Dependability in RESCUE: A Concurrent Engineering Approach to the Specification of Requirements for Air Traffic Management

Neil Maiden, Sara Jones
Centre for HCI Design, City University, Northampton Square,
London EC1V 0HB, UK
Email: n.a.m.maiden@city.ac.uk, s.v.jones@city.ac.uk
Tel: +44 20 7040 8412, Fax: +44 20 7040 8859

Abstract

RESCUE is a concurrent engineering approach to the specification of operational requirements in the domain of air traffic management. In RESCUE, we build on use cases and scenarios, using work from human-computer interaction (HCI) and elsewhere in requirements engineering. Our aim is to provide some validation of the use case models through which requirements are elicited, and thereby obtain a more complete specification of requirements. We focus in particular on situations where things go wrong, in order to increase our understanding of issues relating to system dependability, and enable us to specify requirements which define how the system should behave under those circumstances. In this paper, we consider the way in which RESCUE uses work from requirements engineering, HCI and cognitive psychology to address some of the dependability problems which arise in the domain of air traffic management, and describe a proposed extension to the process which will further strengthen our approach to specifying requirements for dependable systems.

1. Introduction

The RESCUE (Requirements Engineering with SCenarios in a User-Centred Environment) process was initially developed for use in specifying requirements for CORA-2 ('CORA' stands for Conflict Resolution Assistant). CORA-2 is a system that will provide computerised assistance to air traffic controllers to resolve potential conflicts between aircraft. It has since been applied in specifying requirements for two further systems: DMAN (Departure Manager), which will support controllers in scheduling and managing the departure of aircraft from major European airports; and MSP (Multi-Sector Planning), which will be used in scheduling aircraft from gate to gate across multiple, multi-national sectors. These systems are complex socio-technical systems, situated within complex environments. Because of this, the RESCUE process has been designed, by academic researchers working together with staff at Eurocontrol (the European Organisation for the Safety of Air Traffic Management) to integrate current research and best practice in the domains of HCI and requirements engineering.

In RESCUE, we build on use cases and scenarios, with the aim of increasing the coverage of use cases and providing some validation of use case models through the use of other models which can be checked against them. Use cases are an extremely popular approach to requirements elicitation. According to Neill and Laplante [9], they are currently used for requirements elicitation in over 50% of projects. In RESCUE, we use work from HCI and elsewhere in requirements engineering to augment current best practice in this area in order to improve the use case models through which requirements are elicited, and thereby obtain a more complete specification of requirements. We focus in particular on situations where things go wrong – for example where there is an equipment malfunction, or where a pilot behaves in an unexpected manner – in order to increase our understanding of issues relating to system dependability, and enable us to specify requirements which define how the system should behave under those circumstances.

In this paper, we present an overview of the RESCUE process, then consider the way in which RESCUE uses work from requirements engineering, HCI and cognitive psychology to address some of the dependability problems which arise in the domain of air traffic management. We also describe a proposed extension to the process, involving the construction of satisfaction arguments, which will further strengthen our approach to specifying requirements for dependable systems.

2. RESCUE Process Overview

The RESCUE process consists of a number of sub-processes, organised into 4 ongoing streams. These streams run in parallel throughout the requirements specification stage of a project, and are mutually supportive. An overview of the process is provided in figure 1.

