A Systematic Comparison of *i** Modelling Tools Based on Syntactic and Well-formedness Rules

Catarina Almeida, Miguel Goulão, and João Araújo

CITI, FCT, Universidade Nova de Lisboa, Portugal
`acg.almeida@campus.fct.unl.pt`
{`mgoul,joao.araujo`}`@fct.unl.pt`

Abstract. There are several tools currently available in the *i** community. These tools have different features and purposes. Choosing the most adequate tool for a specific modelling situation can be a challenge. To overcome this difficulty, we present a systematic comparison of the *i** tools listed in the *i** wiki page, according to their features, syntax coverage and semantic analysis support. Our comparison highlights the different strengths of those tools, to help identifying situations for which each tool might be particularly useful. We contribute with an aggregated vision of current *i** tool support to the body of knowledge of the *i** community. In addition, this comparison also helps identifying opportunities for further evolution of the surveyed tools.

Keywords: *i** modelling framework, systematic comparison, tool support

1 Introduction

The *i** framework is a modelling language that covers both agent-oriented and goal-oriented modelling. There are several variations of the *i** framework, such as *Yu'95*, *TROPOS*, *Secure Tropos*, *Iterative Tropos*, and *GRL*. There are also several tools currently available to create *i** models. Those tools have different features, purposes, and various levels of conformity with the *i** syntax (usually alligned with one of the above-mentioned *i** frameworks).

Choosing the best tool for a specific purpose can be a challenge, not only because one has to select the most suitable *i** framework for the task, but also because different tools targeted to the same *i** framework provide different kinds of support for the specification of an *i** model, not only on the syntactic, but also on the semantic level. This support includes different levels of correctness checking of the created models. The *i** wiki page [1] includes a comparison of the *i** tools to address this challenge, covering information such as the purpose of each tool, the *i** framework it supports, practical details concerning availability, base platform, maturity, etc., as well as details on the tool modelling suitability, usability, extensibility and interoperability.

In this paper we present a systematic comparison of syntactic and semantic features supported by the different *i** tools. We use the language description available from the *i** wiki to guide our comparison. This provides a language-oriented comparison of the *i** tools, which bridges an important gap in the scope of the comparison currently available in the *i** wiki.

This paper is organized as follows: in section 2, we summarize the objectives of this *i** tools comparison; in section 3 we provide a detailed comparison of the *i** tools, focusing on the syntactic coverage and semantics checking of the *i** models; finally, in section 4, we present conclusions and further work.

2 Objectives of the Research

Our goal is to analyze *i** modelling tools, for the purpose of their comparison, with respect to their syntax coverage, and semantic checking support from the point of view of a requirements engineer, in the context of a survey of the existing tool support available to the *i** community.

More specifically, we aim to answer the following two research questions:

- **RQ1:** Which of the syntactic constructs described in the *i** wiki are supported by the *i** tool?
- **RQ2:** To what extent does each *i** tool support semantic checking of the *i** models built using it?

3 Scientific Contributions

In order to answer our research questions, we tested available *i** tools. The criteria used for the inclusion of the tools in this analysis was their presence in the *i** wiki page [1] and the availability of a functional URL. Some of the tools refered in the *i** wiki are no longer available in the advertised URL, and were therefore excluded from this analysis. Table 1 shows the different tools surveyed in this paper.

Table 1. Analysed Tools

***i** Tool**	**Institution**	***i** Variant**	**Platform**			**Technology**
			Windows	**Linux**	**MacOS**	
OpenOME	Univ. Toronto	Yu'95	Yes	Yes	Yes	Java (JRE)
TAOM4E	Univ. Trento	Tropos	Yes	Yes	Yes	Eclipse plug-in
GR-Tool	Univ. Trento	Tropos	Yes	Yes	Yes	Java (JRE)
STS-Tool	Univ. Trento	Tropos	Yes	Yes	Yes	Java (JRE)
jUCMNav	Univ. Ottawa	GRL	Yes	Yes	Yes	Eclipse plug-in
DesCARTES	U.C. Louvain	Yu'95 / Tropos	Yes	Yes	Yes	Eclipse plug-in

OpenOME [2] is an Eclipse-based tool designed to support goal-oriented, agent-oriented and aspects-oriented modelling and analysis. It is an open source tool and researchers can propose further branches and extensions to the tool. TAOM4E [3] is an Eclipse plug-in that supports a model-driven, agent oriented software development. GR-Tool [4] is a graphical tool for forward and backward goal reasoning in Tropos. STS-Tool [5] is a socio-technical security modelling tool to draw Tropos and Secure Tropos models and to perform the effective formal analysis of functional and security requirements. jUCMNav [6] is an Eclipse plug-in for modelling, analysis and transformation in both GRL and UCM (Use Case Map) notation. It is intended for the elicitation, analysis, specification and validation of requirements. DesCARTES [7] is an Eclipse plug-in that supports various models, such as *i**, NFR, UML and AUML models in the context of Tropos and I-Tropos development. The tool allows the development of the methodology analysis and design models as well as forward engineering capabilities and an integrated software project management module.

