they can increase the consensus on the domain categories. So, to carry out this function, a subscription service for users is needed.

To enrich the semantic knowledge base used by the recommendation and search systems, a relevance feedback tool could be included in the search system that prompts users to answer a displayed question and provide their answer according to the results of the search. The user can vote for search results on a ranked scale. The system also can infer the user vote for an artifact by monitoring the user's frequency of usage of that artifact. These are potential ways to enhance search results.

We must evaluate the system with other methods, taking into consideration that we do not know which and how many agent-oriented artifacts are relevant for a specific query. With the continual agent component registration, new concepts can be included dynamically in the taxonomy and the tag cloud. So, the search should be evaluated by watching how these changes may affect its performance. We posit that if these new concepts match with agent's characteristics, the result of the search system will be more refined and more productive.