IMPLEMENTING TRADING AGENTS FOR ADAPTABLE AND EVOLUTIVE UI-COTS COMPONENTS ARCHITECTURES

José Andrés Asensio, Luis Iribarne, Nicolás Padilla

Applied Computing Group, University of Almeria, Spain

Rosa Ayala

Computers and Environmental Group, University of Almeria, Spain

Keywords: Trading agents, COTS User Interfaces, Model Transformation, Model-Driven Engineering, Cooperative Systems, e-Business Systems Modelling.

Abstract: Most of the complex (e-Business) information systems need to accomplish with the use of open standards. *Environmental Management Systems* (EMS), for instance, state the international regulations of the ISO 14000 family, which establish the requirements to be fulfilled by an EMS in order to be accepted as such. One of these requirements concerns the *User Interfaces Development*. Because of the variety of final users that interact in this sort of complex information system (politicians, technicians, administrators, and so on) and due to a great deal of information (some critical and confidential), it is important to have real and practical scientific/technical proposals in order to build fast and efficient information exploitation systems. The human-computer interaction (HCI) of these systems need user interfaces that adapt to the users profiles' habits, and with intelligent software agents that mediate by the users in the search processes, exploitation and decision-making tasks. In this work we present a part of the SOLERES-HCI, a framework of the Soleres Project for developing COTS user interfaces by using trading agents. Our studies are being applied for developing advanced EMS and approaching *Model-driven engineering* techniques to the UI-COTS development.

1 INTRODUCTION

Organizations and administrations that work with environmental information for specific actions such as territory management or planning and organization of natural resources (among others), need to have innovative quality information systems in order to guarantee the success of their everyday activities. Therefore, for the experts who operate the system not only it is important to have reliable and updated information that helps to make the most appropriate decisions, as it happens in critical complex systems. Due to the variety of final users that cooperate with each other and interact with the system for decision making (for instance, politicians, technicians, administrators, etc.), it is also important to have: (a) exploitation information systems (environmental, in this case) to facilitate the human-human and human-computer interaction and coordination; (b) intelligent user interfaces that adapt to the users profiles' habits and; (c) intelligent software agents that intercede on be-

half of the users and facilitate the information interpretation tasks, the decision-making tasks and prediction/prevention tasks (which are the most important).

In this work, we present a part of the framework Soleres-HCI that supports all the *human-computer interaction* issues of a complex *environmental management system* (EMS). This portion of the framework concerns the developing of adaptable and evolutive user interfaces of the system by using: (a) trading agents that intercede between user agents and the information, (b) developing of traders that follow model-driven engineering perspectives, and (c) specialized Commercial Off-The Self (COTS) components for real-time user interface architectures.

The rest of the paper is structured as follows. Section 2 describes some issues for modelling user interfaces from UI-COTS. Section 3 continues with our SOLERES-HCI proposal, a human-computer interaction perspective based on trading (ISO/IEC, 1997) and software agents. Finally, Section 4 explains some of our ongoing research in the SOLERES project.

2 MODELLING UI-COTS

In recent years, different approaches for the design of user interfaces have been presented, most of them following model-driven UI designs, for instance:

(a) IDEAS (Interface Development Environment within OASIS) (Lozano et al., 2000): A methodology of UI development based on UML models.

(b) OVID (Object, View and Interaction Design) (Roberts et al., 1998): A methodology for UI design directed towards objects developed by IBM.

(c) TERESA (Paterno, 1999): A tool for the UI generation by using ConcurTaskTrees (CTT).

(d) WISDOM (Whitewater Interactive System Development with Object Models) (Nunes, 1998): A methodological proposal for UML-based UI.

(e) UMLi (Pinheiro, 2002): An extension of UML notation for the UI design.

In the SOLERES team there are also current works on UI modelling that use and extend the UML diagrams (Almendros and Iribarne, 2005) (Almendros and Iribarne, 2007). None of these works deal in depth with trading UI-COTS modelling.

