Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

An Extended Goal-oriented Development Methodology with Contextual Dependability Analysis

Danilo F. Mendonça

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof. Dr.ª Genaína Nunes Rodrigues

Brasília
2015

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenadora: Prof.ª Dr.ª Alba Cristina Magalhaes Alves de Melo

Banca examinadora composta por:

Prof. Dr.ª Genaína Nunes Rodrigues (Orientadora) — CIC/UnB
Prof.ª Dr.ª Vander Alves — CIC/UnB
Prof. Dr. Luciano Baresi — Politecnico di Milano

CIP — Catalogação Internacional na Publicação

Mendonça, Danilo F..

An Extended Goal-oriented Development Methodology with Contextual Dependability Analysis / Danilo F. Mendonça. Brasília : UnB, 2015.
28 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2015.

1. LaTeX, 2. metodologia científica

CDU 004.4

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

An Extended Goal-oriented Development Methodology with Contextual Dependability Analysis

Danilo F. Mendonça

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof. Dr.ª Genaína Nunes Rodrigues (Orientadora)
CIC/UnB

Prof.ª Dr.ª Vander Alves
CIC/UnB

Prof. Dr. Luciano Baresi
Politecnico di Milano

Prof.ª Dr.ª Alba Cristina Magalhaes Alves de Melo
Coordenadora do Mestrado em Informática

Brasília, 30 de janeiro de 2015

Dedicatória

Agradecimentos

Resumo

A static and stable operation environment is not a reality for many systems nowadays. Context variations impose many threats to systems safety, including the activation of context specific failures. Goal-oriented software-development methodologies (SDM) adds the 'why' to system requirements, i.e., the intentionality behind system goals and the means to meet then. Contexts may affect what requirements are needed, which alternatives are available and the quality of these alternatives, including dependability attributes. In order to allow a formal and probabilistic analysis of systems affected by context variation and elicited with Goal-Oriented Requirements Engineering (GORE) approach, we have proposed an extension to the TROPOS goal-oriented methodology to include dependability constraints to goals and to provide a more precise and formal requirements verification by translating a contextual goal-model annotated with a behavioural regular expression into a PRISM probabilistic model to be checked against properties defined with the Probabilistic Computation Tree Logic (PCTL). We evaluated the proposal with a case study of a Mobile Personal Emergency Response System (MPERS).

Palavras-chave: LaTeX, metodologia científica

Abstract

Keywords: LaTeX, scientific method

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introduction

1.1 Problem definition

According to Lamsweerde, a poor requirements engineering (RE) is the major source of system failures. Lack of user involvement, requirements incompleteness, changing requirements, unrealistic expectations and unclear objectives are common causes [AXEL]. Goal-oriented requirements engineering (GORE) has gained the attention of both academic and industrial practitioners due to its ability to systematically model the intentionality behind system requirements. More than just presenting the 'what' and the 'how', goal models also express the 'why' of different requirements to exist. Its simple graphical notation allows non-technical stakeholders to take part in the analysis process and have a clear view of the system-to-be. Finally, automated model verification should avoid inconsistencies in the requirements specification.

In traditional GORE methodologies [GORE CA comparison], contribution analysis are based on domain knowledge about the positive, neutral (implicit) or negative impact of a given system alternative to one or more system goals, generally a qualitative goal. By comparing the overall contribution of two or more alternatives, a decision is made about which one should be adopted for the system-to-be. For instance, if one goal is to communicate with a remote user mobile, alternative means for the notification agent could be to send a SMS, an internet based message or a voice call. These alternatives may contribute with different values for qualitative goals such as 'reliable delivery', 'fast delivery', 'convenient delivery', etc.

The problem with this approach is threefold. First, it is based on domain knowledge information that may not exist or may not be precise and reliable. As a consequence, the decision of which alternative to use, either at design time or at runtime, may be biased and lead to unexpected violations - the selected alternative was actually unable to fulfil its qualitative goals. Second, it is limited to a static representation without any dynamic

information that could be used to verify more complex attributes depending on the bigger picture of the system behaviour and its many nuances such as optional, alternative and interleaved executions. Finally, contribution links are deterministic. As such, the design decision based on contribution analysis is reduced to a simple sum comparison of the concurring alternatives with no support for probabilistic verification.