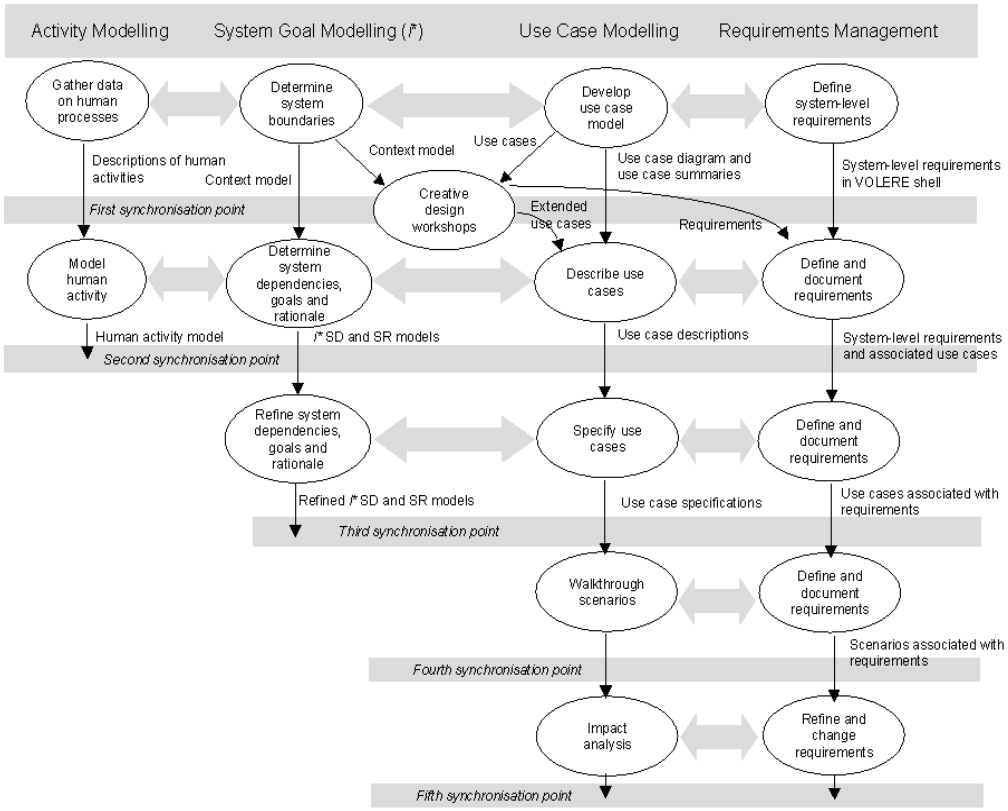

Figure 1: Overview of the RESCUE process

The four RESCUE streams focus on the areas of:

- Analysis of the current work domain, including actors, actions and environment, using human activity modelling (based on work described in [1], [14] and [17]);
- System goal modelling using the *i** goal modelling approach to model strategic dependencies between actors [18];
- Use case modelling and specification, followed by systematic scenario walkthroughs using the CREWS-SAVRE and CREWS-ECRITOIRE approaches to identify requirements for the future system [16];
- Requirements management using VOLERE [13], implemented in Rational's requirements management tool RequisitePro in current rollouts of RESCUE.

In addition to these four streams, the RESCUE process uses the ACRE framework to select techniques for requirements acquisition [6], and creativity workshops**,** based on models of creative and innovative design [5], to discover candidate designs for the future system, and to analyse these designs for fit with the future system's requirements.

Consistency between the various artefacts and deliverables produced at different stages in the RESCUE process is checked at five different 'synchronisation points' during the process, as shown in figure 1.

3. RESCUE's Approach to Dependability

In RESCUE, we aim to specify requirements in relation to availability, reliability, recoverability, safety, security and maintainance, as well as functional requirements and non-functional requirements in other areas such as usability, training, interoperability and performance. Requirements of all these types are identified in the same manner, using the process as a whole. The basic approach to dependability currently employed in RESCUE may be characterised in terms of:

- Understanding the nature of the current system, and those features of the system, such as the environment in which it operates, which will persist once a new system is introduced;
- Understanding what can go wrong in the current system, and with technical and socio-technical systems in the domain in general;
- Extrapolating from this to think about what could go wrong in the future socio-technical system, and thereby specify appropriate dependability requirements.

In addition to this, we also carry out a number of checks between the different models developed in different process streams, with the aim of validating the use case descriptions from which requirements are generated, as described in [7].

In the rest of this section, we will describe each of the above steps in more detail, using examples from the DMAN project. We will also present our proposals for extending the RESCUE approach by incorporating satisfaction arguments, which demonstrate that requirements, and in particular dependability requirements, can be satisfied in the future system.

3.1. Understanding the Current System

An understanding of the current system is developed in the activity modeling stream by building models of human activity which include concepts derived from task analysis [1], cognitive task analysis [14], and work domain analysis [17] as described in [4]. Briefly, our human activity model is composed of a number of human activity descriptions, analogous to use case descriptions for the future system. These human activity descriptions are written using pre-defined templates, which include place-holders for the following types of information:

- Goals - states of the system which one or more actors wish to bring about, for example, a controller may have a goal of communicating a flight plan to a pilot;
- Human actors - people involved in system, for example pilots, air traffic controllers;
- Actions – physical, cognitive and communication actions undertaken by actors to solve problems or achieve goals, for example touch strip (physical), read strip information (cognitive), or talk to pilot (communication);
- Resources – means that are available to actors to achieve their goals, for example flight strips and information about a flight;
- Resource management strategies – how actors achieve their goals with the resources available, for example writing down flight information on the flight strips;
- Contextual features – situational factors that influence decision-making, for example priorities are given to incoming aircraft; and
- Constraints - environmental properties that affect decisions, for example the size of the flight strip bay, which limits the number of strips to work with.