On the other hand, our research aims to study UI following the approach of Component-based Software Development (CBSD) specialized for COTS components (*Commercial Off-the-Shelf*), (Meyers and Oberndorf, 2001). There are really few works showing realistic cases of IS development following the COTS paradigm or using multi-component UI-COTS. In (Iribarne et al., 2004) we developed an experiment COTS composition in Geographical Information Systems (GIS). However this approach doesn't solve a dynamic and evolutive UI-COTS perspective.

Furthermore, the tendency during the last few years in CBSD is to facilitate the automatic integration of commercial components by means of the composition (assembly) of their parts. The advances in trading services (or traders) have played an important role for it. Inspired on the trading model for COTS components in open-distributed systems (Iribarne et al., 2004), (Iribarne et al., 2005) we develop a trading agent service for UI-COTS in EMS systems.

3 TRADING IN SOLERES-HCI

Soleres HCI is the framework of the SOLERES environmental management system, specialized in the human-computer interaction. This level of information system follows the paradigm *of Computer Supported Cooperative Work* (CSCW) and uses technology of agents and multiagent architectures.
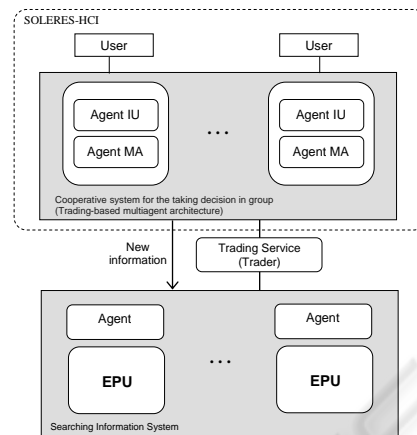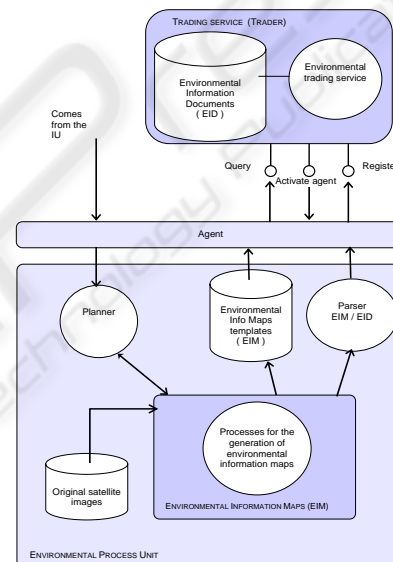

Figure 1: General architecture of SOLERES project.

Figure 2: Some details of an EPU object.

For the process of data exploitation we identify and structure the type of queries and the sort of information suitable to be consulted, by using techniques of hierarchical decomposition (i.e., trees, cut and pruning) and neural networks.

Figure 1 shows the (short) architecture of the SOLERES information system in our project framework. At the *user layer* side (top), the system is designed in order to be used for environmental decision-making tasks and in cooperation among different people (system's users) organized following different organized models (i.e. depending on their hierarchy).