Additionally to the contribution analysis limitation, the context in which systems operate may not be static. Mobile and pervasive computing, among others, are examples of new computer paradigms for which the environment is not static, but dynamic. Battery, signals strength, component's availability and the quality of physical resources and relevant information such as the user geographic location may vary through time, posing a new sort of challenge to the development of socio-technical systems based on these paradigms. The contextualization of the informations gathered at RE phase becomes imperative once its validity may be threatened by changing environment conditions [Finkelstein, CGM]. Accordingly, any improvement for the GORE contribution analysis may have to consider multiple operational contexts.

1.2 Proposed solution

In order to provide a more solid and precise approach for the non-functional verification of different system alternatives and to improve the GORE contribution analysis, we propose the extension of the TROPOS goal-oriented software development methodology with a probabilistic model checking (PMC) approach that has already been explored and is supported by tools such as PRISM model checker[genaína PMC, PRISM]. The resulting verification model should represent activities that fulfils the root goal (global) or any lower level goal (local) and its elicited alternatives. This model should then be checked for properties that will provide estimations for non-functional requirements such as reliability, availability, performance and power consumption.

The PMC technique used by this proposal requires a behaviour system specification. As the goal model proposed in TROPOS is static, no information regarding achievement/execution order, cardinality and priority of goals/tasks is available, except the activity diagram for the detailing of an agent's single capability behaviour and the sequence diagram for agents interaction. Nonetheless, this problem was tackled by Dalpiaz et al. with a regular expression language associated to goals, tasks and dependencies in a goal model to express, e.g., how many times the same goal should be achieved and the execution order of system tasks [RGM]. We have used this proposal to fill the gap between the static goal model and its dynamic representation. Finally, this dynamic view of the goal model is translated to a probabilistic model following PMC technique. Our work

uses the PRISM model checker and its language for this purpose and the Probabilistic Computation Tree Logic (PCTL) for property definition.

To address a dynamic context of operation, context variables and its effects over goals, means and metrics should be parametrized to produce a formula that can check the system and its alternatives for different contexts of operation. This verification, performed as part of the Validation & Verification (VV) phase in RE, should anticipate any (contextual) violations of non-functional constraints. Treating a detected violation at design time may correspond to actions such as making a different choice for underlying components used by this alternative's tasks, optimizing its behaviour specification or even the disposal of this alternative as a means to satisfy its goal if there is at least one other valid alternative. PMC technique also allows the identification of system alternatives with more influence on each metric through sensitive analysis.

Runtime self-adaptation is beyond the scope of this work. However, based on the contextual analysis provided by the CGM and the enriched non-functional and dependability analysis provided by the verification of different alternatives using the PMC technique, it should not be difficult to extend the approach with the additional monitoring, planing and execution capabilities of a self-adaptation loop and have a self-adaptive architecture and mechanism reflected upon its runtime goal model requirements. These concerns should be addressed in future work.

1.3 Evaluation

This proposal was evaluated with the application of the extended TROPOS methodology to the development of a Mobile Personal Emergency Response System (MPERS). This system may be seen as a body area network (BAN) with extended functionalities related to pervasive emergency response based on mobile computing [BAD]. Instead of a home or hospital static environment, the MPERS is conceived to allow patients with different health risk degrees to maintain mobility while they are monitored and assisted. If a medical emergency is detected, a geolocation feature should point out the location where the emergency response team must be addressed to. The MPERS features were based on real emergency response systems available at the industry and also at the BAN explored in previews work by Fernandes[Fernandes].

The evaluation process was focused in revealing the major benefits and limitations of the extended TROPOS proposal. Time to market and complexity is an important aspect for any software development methodology. Also, the soundness and precision of the proposed probabilistic verification is crucial and must be evaluated as they should not result in mislead decisions about which alternatives should be used by the system. Instead,

they must anticipate any violation that could lead to a system failure, specially severe or catastrophic failures, giving analysts valuable information about where the system requirements and specification should be tailored and improved.

1.4 Summary of Contributions

This section summarizes the contributions of this proposal.