The human activity model for DMAN consisted of 15 separate human activity descriptions, each involving between 1 and 7 actors, and where the normal course (the sequence of actions followed under normal circumstances) contained between 5 and 12 actions.

3.2. Identifying Abnormal States and Behaviours in the ATM Domain

Abnormal states and behaviours relevant to a range of socio-technical systems were identified as part of the CREWS-SAVRE project [16] through reference to basic research in cognitive science on human error (see, for example, [12], [10], and [11]), interaction failures in human-computer interaction [15], and failures in human-human and machine-machine communication. The result was 54 classes of abnormal behaviour and state, which were stored in a database and used to identify alternative courses as described below. As part of the CORA-2 project, this view of abnormal behaviour was specialised for the domain of air traffic management using a combination of scenario and laddering techniques from knowledge elicitation with small groups of air traffic controllers [8]. 138 new classes of exceptions that specialized the original 54 classes of abnormal behaviour were elicited in this way. Further elicitation was carried out for the DMAN project, which resulted in the identification of 192 classes of abnormal behaviour and state specific to departure management. A simple example of the class of controller-controller communication exceptions is shown in figure 2.

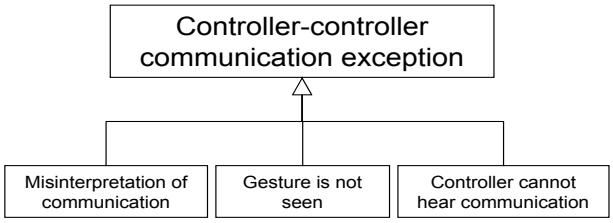

Figure 2: Specialisation hierarchy of controller-controller communication exception classes

3.3. Identifying Dependability Requirements

Requirements, including dependability requirements, are identified during scenario walkthroughs, in which stakeholders are guided to consider requirements relating not just to the normal course of events, but to numerous

alternative courses identified automatically by our scenario walkthrough tool using the categories of error and exceptions identified above. This process is described in more detail in [8]. As an example, consider an action in one of the DMAN scenarios: *The Departure Clearance ATCO finds the flight information on the DMAN display*. An alternative course generated by the scenario walkthrough tool in relation to this action was: *What if the ATCO interface fails during this action?* This prompted one of the stakeholders to identify an availability requirement: *AR1: DMAN shall be available equal to, or more than the FDPS [Flight Data Processing System] and EFPS [Electronic Flight Processing System].*

3.4. Developing Satisfaction Arguments

In addition to the approach described above we intend, in our next project, to extend the RESCUE process to include construction of satisfaction arguments about dependability and dependability-related requirements. Satisfaction arguments are applied within the REVEAL requirements engineering method [2], and recognize the role of domain knowledge, a major focus for RESCUE, in the requirements process. A satisfaction argument relates domain knowledge to the introduction of a system (Machine in Jackson's terms) into the environment (or World in Jackson's terms – [3]). A satisfaction argument is used to show that, given relevant properties of the domain, combined with the specification of the behaviour of the machine or system to be developed, the requirements on the system will hold [2]. In RESCUE the specification of the system to be developed is expressed using use cases that are associated with requirements relating to use case actions. Domain properties are expressed primarily in human activity descriptions, and requirements in Jackson's sense of the term are expressed using *i** system models [18] and the VOLERE shell [13] as requirements on the system as a whole or actors in that system. These requirements are often dependability requirements.

Let us demonstrate the development and use of a possible satisfaction argument using data, descriptions and models from the DMAN project. The DMAN *i** SD and SR models [18] of DMAN actors, goals and dependencies identified an important dependability-related soft goal for Runway ATCOs – that their overall workload shall not be increased. The SR model also identified one task that the Runway ATCO must undertake – respect the CTOT (the required time of take-off for an aircraft) – and a goal that the ATCO must attain – the departure sequence is followed. However, both contribute negatively to the achievement of this soft goal. In simple terms, respecting the CTOT and following the departure sequence increases the ATCO's workload. One of DMAN's specified capabilities is to provide advice to the Runway ATCO on how to respect the CTOT – this capability contributes positively to the satisfaction of the soft goal, as it can reduce the ATCO's workload. In the current version of RESCUE, these *i** models are insufficient to relate other data, models and specifications in RESCUE. Hence we propose to construct a simple satisfaction argument.