This human-computer interaction and human-human interaction is guided by a cooperative system for the decision-taking tasks in group supported by a multi-agent architecture (next layer). Each user of the cooperative system has an UI agent whose function is
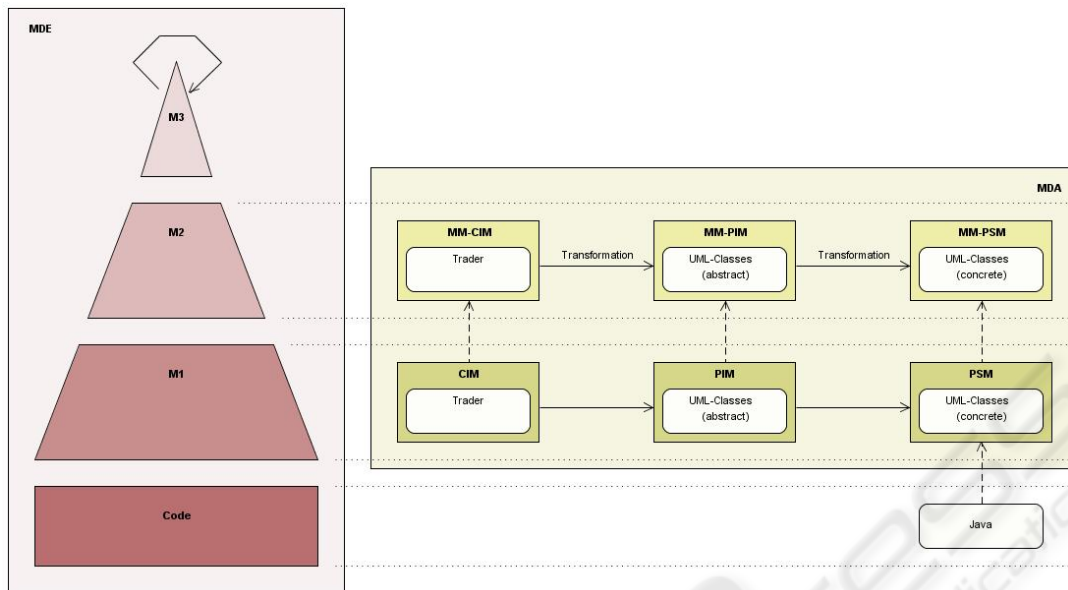
Figure 3: MDA steps of the trader views.

to adapt the UI to his/her needs by identifying his/her interaction habits with the UI. This UI agent mediates between the user and (a) the rest of the system's users (who have their own UI agent), (b) the search information system (next layer) directly, or (c) an environmental software agent (environmental agent - MA).

A virtual consultant cooperates with other agents within a pre-established multi-agent architecture and respects a model of organization and cooperation (that is or project's aim). The purpose of this cooperation (MA) is to facilitate the tasks of information exploitation: they interact with the search information system (we call an *Environmental Process Unit*- EPU) and filter the irrelevant information. The use of this sort of systems is quite useful because of its natural capacity to represent and implement organizational and social aspects that can help to identify and/or solve problems. Figure 2 shows internal details of an EPU.

For implementing the trading agent, we follow a *Model-Driven Engineering* (MDE) perspective based on the classical *Model-driven Architecture* (MDA) of the OMG. In this perspective, the trader model stays the three stages of the MDA: CIM/PIM/PSM (see Figure 3). A trader meta-model (MM) is defined for each stage, which describe the way to generate a diagrammatical model (UML) of the trader.

To translate a model into views (stages), we use model-transformation techniques. In our case, we use ATL for implementing the transformations of the trader. The language *ATL (ATLAS transformation language)* (Jouault and Kurtev, 2006) provides declarative and imperative constructs. The declarative part of *ATL* is based on rules. Such rules consist of a

source pattern matched over source models and of a target pattern that creates target models for each match. Figure 4 shows a piece of the ATL code for the transformation CIM/PIM of one of the five interfaces of a trader agent: the `Lookup` interface. In the code, we can observe how associations, properties or operations of the Lookup class are created.

```
rule Lookup {
  from f01:in_MM!Lookup
  using {
    O_Lookup: out_MM!Operation = 'null';
    O_get_lookup_if: out_MM!Operation = 'null';
    ...
  }
  to t01:out_MM!Class( name<-'Lookup',visibility<-#public )
  do { ...
    O_Lookup<-thisModule.create_Operation
      ('Lookup', #public, '', false, false, 1, 1);
    O_get_lookup_if<-thisModule.create_Operation
      ('get_lookup_if', #public, 'Lookup', false, false, 1, );
    O_get_register_if<-thisModule.create_Operation
      ('get_register_if', #public, 'Register', false, false, 1, 1);
    A_trader_C_Lookup_C_Trader<-thisModule.create_Association
      ('trader_C_Lookup_C_Trader', #public, false);
    Pr_trader_C_Trader<-thisModule.create_Property
      ('trader',#private,'Trader',false, false, 1, 1, false, '', false, false);
    ...
  }
}
```