All the tools were tested in Arch Linux x86_64 and Windows Vista 32 bits, both with Java Runtime Environment 1.6.0. For the tools that are an Eclipse plug-in, the version of the Eclipse was Indigo (3.7) for TAOM4E and Juno (4.2) for jUCMNav and DesCARTES, with Eclipse Modelling Tools.

The syntax coverage analysis aims to check if the tool has *a*) the basic *i** syntax; and *b*) the graphical notation of the *i** syntax (according to the *i** wiki page). Table 2 shows this analysis. The cells with "Yes" denote that the tool complies with both criteria presented earlier. The cells with the symbol "*" after "Yes" mean that the tool complies with *a)* but not with *b)*; i.e., it has a different notation for the given element. The cells with "No" show that the tool does not comply with any of the criteria.

The GR-Tool is the only not supporting any type of actor, as it is mostly targeted to goal reasoning. OpenOME is the only tool that supports all types of actors with the *i** graphical notation. Actors links are often not provided and are only fully supported in OpenOME and jUCMNav. The STS-Tool also supports a type of actor association, namely the association "plays".

Concerning elements, all the tools support goals. Only jUCMNav has all kinds of elements of the *i** graphical syntax. DesCARTES also supports all element types, but two of them use a non-standard notation.

With respect to the support for dependency links, the tools which support them only have commited elements, but not open or critical elements. The GR-Tool and STS-Tool do not support any kind of dependency links.

All the tools have at least two types of contribution links and all of them have the "and" link. It is in the contribution links that the variation of the notation according to the graphical notation of *i** syntax is higher. OpenOME and jUCMNav are the only tools that have all types of contribution links.

As can be observed, OpenOME is the tool with the widest syntax coverage according to the two criteria described above and if it complies with the first one, it always complies with the second. jUCMNav complies with the first criteria except for dependency strengths but not always complies with the second one.

Table 2. *i** Syntax Coverage

	OpenOME	**TAOM4E**	**GR-Tool**	**STS-Tool**	**jUCMNav**	**DesCARTES**
Actors						
Actor	Yes	Yes	No	No	Yes	No
Actor Boundary	Yes	Yes	No	No	Yes	No
Role	Yes	No	No	Yes	Yes*	No
Agent	Yes	No	No	Yes	Yes*	Yes
Position	Yes	No	No	No	Yes*	No
Actors Links						
ISA	Yes	No	No	No	Yes*	No
Is-part-of	Yes	No	No	No	Yes*	No
Plays	Yes	No	No	Yes	Yes*	No
Covers	Yes	No	No	No	Yes*	No
Occupies	Yes	No	No	No	Yes*	No
INS	Yes	No	No	No	Yes*	No
Elements						
Goal	Yes	Yes	Yes	Yes	Yes	Yes
Softgoal	Yes	Yes	No	No	Yes	Yes*
Task	Yes	Yes*	No	No	Yes	Yes
Resource	Yes	Yes	No	No	Yes	Yes
Belief	No	No	No	No	Yes	Yes*
Links						
Dependency Links	Yes	Yes	No	No	Yes	Yes
Means-ends Links	Yes	Yes	No	No	Yes*	Yes
Decomposition Links	Yes	Yes	No	No	Yes	Yes
Contribution Links						
Make	Yes	No	No	No	Yes	Yes
Break	Yes	No	No	No	Yes	Yes
Unknown	Yes	No	No	No	Yes	Yes
Some+	Yes	No	Yes*	No	Yes	Yes*
Some-	Yes	No	Yes*	No	Yes	Yes*
Help	Yes	No	Yes*	Yes*	Yes	Yes*
Hurt	Yes	No	Yes*	Yes*	Yes	Yes*
And	Yes	Yes	Yes	Yes	Yes	Yes
Or	Yes	Yes	Yes	Yes	Yes*	No
Dependency Strengths						
Open Element	No	No	No	No	No	No
Commited Element	Yes	Yes	No	No	Yes	Yes
Critical Element	No	No	No	No	No	No

The semantic analysis aims to determine the level of corretness checking of the created models. Using the descriptions and guidelines available in the *i** wiki, we analysed if the tool verifies when a modelling error is made. Table 3 shows a set of errors and if the tool verifies those errors or not. The N/A means that the tool does not have one or more elements needed to do the evaluation.