The structure of the satisfaction argument is shown in figure 3. The argument relates elements of the use case specification, human activity description and generated ART-SCENE scenario:

- Use case UC6 – DMAN updates the departure sequence – specifies DMAN behaviour for validating the current departure sequence, calculating a new sequence with revised pushback times and MTOT (DMAN-calculated airborne times), and making the revised sequence available to ATCOs including the Runway ATCO so that it can be understand and implemented. These behaviours are included in the satisfaction argument;
- ART-SCENE DMAN scenario 24 takes UC6 as its input and generates normal course events such as *DMAN displays to the Runway ATCO any revised MTOTs for aircraft under his/her control*, and alternative courses including *What if the aircraft is not added to the departure sequence*, and *What if an incorrect SID is used?* These alternatives represent possible hazards and events that can increase the Runway ATCOs workload. They are also included in the satisfaction argument;
- Human activity descriptions developed from data gathered in the Heathrow control tower describe how ATCOs currently optimize the departure sequence for departing aircraft using physical flight strips and timings from a clock. The model describes the physical and cognitive actions needed to construct and revise the departure sequence manually, and the different types of knowledge that were used. Examples include cognitive actions such as *understanding aircraft classifications* and knowledge of factors such as *the required spacing between aircraft*, and *workload needed to optimize the departure sequence*. They are also included in the satisfaction argument.

Using these data and models we were are able to construct possible arguments such as those shown in figure 3. The evidence suggests that, from the specified system, possible abnormal behaviours and tacit knowledge needed to classify aircraft, the specified DMAN system might be unlikely to satisfy the Runway ATCO requirement not to increase their workload.