1. A new contribution analysis approach for the TROPOS Goal-oriented software development methodology.
2. Conversion rules among different decomposition and runtime constraints in a runtime goal model to a PRISM probabilistic model.
3. Inclusion of context effects over goals, means and metrics in the PRISM model using appropriate constructs and parameters for each case.
4. A parser implementation for the regular expression (regex) language used in runtime goal-models to specify temporality, cardinality and goals priority.
5. An automatic generation of the PRISM model representing activities from a TROPOS goal-model annotated with the runtime regex and graphically modelled using the TAOM4E tool that supports TROPOS methodology.

1.5 Document organization

This dissertation is organized as follows. Chapter 2 presents the base concepts of this work and the most important related works. Chapter **??** details the problem tackled by this proposal. Chapter 4 presents the new extended TROPOS methodology, the rules for the translation between the contextual goal model and the probabilistic verification model, the parser for the runtime regex and finally the implementation approach for the automatic generation of the probabilistic model in PRISM language. Chapter **??** evaluates the proposal and describes its benefits and limitations. Finally, Chapter **??** concludes this work with final considerations about the current proposal, related proposals and our future work.

Capítulo 2

Baseline

2.1 Goal-oriented Requirements Engineering

Goal-oriented requirements engineering brings forward the intentionality behind system requirements. More than just presenting the *what* and the *how* of a system-to-be, it provides the justification for each requirement, that is, they also present the *why*. Through a directed graph tree that begins with a root goal, goals are connected trough decomposition links. Root and higher level goals are related to strategical concerns, while lower level and leaf-goals are related to technical and operational features of the system.

The main purpose of a goal model is to support the early process of RE, including the elicitation of social needs and dependencies, the actors involved in delivering functionalities and resources, the decomposition of higher-level goals into more granular and detailed requirements chunks, the operationalization through means-end tasks and finally the comparison between different alternatives for the system-to-be. A goal model is said to be valid and complete if it follows all its syntactic rules and if all system goals are either decomposed, delegated to other actors or fulfilled by operational system tasks.

Three frameworks/methodologies, namely KAOS, i* and TROPOS, represent the foundations for the goal model analysis used by a variety of other proposals [KAOS, i*, TROPOS]. Despite some differences among their syntax, they all share a set of core concepts:

Entities

- **Actor:** an entity that has goals and can decide autonomously how to achieve them. They represent a physical, social or software agent. E.g.: A patient, an emergency center, a doctor and a Mobile Personal Emergency System running in patient's smartphone.

- **Goal:** are actors' strategic interests. A goal with a clear-cut criteria for its satisfaction is called a hard goal. In opposition, softgoals has no clear-cut criteria for deciding whether they are satisfied or not and are usually associated to non-functional requirements of an actor. E.g.: vital signs are monitored, emergency is detected, emergency center is notified (hard goals) and system availability, detection precision, emergency awareness (softgoals).

- **Task:** an operational means to satisfy actors' goals. E.g.: monitor temperature sensor, persist vital signs data, request emergency assistance.

Relations

- **AND/OR Decomposition:** a link that decomposes a goal/task into sub-goals/sub-tasks, meaning that all (at least one) decomposed goal(s)/task(s) must be fulfilled/executed in order to satisfy its parent entity.

- **Means-end:** a means to fulfil an actor's goal through the execution of an operational task by the same actor.

- **Contribution link:** a positive or negative contribution between a given goal/task to a softgoal. Contribution links are used for deciding between alternative goals/tasks at design time (contribution analysis).

2.2 TROPOS Goal-oriented Software Development Methodology

TROPOS is a GORE methodology based on the i* framework. Its main improvement is the addition of new phases of requirements engineering and system design, namely [TROPOS]:

- Late requirements engineering: Beyond the social dependency modelling with actors diagrams representing stakeholders and their needs in early

requirements phase, a late requirements phase focuses on the system actor analysis. In this phase, system goals are inherited from stakeholders needs and represent both functional and non-functional requirements. Each goal has to be further decomposed in more granular sub-goals, delegated to other actors or to be fulfilled by means-end tasks.

- Architectural design: In this phase, new actors representing sub-systems are created to fulfil different system goals. The idea is to shape the solution using a multi-agent architecture style instead of a monolithic system approach. Data and control interconnections are represented as dependencies.

- Detailed design: The last phase is characterized by the specification of agent capabilities and interactions though UML activity and sequence diagrams. Also, the implementation platform and other specific implementation details are addressed in order to directly map the design to system code.

