REDEPEND: Extending *i** Modelling into Requirements Processes

James Lockerbie and Neil Maiden
Centre for HCI Design, City University, London, UK
ac769@soi.city.ac.uk, n.a.m.maiden@city.ac.uk

Abstract

This paper describes our REDEPEND tool for i goal modelling and analysis, and new features designed to make it more usable and useful. Pattern-based techniques that generate text requirements statements from graphical i* models improve the productivity and utility of REDEPEND in requirements projects.*

1. *i** and REDEPEND

*i** has been available in research communities for more than 10 years, but it has not been applied widely in industrial requirements projects. This is despite undoubted strengths, which include a simple but formal and stable semantics, a graphical modelling notation that is simple to use, models that are amenable to computational analysis, and applicability in both agent-oriented and goal-oriented requirements methods. We believe that one reason for this lack of industrial uptake is the absence of robust, useful and usable tools for developing and analysing *i** models. The original *i** authors [4] developed the Java-based OME tool, however elsewhere tool-based support for *i** requirements modelling is limited to Alexander's simple extension of ScenarioPlus. One objective of our research is to deliver a usable and useful *i** modelling tool within commonly used graphical modelling tools – in this case MS Visio. This tool is called REDEPEND.

REDEPEND is designed to provide systems engineers with *i** modelling and analysis functions, coupled with additional functionality and reliability of Microsoft Visio. It provides a graphical palette from which systems engineers can drag-and-drop *i** concepts to develop Strategic Dependency (SD) and Rationale (SR) models.

REDEPEND also provides systems engineers with simple model checking functions for SD and SR models. Earlier versions implement modelling constraints that, if activated, forbid a user to add or change a model element that violates *i** model constraints. Examples that are forbidden include task decomposition links that decompose elements that are not a task, and contributes-to soft goal links that do not have a soft goal element as an end.

2. Extending REDEPEND

REDEPEND has been used successfully to model complex socio-technical systems in European air traffic control projects [2,3]. However, although requirements analysts were able to develop and use the models within each project, some usability problems arose when developing and reviewing large *i** models, and questions were asked about the wider utility of *i** modelling given the efforts needed to develop the models in the first place. Hence, over the last 12 months we have extended REDEPEND with new features that are designed to make it more useful and usable to requirements engineers:

- New modelling and analysis features based on user feedback in real-world projects, to ensure that REDEPEND is usable when modelling and analysing large system models of 100s of elements;
- New components to generate from *i** models text requirements statements needed by organizations to deliver requirements that are legally binding to justify the effort needed to produce *i** models and make requirements projects more productive.

We believe that these new features are essential to the successful uptake of *i** in industrial projects. Let us describe these features in more detail.

2.1. Improved Usability Features

REDEPEND has been extended with new features to make it more usable when handling larger *i** models:

- One-click navigation between elements on related *i** models, for example between an actor element on a SD model and that actor's corresponding SR model, to facilitate rapid access to different parts of a modelled system according to viewpoints;
- Change synchronization features that propagate a model change in all related models, for example changing an actor name in all models specifying that actor, to ensure model consistency;
- Check features to highlight and shade-out model elements using layers, to partition and mark up models during analysis and review tasks;

- Model colour-coding, to highlight model features during walkthroughs.

Some of these features are shown in Figure 1.

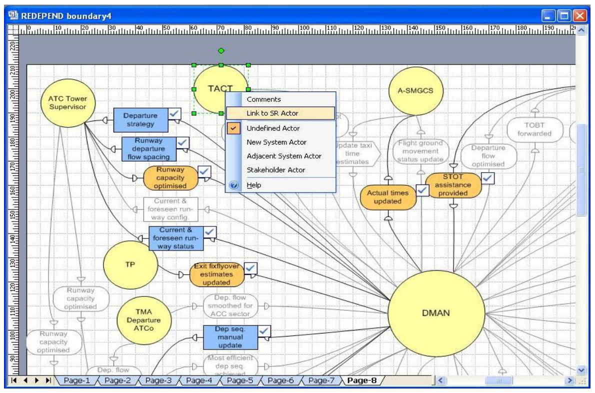
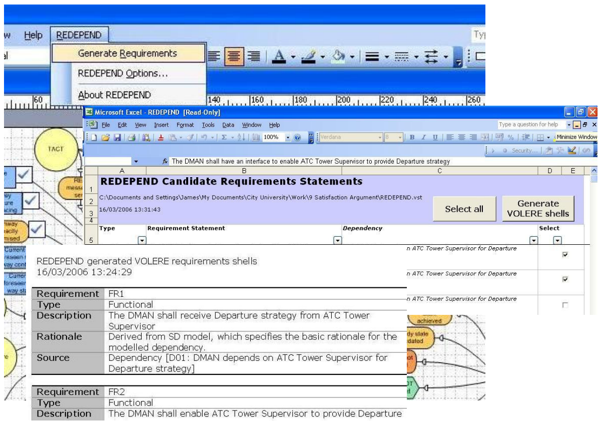


