OperettA: A prototype tool for the design, analysis and development of multi-agent organizations

Daniel M. Okouya, Virginia Dignum
Dept. Information and Computing Sciences, Utrecht University
3508 TB Utrecht, The Netherlands
{maatari, virginia}@cs.uu.nl

ABSTRACT
OperettA is a graphical tool that supports the design, verification and simulation of OperA models. It ensures consistency between different design parts, provides a formal specification of the organization model, and is prepared to generate a simulation of the application domain.

Categories and Subject Descriptors
D.3.3 [**Sofware Engineering**]: Design Tools and Techniques

General Terms
Design

Keywords
Agent Organizations, Agent Software Engineering, Tool Support

1. INTRODUCTION
The OperettA is an IDE (Integrated Development Environment) developed to support the design, analysis and development of agent organizations using the OPERA methodology [1]. It is intended to support software engineers and developers in both developing and documenting the various aspects of specifying and designing a multi-agent organization.

The OperA model proposes an expressive way for defining open organizations that explicitly distinguishes between the organizational model and the agents who will act in it. That is, OperA enables the specification of organizational requirements and objectives, and at the same time allows participants to have the freedom to act according to their own capabilities and demands. The OperA framework consists of three interrelated models. The **organizational model (OM)** is the result of the observation and analysis of the domain and describes the desired behavior of the organization, as determined by the organizational stakeholders in terms of objectives, norms, roles, interactions and ontologies. The **social model (SM)** maps organizational roles to specific agents. Agreements concerning the role(s) an agent will play and the conditions of the participation are described in social contracts. The **interaction model (IM)** specifies the interaction agreements between role-enacting agents as interaction contracts.

Cite as: Title, Author(s), *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

The Electronic Institution Development Environment (EIDE) for ISLANDER which results in an AMELI [2] implementation can be seen as similar to OperettA. However, EIDE is developed among other things for the specification of fully regimented institutions, and as such does not meet the OperA's requirements of internal autonomy and collaboration autonomy [1]. OperettA has been implemented following the model driven software development paradigm, which enables the introduction and combination of different formal methods hence enabling the modeling activity through systematic advices and model design consistency checking.

2. THE OperettA ARCHICTECTURE
The Operetta prototype is implemented using MetaEdit+[1], a generic customizable model driven software development environment suitable for prototyping. The prototype incorporates Racer DL[2] reasoning system, SWI-prolog interpreter[3], MCMAS[4] model checker and Brahms[5] as a possible simulation environment. In the following section we present the different features currently included in OperettA, which architecture is depicted in Figure 1.

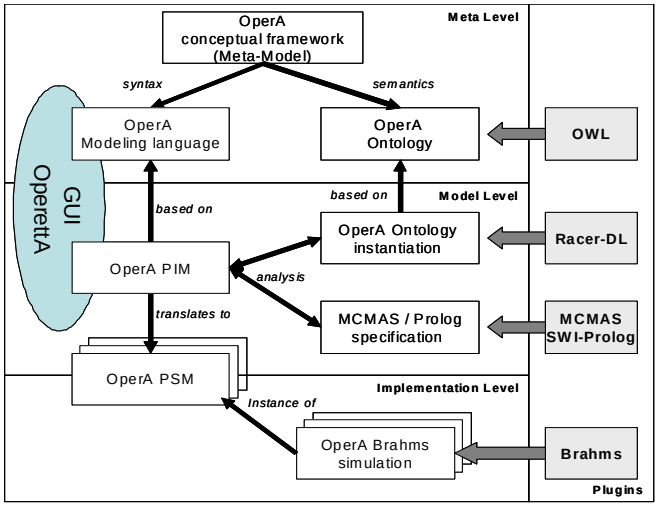

Figure 1 - The OperettA Conceptual Architecture

[1] http://www.metacase.com/
[2] http://www.sts.tu-harburg.de/~r.f.moeller/racer/
[3] http://www.swi-prolog.org/
[4] http://sse.cs.ucl.ac.uk/projects/mcmas/
[5] http://www.agentisolutions.com/

Following the model driven paradigm, OperettA consists of 3 different levels. The **Meta Level** is directly based on the OperA conceptual framework and provides its syntax and semantic specifications. Syntax is derived from the OperA BNF and semantics are defined as an OWL[6] ontology. At the **Model Level** the development environment for OperA OM specifications is defined. It provides a multi-viewed GUI (graphical user interface) and model verification support, as described in section 3. At this level, a Platform Independent Model (PIM) is constructed for the organization. Finally, the **Implementation Level** (under construction) enables the generation of Platform Specific Models (PSM). Brahms is currently used as a simulation environment for organizations in which normative properties of the organization can be verified for different populations with emergent behavior.