Table 3. Semantic Analysis

	OpenOME	**TAOM4E**	**GR-Tool**	**STS-Tool**	**jUCMNav**	**DesCARTES**
Actors and relations						
Actors without links	Yes*	No	N/A	No	No	No
Actor inside another actor boundary	Yes	Yes	N/A	Yes	No	Yes
Dependencies						
Dependency link without a dependum	Yes*	Yes	N/A	N/A	Yes	Yes
Dependency links with different directions	No	Yes	N/A	N/A	No	Yes
Dependency link inside an actor boundary	Yes*	Yes	N/A	N/A	Yes	N/A
Other link rather than dependency link between an element and an actor	Yes	Yes	N/A	Yes	No	Yes
Associations						
ISA between actors of different types	No	N/A	N/A	N/A	Yes	N/A
Is-part-of between actors of different types	No	N/A	N/A	N/A	Yes	N/A
Other association rather than Plays between Agent and Role	No	N/A	N/A	Yes	Yes	N/A
Other association rather than Covers between Position and Role	No	N/A	N/A	N/A	Yes	N/A
Other association rather than Occupies between Agent and Position	No	N/A	N/A	N/A	Yes	N/A
INS between others than agents	No	N/A	N/A	N/A	Yes	N/A
Associations between elements that are not actors	Yes	N/A	N/A	Yes	Yes	N/A
Internal Elements						
SR elements outside actor boundary	No	Yes	N/A	Yes	No	N/A
Softgoal decomposition in sub-softgoals or sub-tasks	No	No	N/A	N/A	No	Yes
Goal decomposition in sub-goals or subtaks	Yes*	No	N/A	N/A	No	Yes
Goal decomposition without means-end	No	No	N/A	N/A	No	Yes
Means-end where a goal is the mean	No	No	N/A	N/A	Yes	No
Means-end different from "task–>goal"	No	No	N/A	N/A	No	No
Decomposition beyond the actor boundary	No	Yes	N/A	N/A	No	N/A
Means-end beyond the actor boundary	No	Yes	N/A	N/A	No	N/A
Means-end decomposition to refine a softgoal	No	No	N/A	N/A	No	Yes
Softgoal decomposition without contribution links	No	No	N/A	N/A	No	No
Any kind of direct relation between goals	Yes*	No	N/A	N/A	No	Yes
Link between an element inside the actor boundary and that actor	Yes	Yes	N/A	Yes	Yes	N/A
Contribution Links						
Contribution links between any element to any element rather than softgoal	No	No	N/A	N/A	Yes	No
Contribution link between actors	Yes	Yes	N/A	Yes	Yes	Yes
Contribution link between goals and subgoals or sub-tasks	No	No	No	Yes	Yes	No

Some of the OpenOME categories are labeled with "Yes*". This means that the tool itself does not perform the corresponding analysis by default, but can do it with the add-on "Syntax check". However, syntax checking does not work for Linux and may not work for Mac as the executable version of prolog is not available for these platforms. In jUCMNav, it is necessary that the static semantics checking properties are selected.

*i** tools support error prevention in two different ways. One is by not allowing the user to do what he intends, if it is not correct. The other one is by allowing the user to do so, but inform him that the resulting model is incorrect. In general, the level of correctness checking of the created models is not too deep. Note that the set of modelling errors that can be made depends on the available modelling elements, which vary from one tool to the next. On average, about 39% of the considered modelling errors are not applicable, due to the lack of support of the corresponding modelling elements by the tools.

There are some errors that all the tools (except those that do not support the used model elements) verify. This is the case of a dependency link without a dependum, associations between elements that are not actors, a link between an element inside the actor boundary and that actor, and a contribution link between actors. The opposite can also be noticed, i.e., there are some errors that none of the tools verify. This is the case of means-ends relations different from "task–>goal", softgoals decomposition without contribution links and contribution links between any element to any element rather than a softgoal.

jUCMNav is the tool with the highest number of verified errors, with a verification percentage of 50%, followed by OpenOME and TAOM4E. They are also the tools with the lowest number of errors that were not possible to analyse.

4 Conclusions and Future Work

This paper provides a systematic comparison of some *i** tools, listed in the *i** wiki. The goal here is to complement existing comparison work by providing a comparison on syntatic and semantic properties of the *i** modelling tools.

The contribution of this work is to provide relevant information to practitioners when selecting an *i** tool, to tool developers when improving existing tools, and to researchers when studying the shortcomings of existing tools.

We plan to extend this analysis to other tools and include non-functional aspects in the evaluation.

References

1. *i** wiki: `http://istarwiki.org/`
2. OpenOME: `https://se.cs.toronto.edu/trac/ome/`
3. TAOM4E: `http://selab.fbk.eu/taom/`
4. GR-Tool: `http://troposproject.org/tools/grtool/`
5. STS-Tool: `http://www.sts-tool.eu/`
6. jUCMNav: `http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG/`
7. DesCARTES: `http://www.isys.ucl.ac.be/descartes/`