CEUR Proceedings of the 5th International i Workshop (iStar 2011)*

Tropos modeling, code generation and testing with the Taom4E tool

Mirko Morandini, Cu Duy Nguyen, Loris Penserini, Anna Perini, and Angelo Susi

FBK-CIT, Trento, Italy,
{morandini,cunduy,penserini,perini,susi}@fbk.eu,

Abstract. We present Taom4E, a tool for the development of software following the agent-oriented software engineering methodology Tropos. The Eclipse plug-in Taom4E supports visual goal modelling and includes functionalities for code generation and testing for goal-oriented agent platforms.

Key words: Goal-oriented development, modelling tools, code generation.

Introduction

The agent-oriented software engineering methodology Tropos, which roots in *i** for requirements modelling, becomes increasingly popular. Modelling tools are essential for such a methodology. We present Taom4E[1] (Tool for Agent Oriented visual Modelling for the Eclipse platform) [1], a tool which supports goal-oriented modelling, code generation and testing for goal-directed systems, following the Tropos methodology and its extension Tropos4AS. Taom4E supports the Tropos modelling activities for early and late requirements analysis and architectural design and contains extensions for an automated agent-oriented implementation and testing.

The Taom4E Tool

The Taom4E tool provides a graphical model editor, based on the Tropos metamodel, as defined in [2], and allows various views on this model. Combining Tropos actor- and goal diagrams, representing strategic dependency (SD) and strategic rationale (SR) diagrams in *i**, the *Mixed Diagram* is the main graphical representation in Taom4E, throughout the supported development phases. In the Tropos early and late requirements analysis phases, actor diagrams, displaying the dependencies between actors, can be graphically created. Actors are

[1] Taom4E is developed by the Software Engineering unit at Fondazione Bruno Kessler (FBK), Trento. The current version 0.6.3 is downloadable under GPL license from the tool homepage http://selab.fbk.eu/taom.

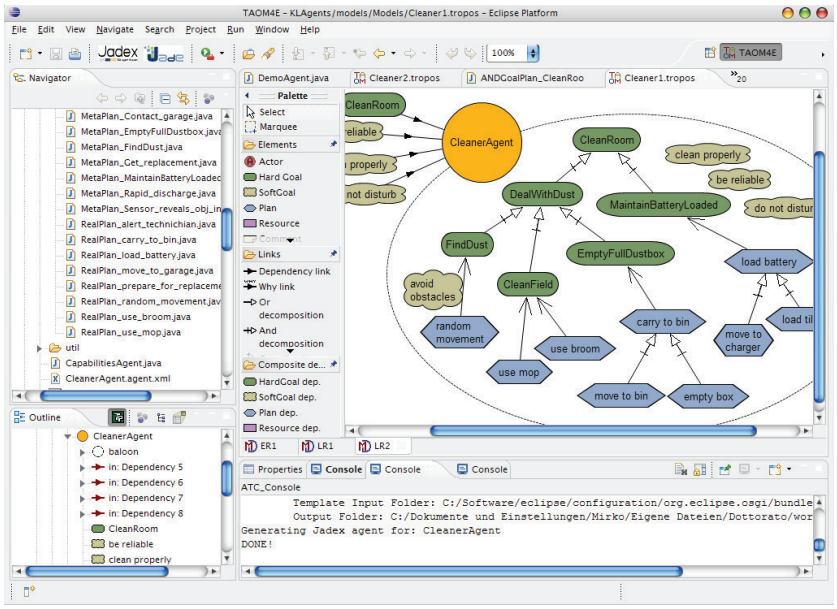

Fig. 1. Interface of the TAOM4E Eclipse plug-in.

detailed in a goal diagram, which is graphically shown in a "balloon" associated to the actor. In this balloon, delegated goals are visualized and can be decomposed to more detailed goals, operationalized by plans or delegated to other actors. An actor representing the system-to-be can be furthermore detailed to a multi-agent system, in an *architectural design diagram*. These development phases provide different views on a single TAOM4E model, thus ensuring traceability, but not yet preserving the model history. Figure 1 shows the principal view of the tool front-end. Models are edited according to the Tropos graphical notation, provided by the *Palette* window. Tabs are used to switch between model views of the different phases (ER, LR,...). Visualized diagrams are views on the whole model, whose components are displayed in the *Outline* window. In the *Navigator* in the left-hand side window, the folders with the output of the ***t2x*** code generation tool are actually shown, for each actor in the system.

Architecture and Extensions

Various extensions to the TAOM4E model editor are provided. The ***t2x*** code generation tool maps goal models to a goal-directed implementation on the Jadex BDI agent platform, explicitly preserving goal models at run-time and providing the proper middleware for navigating this model and acting according to it. Agent code can be generated from the graphical interface, and the implemented prototypes are executable directly from the Eclipse user interface. Moreover, extensions for environment and condition modelling according to the Tropos4AS

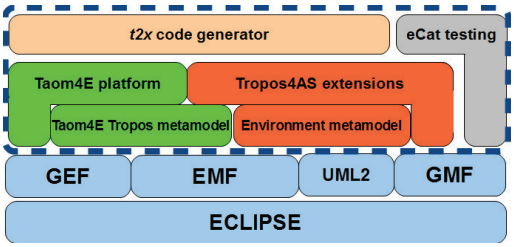

Fig. 2. Tool architecture: Taom4E and its extensions.

framework [3], are available. Figure 2 summarizes the architecture of the toolset, which consists of various plug-ins for the Eclipse development platform, built on the Eclipse EMF and GEF frameworks for model-driven development. They are installable through the Eclipse installation manager by adding the link provided on the tool homepage. The models are saved in XMI format and can be accessed by other applications through an EMF-based interface.

Goal-Oriented testing has been proposed as a complementary activity to goal-oriented modelling and code generation, with the aim to support testing and validation along the process phases [4]. The *eCAT* tool derives test cases directly from Tropos goal models and uses them to test implemented agents, using FIPA-standard messaging services.

Conclusion

The Taom4E tool was used in several projects that involved Tropos, and in various university courses on requirements engineering and on goal-oriented development. As a design limitation, the tool can only handle a single model per file, therefore changes to one view of the model are projected to the whole model. Moreover, no automatic diagram repositioning is implemented. For addressing this and other issues, future work concerns the reimplementation of the graphical front-end with state-of-the-art technology. Also, we are working on the extendibility of the graphical model editor with additional concepts.

References

1. Morandini, M., Nguyen, D.C., Perini, A., Siena, A., Susi, A.: Tool-supported development with tropos: The conference management system case study. In: Agent Oriented Software Engineering VIII. Volume 4951 of LNCS. (2008) 182–196
2. Susi, A., Perini, A., Giorgini, P., Mylopoulos, J.: The Tropos Metamodel and its Use. Informatica (Slovenia) **29**(4) (2005) 401–408
3. Morandini, M., Penserini, L., Perini, A.: Towards goal-oriented development of self-adaptive systems. In: SEAMS '08: Workshop on Software engineering for adaptive and self-managing systems, ACM (2008) 9–16
4. Nguyen, D.C., Perini, A., Tonella, P.: ecat: a tool for automating test cases generation and execution in testing multi-agent systems. In: AAMAS. (2008) 1669–1670