Analysing the Style of Textual Labels in i^* Models

Arian Storch[1], Ralf Laue[2], and Volker Gruhn[3]

[1] it factum GmbH
arian.storch@it-factum.de
[2] University of Applied Sciences of Zwickau, Department of Information Science
ralf.laue@fh-zwickau.de
[3] Paluno - The Ruhr Institute for Software Technology, University of Duisburg-Essen
volker.gruhn@paluno.uni-due.de

Abstract. An important quality aspect for conceptual models (such as i^* models) is the quality of textual labels. Naming conventions are aimed to make sure that labels are used in a consistent manner. We present a tool that checks automatically whether a textual label in an i^* model adheres to a set of naming conventions. This does not only help to enforce the use of a consistent labelling style, it also helps to detect modelling errors such as goals in i^* models that should be softgoals (or vice versa).

1 Introduction

i^* is a frequently used visual language for modelling the social relationships between actors. The language contains graphical symbols for various concepts (such as goal or task) which have to be used correctly if the model should be useful. In [1], Horkoff et al. analysed 30 i^* models from student works and academic papers in order to discover model elements that use a convention contrary to the generally accepted guidelines published in the i^* Wiki [1]. They found a large number of problems in these models that result from the fact that the wrong type of model element was used. In the 30 models, Horkoff et al. identified 10 problems of the type "softgoal should be goal", 15 problems "goal should be softgoal", 8 problems "task should be softgoal" and 7 problems "softgoal should be task".

We believe that a way to reduce such problems can be to use a consistent labelling style throughout the model. By forcing the modeler to think whether an element should be named "reduce waste" or "waste to be reduced", he or she is also forced to think whether the concept should be represented as a task or as a goal. Furthermore, the usage of common labelling styles can reduce the difficulty to understand a model.

[1] http://istar.rwth-aachen.de/tiki-index.php?page=i*+Guide

2 Labelling Styles for i^* Models

We are aware of three papers that suggest conventions for the style of i^* element labels. In [2], the authors recommend the style

Element type	Syntax	Example
Goal	object + "be" + verb in passive voice	Information is published
Softgoal	quality attribute (+ object or task)	Secure access
Task	verb in infinitve + object	Confirm agreement
Resource	name of the object	User access

[3] suggests a similar style:

Element type	Syntax	Example
Goal	subject + "be" + verb	Result be correct
Softgoal	softgoal [topic]	Improve [IT skills]
Task	verb + object	Fill out application form
Resource	Noun	Confirmation

[4] suggests the labelling style:

Element type	Syntax	Example
Goal	object + passive verb	Information collected
Softgoal	goal syntax + complement (object) complement ([Dependum])	Information checked quickly Timely [Virus List]
Task	verb (+ object) (+ complement)	Answer doubts by e-mail
Resource	(adjective +) object	Updated Virus List

In the above tables, parentheses are used to denote optional elements; brackets are part of the label. To our best knowledge, current i^* modelling tools are not able to validate labels with respect to such conventions. However, there is a great amount of work on checking the labels in business process models. When developing our approach for style checks of i^* labels, we made use of the experiences with tools developed in this context (see [5–7]; the last reference contains an overview on more papers on the topic).

Although the styles shown in the above tables seem to be quite similar, a closer look shows that there are subtle differences. Let's assume that a Resource label should be a *"name of the object"*. This is more restrictive than saying it should be a noun, because nouns can be accompanied by articles, attributive and adjectives. In that case *"mail"* would be valid, but *"sent mail"* won't. Similarly, if we would require that a Softgoal should rather be labeled with a quality attribute than with a Goal syntax, *"usable"* would be valid, but not *"User interface is usable"*. For this reason, we decided to regard a label as having the correct style if it adheres to the following superset of style rules:

Element type	Syntax	Example
Goal	object + passive verb	Trip advice is provided
Softgoal	quality attribute (+ object) (+ complement)	Precise information
	goal syntax (+ complement)	Document is sent securely
Task	verb (in present form) + object (+ complement)	Use back-end user interface
Resource	object	Route card

3 Label-Checking Algorithm and Patterns

3.1 Third-Party Frameworks for Natural Language Processing

To analyse, process and validate a label, we first need to know what kinds of words it contains. This process is called part-of-speech (POS) tagging. POS-taggers typically combine lexical databases with statistical algorithms to determine the kind of a word, a part-of-speech or even a phrase within a sentence[8]. One of the most popular POS tagger is the Stanford Parser[2] which is contained in a toolset developed by the Stanford Natural Language Processing Group[3]. Another frequently used tool is WordNet[9][4], a lexical database which provides information about semantic and lexical relations between words. A combined API for both frameworks is provided by a tool called *qap* (quality assurance project) which is currently developed by the bflow* Toolbox team[5]. The API provides an easy access to the Stanford Parser and WordNet. Using that API, we could focus on the implementation of the labelling style checks which will be discussed in the following subsections.