Implementation phase is also specified by TROPOS methodology, but it is out of the scope of this work as our objective is to improve the analysis and the solution that will be later implemented.

2.3 Contexts

Context may be defined as the reification of the environment that surrounds the system operation [FINKElSTEIN]. Contexts, as already stated, may not be static, but dynamic. A system has no control over its context of operation. Accordingly, a system must be able to support different contexts of operation without violating its goals. Moreover, systems should be able to monitor the state of its surrounding environment and decide which alternative will be used regarding both the availability of that alternative and the optimization of non-functional requirements.

In GORE, dynamic contexts may affect what goals a system have to reach, the means available to meet them and also the quality achieved by each alternative[CGM]. Root goal and higher level strategical goals are not contextualized as they represent the main purpose of a system [Finkelstein]. As these goals are decomposed in more granular sub-goals, a context condition may dictate:

1. If the goal is required for that context, limiting 'what' a system should do;

2. If a sub-goal or task is adoptable, limiting the 'means' to fulfil a required goal;

3. The positive, neutral or negative contribution of using some goal or task to another goal, usually a qualitative softgoal;

The third effect is the main focus of this work, as it is related to the GORE contribution analysis that we aim to improve.

2.4 Dependability Analysis

The concept of dependability is related to dependence and trust as well as the ability of a system to avoid failures that are more frequent and more severe than certain threshold [AVIZIENIS]. According to Avizienis et al., dependability encompasses the following attributes:

- Availability: readiness for correct service.

- Reliability: continuity of correct service.

- Integrity: absence of improper system alterations.

- Safety: absence of catastrophic consequences on the user(s) and the environment.

- Maintainability: ability to undergo modifications and repairs.

Correctness is opposed to failures. A failure is a perceived deviation from system expected behaviour that may have variable degree of consequence on the user(s) and the environment. Failures are caused by a specification faults or specification violations. In the first case, either the goals or the means to fulfil then are incorrect or incomplete. In the second case, system implementation did not followed its operational specification due to a faulty implementation or a deviation from normal behaviour took place in one or more components involved in the execution.

The scope of this work is restricted to specification violations, i.e., we assume that a system specification is valid and has no false assumptions, incompleteness or inconsistencies. Failures are restricted to anomalous behaviour of the technical components or social actors participating in the execution of system tasks. Accordingly, our approach is focused on dependability attributes that should be estimated and compared to their required constraint values and also to the sensitive analysis of how different system parts contribute to the overall value of those attributes. Sensitive analysis may be considered analogous to the original GORE contribution analysis.

2.5 PRISM Probabilistic Model Checker

A model checking is a formal method that aims to automatically verify if a system model meets its specification for defined properties. Probabilistic and state based model checking supports the verification of finite-state probabilistic models such as discrete-time Markov chain (DTMC), continuous-time Markov chain (CTMC) and Markov decision process (MDP), among others. Different types of properties may be defined to verify a system model and reveal meaningful information for systems audition and validation.

The PMC technique used in this approach is supported by the PRISM model checker tool [PRISM]. PRISM allows the modelling and analysis of

systems which exhibit random or probabilistic behaviour. The decision of using PRISM as the probabilistic state based model checker was due to the number of successful case studies that have used this tool, indicating its maturity [PRISM CS], and also due to its rich environment that is able to represent different kinds of probabilistic models and their evaluations.

PRISM is suitable for many different kinds of model evaluations depending on the abstraction level, the type of probabilistic model and the PCTL properties to be analysed. PRISM language offers a rich set of constructs that may represent system modules and components, among others architectural and design configurations. Both qualitative and sensitive analysis are available.

As it will be explained in later sections, goal models may be easily extended with the proper information required for the verification of some important dependability attributes. The objective is to anticipate non-functional dependability violations and to support the decision of which alternatives to use in the system-to-be. A model checking technique should be used for dependability analysis as long as:

- A formal system model may be built;
- Properties representing dependability attributes may be defined;
- The analysis overhead is justified, e.g., by its criticality.

Finally, PRISM also supports a parametric model verification. That is, instead of providing the final evaluation for a given property, models may use parameters instead of initialized variables and the verification will output a parametric formula whose evaluation will check the model for any valid combination of parameters values.