3. FEATURES OF OperettA

The current version of OperettA provides graphical capabilities to design and analyze the OM for an application domain. The tool generates an OWL description of the OM for the domain used for consistency checking. This description also enables the integration of organizational information to be used by other (semantic web) based applications or services. Following is a list of the main features of OperettA.

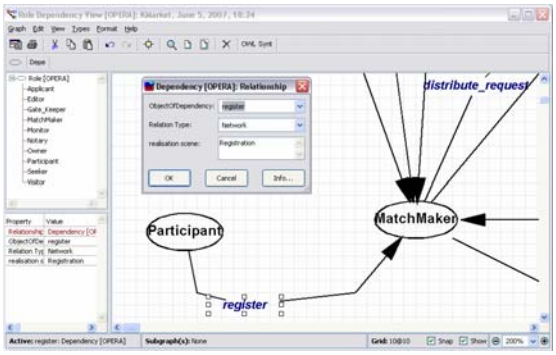

Figure 2: The OperettA Prototype

- **Multiple views with hierarchical integration support.** The tool allows for the edition of multiple hierarchically structured views. Currently the views supported are: social structure view, role definition view, and interaction structure view. In figure 2, the Social Structure view is depicted. The hierarchical nesting of views is provided for modularity support. For instance, the user can define the organization interaction structure with as many layers as necessary to keep each layer manageable in size; thus an infinite nesting of interaction structure is allowed in order to simplify his visualization. It is also possible to choose between using dialog box and using another layer of abstraction to define role behavior separately, while defining the social structure of the organization.
- **Syntax checking.** The tool supports complete syntax checking of the models. Constraints over modeling diagrams are defined through the meta-model. A systematic syntax analysis is applied before closing a dialog box**.** As a general rule in OperettA, syntax checking is applied at design time. Examples of syntax checks are: **(i) Naming:** it is not possible for two entities to have the same name, for example, two roles with the same name; (ii) **Typing:** checks type usage against model definition and domain ontology. For example, checks the type of predicate arguments in pattern and norm definitions. (iii) **Links**: it is not possible to link two interactions scenes without a transition in between.
- **Static analysis.** We use Racer DL to validate the **OWL instance** derived from the PIM against the OperA ontology. This evaluation provides advices on structural properties. For example, if two roles cooperate in a scene but without having a social link, OperettA will advise the designer to define such a dependency, suggesting an adequate dependency type. More generally checks could be classified in: (i) **Intra-models properties**: a role- dependency is defined for an objective or sub-objective that is not defined for either of the two roles. (ii) **Inter-models properties:** A role-dependency requires an interaction scene which is not part of the interaction structure.
- **Dynamic and normative analysis.** It enables the verification of the consequences of norms and organizational constraints. For example: it will indicate a possible inconsistency for a role with objective *'buy food'* and a norm *'FORBIDDEN (spend-money).* Note that these checks do not yet include checking different interpretations and violations.
- **Report Generation** Containing the full model, including graphics and index over all the design entities. Documents and images can be saved in different formats.

4. CONCLUSION AND FUTURE WORK

OperettA provides a number of features that are extremely useful in the modeling of complex environments that require the integration of organizational structures and individual (emergent) behaviors. We are currently working on the integration of OperA and Brahms for the specification and evaluation of integrated models [3]. In the future, other implementation platforms (such as Repast) will be supported, and the tool will be integrated in an OperA open environment that enables the participation of heterogeneous agents in the same organization.

Acknowledgements. This research is funded by the Netherlands Organization for Scientific Research (NWO), through Veni-grant 639.021.509. The authors are grateful to C. Tick, R. van der Meulen and D. Acay for their contributions to the project.

5. REFERENCES

[1] V. Dignum, F. Dignum, J.J. Meyer (2004): An Agent-Mediated Approach to the Support of Knowledge Sharing in Organizations. *Knowledge Engineering Review*, Cambridge University Press, 19(2), pp. 147-174, 2004.

[2] Esteva, M., Rosell, B., Rodriguez-Aguilar, J. A., and Arcos, J. L. (2004). AMELI: An Agent-Based Middleware for Electronic Institutions. In: *Proc. AAMAS'04*. 236-243

[3] B.J. van Putten, V. Dignum, M. Sierhuis, S. Wolfe (2008): Integrating Organizational and Emergent Views on Agent-Based Modeling. *Submitted*.

[6] http://www.w3.org/TR/owl-ref/