Figure 4: A piece of the ATL transformation

Figure 5 shows the PIM trader model: a partial class diagram. In this framework we have also used the GMF/Eclipse to develop a tool for documenting CIM trader models. We use the OCL language of UML/OMG for describing the semantical restrictions
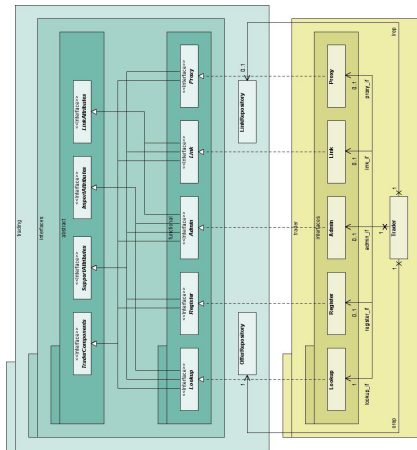
Figure 5: Informal class diagram model of the trader.

of a trader. At the end, **java** code generation from PSM trader models can be obtained by using a parser written in MOFScript.

For the implementation of the described framework we have used the Eclipse platform and JADE (http://jade.tilab.com/) for agents.

For space reasons, we have included here a part of the Soleres-HCI framework. A more complete version of the SOLERES project and details about the model transformations and implementations of the trading agent (p.e., Java code) are available at http://www.ual.es/acg/soleres.

4 ONGOING RESEARCH

The following are the specific objectives of our ongoing line of research.

Firstly, we plan to study the design of intelligent user interfaces. We will look into how to adapt the UI to different user's profiles.

Our scientific objective (complement to previous one) is to study algorithms of dynamic-services composition in UI-COTS components architectures. We try to analyze this kind of component and its marketplace (i.e., how many exist, which types, how to define them, what kind of existing repositories, etc.).

Furthermore, starting from previous works (Iribarne et al., 2004), we plan to implement trading services of UI-COTS components and their applications in *Cooperative Systems* (specifically in environmental management systems.

ACKNOWLEDGEMENTS

REFERENCES

Almendros, J. and Iribarne, L. (2005). Designing GUI components from UML use cases. In *IEEE Int. Conf. on the Engineering of Computer-Based Systems*, pages 210–217. IEEE Computer Society.

Almendros, J. and Iribarne, L. (2007). *Visual Languages for Interactive Computing: Definitions and Formalization*, chapter User Interaction and Interface Design with UML. Idea Group Inc. Hersey, U.S.A.

Iribarne, L., Troya, J., and Vallecillo, A. (2004). A trading service for cots components. *Computer Journal*, 4(3):342–357.

Iribarne, L., Troya, J., and Vallecillo, A. (2005). *The Development of Component-Based Information Systems*, chapter Trading for COTS Components to Fulfil Architectural Requirements. M.E. Sharpe, Inc.

ISO/IEC (1997). Information technology - open distributed processing - trading function: Specification. Technical report, ISO/IEC 13235-1, ITU-T X.950.

Jouault, F. and Kurtev, I. (2006). On the architectural alignment of ATL and QVT. In *SAC '06: Proc. of the 2006 ACM Symp. on Applied Computing*, pages 1188–1195, New York, NY, USA. ACM.

Lozano, M., Ramos, I., and González, P. (2000). User interface specification and modeling in a object oriented environment for automatic software development. In *IEEE 34th Int. Conf. on Tech. of OO Lang. and Systems*, pages 373–381. IEEE Comp. SP.

Meyers, B. and Oberndorf, P. (2001). *Managing Software Acquisition. Open Systems and COTS Products*. Addison-Wesley.

Nunes, N. (1998). *Object modeling for user-centered development and user interface design: The wisdom approach*. PhD thesis, Univ. de Madeira.

Paterno, F. (1999). *Model-Based Design and Evaluation of Interactive Applications*. Springer.

Pinheiro, P. (2002). *Object Modelling of Interactive Systems: The UMLi Approach*. PhD thesis, University of Manchester.

Roberts, D., Berry, D., Isensee, S., and Mullaly, J. (1998). *Designing for the User with OVID: Bridging User Interface Design and Software Engineering*. New Riders Publishing.