Capítulo 3

Related Work

3.1 Contextual Goal Model

The Contextual Goal Model (CGM) [CGM] proposes the contextualization of required goals, adoptable means (goals/tasks) and contribution links values. The main benefit of this work is to enrich the original goal model with the contextualization of entities and relations affected by context variations and to provide a rationale for context analysis. In contrast, the main problem tackled by the our work is the verification of non-functional attributes such as dependability attributes that needs a more precise and less biased approach instead of the existing contribution analysis that is based on analysts direct evaluation of the forward impact between goals/tasks and softgoals.

In this regard, the CGM provided more realistic and precise contribution analysis contextualized by environment conditions, but did not change the nature of the contribution analysis process. Our work has benefited from the CGM conceptual model and has extended the non-functional GORE analysis with a context-dependent formal verification, i.e., that includes different context effects in the probabilistic model used by the PMC to contextually estimate the values of required non-functional attributes of the system and provide a reliable decision criteria for the selection of concurring alternatives in the goal model given different contexts.

3.2 Awareness Requirements

Souza et al. [AwaReq] proposed the Awareness Requirements (AwReq) as a meta-requirement in a goal model, i.e., AwReq specify the success/failure rate and temporal constraints for other requirements in the model, including goals, tasks, domain assumptions and other AwReqs (*-meta-requirement). The objective is to enrich the original goal model and provide constraints for system behaviour and to support self-adaptation, as runtime AwReq violations should be addressed by corrective actions. AwReq are formalized by a temporal logic formula, namely the Object Constraints Logic with Temporal Message (OCLtm).

Despite its contribution to the specification of meta-requirements in goal models, AwReq do not provide an approach to analyse and validate its meta-requirements before system implementation and monitoring. Original GORE contribution analysis could be used to define the impact of a given alternative to some attribute or value composing one or more AwReqs. However, the paper have only mentioned the formalization and monitoring through code instrumentation. In contrast, our approach relies on model based verification of meta-requirements similar to AwReq through probabilistic model checking technique that can be performed at design time and provide alternative design decision criteria and anticipate violations that must be treated before implementation.

3.3 Runtime Goal Model

Despite the use of goal models to support the runtime monitoring and adaptation, Dalpiaz et al. argued that these works are 'using design artefacts for purposes they are not meant to, i.e., for reasoning about runtime system behaviour'. As such, they proposed a conceptual distinction between the static goal model, named Design Goal Models (DGM), and the Runtime

Goal Model (RGM) that extend DGM with 'additional state, behavioural and historical information about the fulfilment of goals' [RGM].

The main purpose of the RGM approach is to provide the proper specification of behaviour information among system goals. RGM defines a class model, while the Instance Goal Model (IGM) captures instance states of runtime monitored goals that must conform to its class specification. IGM are useful to have an instance representation of the RGM provided by the monitoring of the activities involved in fulfilling system goals. If the monitored IGM violates the RGM, then a corrective action should take place. Our work has benefited from the RGM specification language by using the proposed regular expression to have a behaviour specification and generate the probabilistic model. Instance representation is out of the scope of our proposal.

3.4 Dependability Contextual Goal Model

The current work has been preceded by another proposal concerning goal-oriented requirements engineering, dependability analysis and dynamic contexts, namely the Dependability Contextual Goal Model (DCGM) [DCGM]. The contribution was focused on both dependability requirements and estimations based on declarative fuzzy logic rules and a variable context of operation.

In DCGM, a failure classification scheme was used to classify the consequence level and domain of failures in achieving system goals. This process lead to the definition of dependability constraints that must be achieved by the means-end tasks used to fulfil leaf-goals in specific contexts of operation, i.e., to the specification of contextual dependability requirements. These requirements inherited the same concept of the AwReq, but instead of being static, they could be associated to a context condition. Another facet of the DCGM is the contextual failure implication, which consisted of a dependabi-

lity specific GORE contribution analysis supported by fuzzy logic to define IF-THEN rules between context conditions and the level of a dependability attribute, e.g., availability and reliability.

The main drawback of this proposal was the lack of scalability, as declarative rules must be provided for different goals, attributes and contexts, proving to be a time-consuming task for the analysts. A second problem was the subjectivity of the rules, as they were based in domain knowledge that was also used to shape membership functions. This problem, as much as in GORE contribution analysis, lead to the idea of coupling a more precise and reliable verification approach such as the PMC technique. Still, the idea of a failure classification and the specification of contextual non-functional requirements were kept.