Figure 1. REDEPEND's new usability features designed to support scaleable *i system modelling**

2.2. New Productivity Features

To make REDEPEND more useful to requirements analysts we designed simple patterns – recurring syntactic and semantic structures in the *i** models – that are applied automatically to any SD model expressed in REDEPEND to generate textual requirement statements. Our patterns are not traditional in the design sense – a solution to a problem in context. Rather each pattern defines one or more desired properties (requirements) on the future system that must be satisfied for the SD model dependency to hold for the future system. As such, the SD model, which has been signed off as complete and correct, informs further discovery and specification of requirements statements. The concepts and patterns underlying this approach are described at length in [3]. The 19 patterns are represented in an MS Excel file, so that new patterns can be added without requiring any changes to REDEPEND software. Full details of REDEPEND v4.1 are in [1].

Figure 2 demonstrates how REDEPEND generates requirements from an analyst's perspective. The top image shows how an analyst accesses the requirements generation function, from the REDEPEND top-line pull-down menu. The middle image shows how REDEPEND delivers the candidate requirements into tailored MS Excel sheets. The analyst can tick and un-tick selected requirement statements prior to generating structured VOLERE shells in MS Word, as depicted in the bottom image. The selected requirements are automatically generated into an MS Word document, as this is the most common storage mechanism for requirements, even when using requirements management tools such as DOORS and Requisite Pro. Each requirement in the document is structured using and expressed with a partially complete VOLERE shell. For each requirement, the shell specifies a unique identifier; the requirement type; the requirement description; a rationale of canned text describing how the requirement was generated; and the source dependency in the SD model.



The DMAN shall have an interface to enable ATC Tower Supervisor to provide Departure strategy

REDEPEND Candidate Requirements Statements

C:\Documents and Settings\James\My Documents\City University\Work\9 Satisfaction Argument\REDEPEND.vst

16/03/2006 13:31:43

Type	Requirement Statement	Dependency	Select
		n ATC Tower Supervisor for Departure	✓
		n ATC Tower Supervisor for Departure	✓
		n ATC Tower Supervisor for Departure	☐

REDEPEND generated VOLERE requirements shells
16/03/2006 13:24:29

Requirement	FR1
Type	Functional
Description	The DMAN shall receive Departure strategy from ATC Tower Supervisor
Rationale	Derived from SD model, which specifies the basic rationale for the modelled dependency.
Source	Dependency [D01: DMAN depends on ATC Tower Supervisor for Departure strategy]

Requirement	FR2
Type	Functional
Description	The DMAN shall enable ATC Tower Supervisor to provide Departure

Figure 2. The three stages of requirements generation in REDEPEND

3. Ongoing Work

We are currently applying this new version of REDEPEND in two projects, to specify requirements for new systems to reduce the environment impact of regional airports funded by the UK DTI, and to analyse safety-related requirements for a new infringement detection system with the UK's National Air Traffic Services. We aim to report feedback from both projects at the RE'06 conference.

4. References

[1] Lockerbie J., 2005, 'Automating the Pattern-Based Generation of Requirements from *i** System Models', MSc Dissertation, School of Informatics, City University, November 2005.

[2] Maiden N.A.M., Jones S.V., Manning S., Greenwood J. & Renou L., 2004, 'Model-Driven Requirements Engineering: Synchronising Models in an Air Traffic Management Case Study', to appear in Proceedings CaiSE'2004, Springer-Verlag.

[3] Maiden N.A.M., Manning S., Jones S. & Greenwood J., 2006, 'Generating Requirements from Systems Models using Patterns: A Case Study', Requirements Engineering Journal 10(4), 276-288.

[4] Yu E. & Mylopoulos J.M., 1994, 'Understanding "Why" in Software Process Modelling, Analysis and Design', Proceedings, 16th International Conference on Software Engineering, IEEE Computer Society Press, 159-168.