3.2 Our Approach

Our goal is to compare the label of an i^* model element to the style rules for this element type. For this purpose, we make use of the Stanford Parser, a statistical parser that works out the grammatical structure of sentences. It can recognize which groups of words go together as phrases and which words are the subject or object of a verb[2].

The parser provides viable results when the input is a complete sentence. However, this is unfortunately often not the case for a label in an i^* model. Assume that a Resource is labeled with *"log message"*. This phrase is ambiguous, because *"log"* can be either a verb or a subject. To determine the correctness, the parser needs more context information, which we can derive from the element type. By analysing "log message" only, the parser will find that "log" is a verb and "message" is a noun. In that case, it cannot recognize that this phrase is

[2]http://nlp.stanford.edu/software/lex-parser.shtml
[3]http://www-nlp.stanford.edu/
[4]http://wordnet.princeton.edu/
[5]http://www.bflow.org/

an object which is valid for a Resource label. Similar problems exist with other element types. To deal with such problems, we have decided to add additional words to the labels, thus trying to create complete sentences.

Our general approach can be summarized as follows: First, we complement the label with a prefix that depends on the element type such that for correctly named labels, we get a complete sentence. For the label to be valid, the resulting sentence has to be syntactically correct. In the next step, the Stanford Parser processes the sentence and creates the so-called phrase structure tree of the sentence. It assigns to each word a part-of-speech (POS) tag such as CC (coordinating conjunction), DT (determiner), EX (existential *there*), IN (preposition), JJ (adjective), NN (noun), VB (verb), VBN (Verb, past participle) and VBZ (verb, 3rd person singular present)[10].

We define a pattern of valid sequences of POS tags for each type of modelling element. The label is regarded as valid if its POS tags match this pattern, allowing that the label may contain additional words *after* the pattern (this way both "Test database" as "Test database for consistency" would be regarded as a task).

We describe the style rules by a pattern in the Extended Backus-Naur Form. At first, we describe the POS tags of an **Object**. Examples of valid objects and the corresponding sequences of POS tags are:

Label	Sequence of POS tags
bank	NN
test run	NN NN
list of credit cards	NN IN NN NN
list of valid credit cards	NN IN JJ NN NN
summarized balance sheet	JJ NN NN

This is expressed by the following rules:
NNSEQ := NN, {NN}; *(examples: "bank", "test run")*
JNS := [JJ, {JJ}], NNSEQ; *(examples: "cheque", "valid cheque")*
OBJECT := [DT], JNS, [IN, JNS]; *(example: "a complete list of valid credit cards")*

Resource style check Given a label L, we complement it with the prefix "There is a" and the suffix "." (period), i.e. the parser analyses the (potential) sentence "There is a L." . We conclude that the label is correct, if L matches the pattern of an object.

We use *"There is"* as prefix in order to increase the probability of identifying *"is"* as the only verb of the sentence. For instance, if we have to validate the label *"summarized balance sheet"*, *"summarized"* (without this prefix) would be wrongly tagged as verb.

Task style check Given a label L, the parser analyses the (potential) sentence "I L."