3.5 Formal TROPOS

The idea behind the formalization of a goal model, as proposed by Formal TROPOS [FTROPOS], is to provide a verifiable specification of sufficient and necessary conditions to create and achieve intentional elements like goals, tasks and dependencies in the model and invariants for each of these elements. In addition to this, new *prior-to* links describe the temporal order of intentional elements and cardinality constraints may be added to any link in the model. Finally, Formal TROPOS uses a first-order linear-time temporal logic as a specification language.

The nature of the verification of a goal model with Formal TROPOS specification is different from the PMC used by our work. Formal TROPOS aims to provide the information required for a consistency verification of the model. The verification is not only for the abstract TROPOS syntax, but also to domain specific information of how intentional elements are created and fulfilled in time. Once the model starts to have more elements and relations,

its consistency checking becomes non-trivial, justifying the use of a formal specification that can be verified by a model checker tool.

In our work, PMC technique is used to build a probabilistic model representation of the goal model enriched by RGM behaviour specification and enable the verification of properties that depend on how activities in the model are organized in terms of time, cardinality and priority and how each activity individually contributes to the property being verified. For instance, if power consumption is to be checked, each activity has to be associated to a power consumption unit and the global consumption value is evaluated considering any non-determinism specified in the execution workflow. Thus, even if the Formal TROPOS language allows the specification of dynamic aspects of a goal model, it is tailored for consistency checking of requirements and not for the verification of dependability and other non-functional requirements.

Capítulo 4

Proposal

In the PMC technique adopted by this proposal, a behavioural specification, usually provided by UML activity and sequence diagrams, are manually converted to a probabilistic model in PRISM language. As a goal model goes from strategical root goal to operational leaf-goals, and each leaf-goal describes a desired state reachable by either a delegation to other actor or by a operational task, then a behaviour specification as proposed by the RGM may be seen as an activity diagram and be used to generate a probabilistic model in PRISM language. This allows the model checking of the corresponding goal model as a set of activities for which temporal and other behaviour aspects are specified by the runtime regex of the RGM.

- Making a different choice for underlying components: In some cases the replacement of a technical component for another of the same class can improve the quality of how they achieve their goal. For instance,

- Behaviour optimization: The quality may also depend on the pattern used for the activities execution. The specification of a different pattern may eliminate the non-functional violation.

- Contextualizing the alternative: An alternative may only violate a NFR in specific contexts. In this case, different valid alternatives may be used according to the context of operation.

- Alternative disposal: If the alternative is in absolute violation or if its validity is restricted to contexts that have at least one other valid alternative, this branch can be eliminated from the model.

To evaluate the current proposal with the MPERS case study, we have used the a discrete-time Markov chain (DTMC) probabilistic model and focused on the verification of properties related to dependability, i.e., the reachability of the final success state of a set of goal model activities that represents:

- if the set is composed of the minimum set of activities that satisfies the root goal: its global reliability;
- if the set is composed of the minimum set of activities that satisfies any lower-level goal: its local reliability.

Dependability analysis is used to provide information about different dependability attributes related to system failures. These metrics may be specified as non-functional requirements for isolated system functionalities or for the whole system. Instead of softgoals, we use meta-requirements over functional goals with clear-cut quantitative criteria such as '99.999%' reliable - a probabilistic value to make it compatible with the PMC estimation results.

To reduce the effort of codifying the verification model, an automated generation of the PRISM probabilistic model was implemented based on an existing open source tool for TROPOS development support named TAOM4E[citation]. TAOM4E provides a graphical environment for goal modelling with TROPOS methodology based on the well known Eclipse Modelling Framework (EMF) and Graphical Editing Framework (GEF). The GORE to PRISM generator was implemented as a Eclipse plugin and integrated to the TAOM4E environment.

The purpose of the automated code generation for the probabilistic PRISM model is to optimize the formal verification step by abstracting the PRISM

language from the analysts and reduce the overhead and time of the model verification. This should increase the feasibility of adopting the extended TROPOS methodology by keeping analysts with their original responsibility of modelling and analysing the system, its social environment and its different contexts of operation.