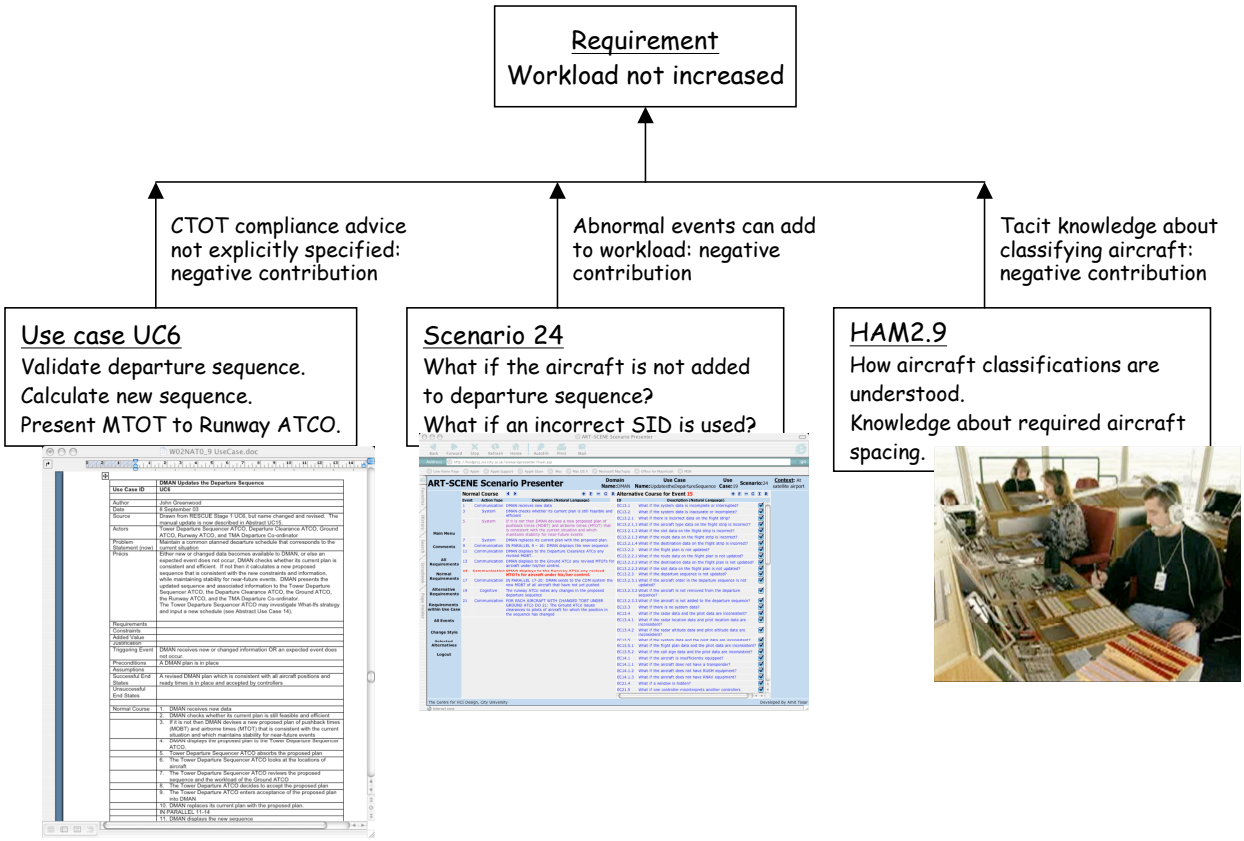

Figure 3: An example satisfaction argument constructed using data in DMAN use case specifications (on the left), in ART-SCENE scenarios (in the middle) and human activity descriptions (on the right)

4. Conclusions

In this paper, we have described the approach taken in RESCUE to the specification of operational requirements relating to the dependability of complex socio-technical systems in the domain of air traffic management. We have also proposed an extension to our approach which will strengthen our ability to demonstrate that high-level dependability requirements can, in fact, be satisfied by the system we specify. Because RESCUE aims at specifying operational requirements, i.e. requirements expressed in the stakeholders' terms, rather than detailed technical specifications, the resulting operational requirements document is not amenable to any form of formal dependability analysis. Preparation of a detailed system specification, safety case analysis and hazard analysis are all outside the scope of our process. However, even at the level of operational requirements, we believe that there is much to be gained from drawing on research in different disciplines, and the systematic application of human expertise, in order to map out the important aspects of dependability at a relatively high level. An important implication of this is the need for multi-disciplinary teams, with expertise in HCI and requirements engineering, as well as processes and design artefacts which allow effective collaboration between individuals from different backgrounds. In RESCUE, we believe that we have gone at least some way towards addressing this need.

5. References

[1] Diaper D (ed), *Task Analysis for Human-Computer Interaction*, Ellis-Horwood, 1989.

[2] Hammond J., Rawlings R. & Hall A., 2001, 'Will It Work?', Proceedings 5th IEEE International Symposium on Requirements Engineering, IEEE Computer Society, 102-109.

[3] Jackson M., 1995, 'Software Requirements and Specifications', Addison-Wesley.

[4] Jones S, Maiden N, Manning S and Greenwood J, "Human Activity Modelling in the Specification of Operational Requirements: Work in Progress", to be presented at

'Bridging the Gaps II', a workshop to be held at ICSE 2004.

[5] Maiden N. and Gizikis A, "Where Do Requirements Come From?", *IEEE Software*, Sept/Oct, 18(4), 10-12, 2001.

[6] Maiden N, and Rugg G, "ACRE: Selecting Methods For Requirements Acquisition", *Software Eng Journal* 11(3), 183-192, 1996.

[7] Maiden N, Jones S, Manning S, Greenwood J and Renou L, "Model-Driven Requirements Engineering: Synchronising Models in an Air Traffic Management Case Study", to appear in Proceedings of CaiSE, 2004.

[8] Mavin A and Maiden N, "Determining Socio-Technical System Requirements: Experiences with Generating and Walking Through Scenarios", Proceedings RE03, IEEE CS Press, 2003, 213-222.

[9] Neill C and Laplante P, "Requirements Engineering: The State of the Practice," *IEEE Software*, Nov/Dec, 40–45, 2003.

[10] Norman D, "The Psychology of Everyday Things", Basic Books Inc, 1988

[11] Rasmussen J, Pejtersen A and Goodstein L, "Cognitive Systems Engineering, John Wiley and Sons, 1994

[12] Reason J, "Human Error", Cambridge University Press, 1990

13] Robertson S. and Robertson J, *Mastering the Requirements Process*, Addison-Wesley-Longman, 1999.

[14] Schraagen J, Ruisseau J, Graff N, Annett J, Strub M, Sheppard C, Chipman S, Shalin V and Shute V, "Cognitive Task Analysis", RTO Technical Report 24, North Atlantic Treaty Organisation, 2000.

[15] Shneiderman B, "Designing the User Interface: Strategies for Effective Human-Computer Interaction", Addison Wesley, 1997

[16] Sutcliffe A, Maiden N, Minocha S, Manuel D, "Supporting Scenario-Based Requirements Engineering", *IEEE Trans Software Engineering*, 24(12), 1072-1088, 1998.

[17] Vicente, K., *Cognitive Work Analysis*, Lawrence Erlbaum Associates, 1999

[18] Yu E, "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering", in *Proc 3rd Int. Symp. on Requirements Engineering*, IEEE CS Press, 226-235, 1997