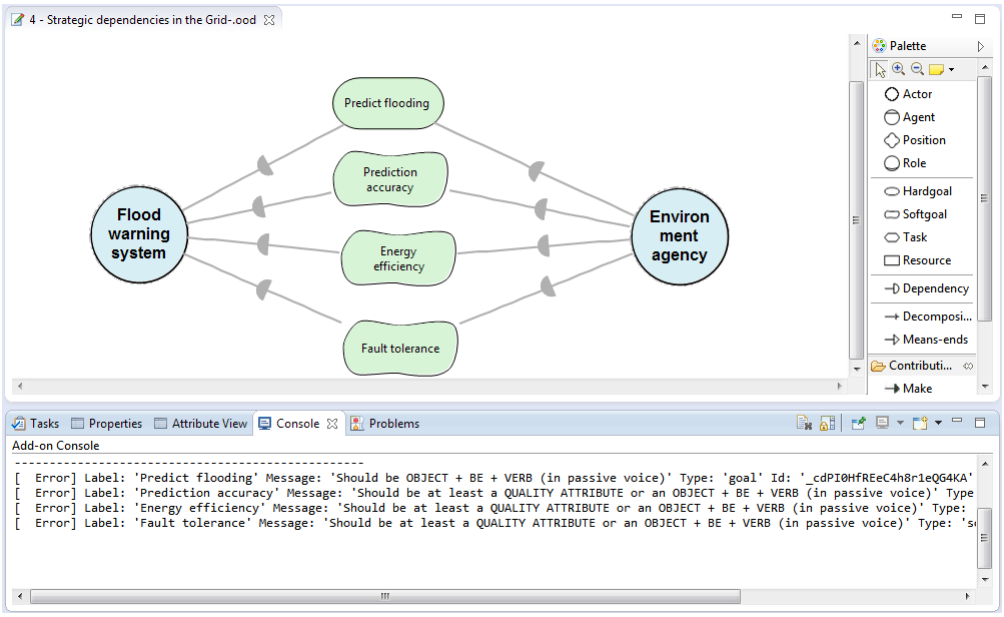

Fig. 1. Result of a Style Check in *openOME*

L is required to match the pattern of a Task:
TASK := VB, [IN], OBJECT; *(examples: "Test database", "Add to score")*

Goal style check A Goal label is required to match the pattern:

GOAL := OBJECT, ("is" | "are"), VBN; *(example: "Results are corrected")*

Softgoal style check Because there are two competing rules, the validation is done in two steps. First, the label is validated by the "quality attribute" rule. For this purpose, the parser analyses the (potential) sentence *"It is L."*. Second, it is checked whether the label is a goal, followed by an arbitrary complement.

L is required to match the pattern of a Softgoal (possibly followed by an arbitrary complement):
SG := QA | GOAL; (quality attribute or goal)
QA := JJ, {JJ}, [OBJECT]; *(example: "inexpensive delivery")*

Fig.1 shows the validation result of a model from our tool within the i^* modelling tool *openOME*[6].

[6] http://www.cs.toronto.edu/km/openome/

4 Conclusion

In this paper, we presented a set of POS tag patterns that can be used to validate the adherence of i^* labels to a set of style recommendations. We derive the correctness of a label from its element type and the related style recommendation. We achieved reliable results by complementing the given (normally short) labels to (potentially) whole sentences.

These patterns have been implemented using the tool *qap* that provides an API to WordNet and the Stanford Parser. Our tool prototype is both extensible as configurable. It is easy to change our patterns, remove or add new ones. Though we tested our patterns within *openOME*, there is no technical dependency between *openOME* and our prototype. *qap* and our style checks can be used with any other tool as well.

A drawback we observed quite often is an incorrect spelling which flaws the reliability of the checks. For instance, "relevant advices" will lead to another result than "relevant advises" because the POS tagger cannot identify the word correctly.

In future, we plan to add additional linguistic analysis functionality to our tool in order to make more sophisticated analysis possible.

References

1. Horkoff, J., Elahi, G., Abdulhadi, S., Yu, E.: Reflective analysis of the syntax and semantics of the i* framework. In: Advances in Conceptual Modeling, Challenges and Opportunities. Volume 5232 of LNCS. Springer (2008) 249–260
2. de Pádua Albuquerque Oliveira, A., do Prado Leite, J.C.S., Cysneiros, L.M.: Using i* meta modeling for verifying i* models. In: 4th International i* Workshop4. (2010) 76–80
3. de Pádua Albuquerque Oliveira, A., Cysneiros, L.M.: Defining strategic dependency situations in requirements elicitation. In: Workshop em Engenharia de Requisitos. (2006) 12–23
4. Martínez, C.P.A.: Systematic Construction Of Goal-Oriented COTS Taxonomies. PhD thesis, Universitat Politecnica de Catalunya (2008)
5. Becker, J., Delfmann, P., Herwig, S., Lis, L., Stein, A.: Formalizing linguistic conventions for conceptual models. (2009) 70–83
6. Leopold, H., Smirnov, S., Mendling, J.: On the refactoring of activity labels in business process models. Inf. Syst. **37** (2012) 443–459
7. Leopold, H., Eid-Sabbagh, R.H., Mendling, J., Azevedo, L.G., Baião, F.A.: Detection of naming convention violations in process models for different languages. Decision Support Systems **56** (2013) 310–325
8. Megyesi, B.: Shallow parsing with pos taggers and linguistic features. The Journal of Machine Learning Research **2** (2002) 639–668
9. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press (1998)
10. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: The Penn Treebank. Comp. linguistics **19** (1993) 313–330