Automated Mapping from Goal Models to Self-Adaptive Systems

Mirko Morandini  
FBK-IRST  
Via Sommarive 18  
I-38050 Trento, Italy  
morandini@fbk.eu

Loris Penserini  
Universiteit Utrecht  
Padualaan 14, De Uithof  
Utrecht, The Netherlands  
loris@cs.uu.nl

Anna Perini  
FBK-IRST  
Via Sommarive 18  
I-38050 Trento, Italy  
perini@fbk.eu

Abstract

Self-adaptive systems should autonomously adapt at run time to changes in their operational environment, guided by the goals assigned by their stakeholders.

We present a tool that supports goal-oriented modelling and generation of code for goal-directed, self-adaptive systems, supporting Tropos4AS, an extension of the software engineering methodology Tropos.

1 Introduction

Goal-oriented software engineering approaches provide suitable design abstractions to capture the requirements of complex systems that have to autonomously adapt to their operational environment at run-time, using mentalistic abstractions such as actors, goals, and alternatives [1].

In this paper we illustrate a tool that supports the modelling of adaptivity features and code generation, following Tropos4AS, an extension of the agent-oriented software engineering methodology Tropos [4] for self-adaptive systems [2]. Up to our knowledge, no goal oriented methodology provides tool support to designers to model requirements for adaptive systems and to generate code from them.

In detail, we extended the Tropos modelling tool Taom4E to deal with adaptivity modelling and introduced the tool t2x \(^1\), which covers the implementation phase of Tropos4AS by providing an automated mapping from goal models to code.

The mapping, proposed in [3] and further extended in [2] for adaptivity concepts, creates a Belief-Desire-Intention (BDI) agent-based implementation for a self-adaptive system, on the target platform Jadex [5]. In the next sections we briefly present the tool features on a simple cleaner robot case study. The robot’s goal is to clean a room, dealing also with battery loading and dust box emptying.

2 Tropos for Self-Adaptive Systems

Tropos [4] is an agent-oriented software development methodology that proposes the use of knowledge level concepts such as actor, goal, plan and dependency along the whole software development process, with an emphasis on alternatives modelling. It covers all software development phases, from requirements analysis to design and implementation, and uses a specific visual modelling language supported by Taom4E.

In [2] we enrich Tropos to model self-adaptive systems by: the introduction of goal types and goal relationships to characterise the run-time behaviour of goals; the definition of the environment which surrounds the system; and the definition of conditions to correlate goal achievement with the environment. These activities are supported in our extensions to Taom4E.

3 Automated Mapping

3.1 Specification for the mapping

\(^1\)Actual versions of t2x (Tropos to JadeX) and the modelling tool Taom4E are available at http://se.fbk.eu/en/tools.
ent goal types (achieve, maintain, perform), whose def-
inition enriches the expressiveness of goal models, mak-
ing it possible to deal with goal creation and achieve-
ment conditions at run-time (Figure 1).

Figure 1. Goal model and Environmental model, with the correlations between them.

Environment entities are mapped to the agent’s be-
lief base and also directly to Java classes, using avail-
able UML tools. In the Cleaner robot example (Fig-
ure 1) a condition is modelled between the goal Main-
tainBatteryLoaded and the entity battery, relating the
state of the goal to a state in the environment. The use
of different types of conditions (creation, target, main-
tain, . . . ) allows to guide the goal achievement process,
triggering or guarding transitions between goal states.

3.2 The tool t2x

On the Jadex platform, agents are implemented by
defining their beliefs, goals and plans in an Agent Def-
inition File (ADF). The tool t2x analyses an actor’s
goal model and generates an ADF with goals and plans,
resembling the source goal model hierarchies. The
plans in means-end relation to leaf goals are mapped
also to JAVA files, where the implementation of each
single plan can be carried out. Figure 2 shows the de-
inition of a goal of type achieve, which triggers the ex-
cution of associated plans, until reaching the defined
target condition.

Conditions are directly mapped to a goal definition.
They use boolean formulas to link the goal achieve-
ment process to facts in the belief base, which repre-
sent the environment entities, implemented in JAVA
classes. Also intra-goal relations, like inhibition, are
directly mapped to this goal definition.

The agent generated by t2x is ready to be executed
and exhibits the modelled behaviour, related to goal
dispatching and plan execution orders.

<achievegoal name="EmptyFullDustbox">
<targetcondition>
$beliefbase.dustbin.empty()
</targetcondition>
...
<deliberation>
<style attr="Inhibits">CleanField</style>/>
</deliberation>
</achievegoal>

Figure 2. Excerpt of a goal definition gener-
ated for the Jadex agent definition file.

4 Conclusions

In this demo we present an extension of TAOM4E
and the tool t2x, to support the development of self-
adaptive systems. Our development framework, Tro-
pos/AS, is based on both an extended version of the
Tropos agent oriented software engineering methodology and a set of mapping specifications from the
extended goal models to an implementation as BDI agents. The tools provide an important contribution
towards two directions: extending the TAOM4E tool to allow designers for a visual modelling of the require-
ments of self-adaptive systems, and providing an auto-
mated mapping process for the generation of BDI agent
code for the Jadex platform.

References

[1] H. J. Goldsby, P. Sawyer, N. Bencomo, D. Hughes, and
B. H. C. Cheng. Goal-based modeling of dynamically
adaptive system requirements. In ECBS’08, Belfast,
Northern Ireland, April 2008.
goal-oriented development of self-adaptive systems. In
Workshops on Software Engineering for Adaptive and
Self-Managing Systems (SEAMS’08) at ICSE, Leipzig,
[3] L. Penserini, A. Perini, A. Susi, M. Morandini, and
J. Mylopoulos. A Design Framework for Generat-
ing BDI-agents from Goal Models. In 6th Int. Conf.
on Autonomous Agents and Multi-Agent Systems (AA-
MAS’07), Honolulu, Hawaii, 2007.
High variability design for software agents: Extending
A bdi reasoning engine. In J. D. R. Bordini, M. Das-
tani and A. F. Seghrouchni, editors, Multi-Agent Pro-