Enterprise Architecture analysis using Fault Trees and MODAF

Ulrik Franke, Pontus Johnson, Evelina Ericsson,
Waldo Rocha Flores, and Kun Zhu*

Industrial Information and Control Systems,
Royal Institute of Technology, Stockholm
{ulrikf, pj101, evelinae, waldorf, zhuk}@ics.kth.se

Abstract. Analysis of dependencies between information systems, business processes, and strategic goals is an important part of the discipline of Enterprise Architecture (EA). However, EA models typically provide only visual and qualitative decision support. This paper shows how EA frameworks for dependency analysis can be extended into the realm of quantitative methods by the use of techniques from Fault Tree Analysis (FTA). Using MODAF, the UK Ministry of Defence Architecture Framework as an example, we give a list of criteria for the extraction of a metamodel for FTA use, and provide such a metamodel for MODAF. Furthermore, we use this MODAF FTA metamodel to perform dependency analysis on a sample MODAF model.

Keywords: Enterprise Architecture, Fault Tree Analysis, dependency analysis, MODAF

1 Introduction

During the last decade, Enterprise Architecture (EA) has grown into an established approach for management of information systems in organizations. EA is model-based and does not only increase the general understanding of an organization's business and information system landscape, but also aids in decision making. Formal analysis approaches for EA [1], including sub-disciplines such as maintainability [2] and interoperability [3], are a growing field.

MODAF, the Ministry of Defence Architecture Framework, is designed to support the creation of an enterprise architecture for the British Ministry of Defence. Dependency analysis, i.e. the connection of high level concepts such as strategic capabilities (airlift, search and rescue, etc.) with the detailed and precise concepts that constitute low level technical systems (particular vehicles, radars, IT systems, etc.) has been identified as a key use of MODAF [4].

The present paper proposes an improvement and formalization of EA dependency analysis by methods from Fault Tree Analysis (FTA). FTA is a combinatorial model of systems dependability, widely used for safety and reliability

* The authors gratefully acknowledge the useful comments provided upon the present work by Mathias Ekstedt, Per Närman, Teodor Sommestad, and Johan Ullberg.

evaluations [5]. The method translates the failure behavior of a physical system into *events* connected by *arcs*. A visual model portrays the relationships in an accessible way, while a corresponding logical model enables quantitative evaluation. More details on FTA can be found in [6]. This paper aims to extend the EA analysis toolbox with the FTA method, enabling more powerful dependency analysis. MODAF is used as a running example.

The remainder of this paper is structured as follows. Section 2 contrasts the present paper with some related work. Section 3 is the locus of the main contribution, describing how to map MODAF models into fault trees. Section 4 gives a practical example of the method. Section 5 discusses the contribution and concludes the paper.

2 Related work

Model based generation of fault trees has been previously addressed. [7] describes generation of dynamic fault trees from data flow models. In [8], fault trees are extracted from UML system models. However, such fault tree derivation is often time-consuming and error-prone due to complex component interactions. Thus, [9] proposes more automation, viz. an algorithm for fault trees generation from Little-JIL process definitions. The present contribution goes beyond the scope of technical systems and processes directly supported by these systems, extending FTA into the realm of Enterprise Architecture.

3 A metamodel for the use of FTA in MODAF

In this section, the possibility of mappings between MODAF models and fault trees is discussed. Criteria for the use of FTA in MODAF are listed, and a metamodel based on these criteria is then presented.

3.1 Selection criteria

While FTA is a formal, mathematically precise method, MODAF models span wider fields (viz. all activities of the UK Ministry of Defence) and are as a consequence less clear-cut. We therefore define a list of selection criteria for identifying MODAF Meta Model (M3) objects appropriate for use in FTA.

Criterion 1 (Causal chain). M3 objects relevant to FTA must be part of a causal chain connecting technical systems to higher level concepts on the way to the strategic enterprise goals from the MODAF Strategic view.

Criterion 2 (Proper abstraction). M3 objects relevant to FTA must not make the overall architecture unnecessarily detailed.

Criterion 3 (Relevance to failures). M3 objects relevant to FTA must have at least one of the following properties (i) be the subject of failure (prospective fault tree events) or (ii) be able to transfer failures (prospective fault tree arcs).

Criterion 4 (Concreteness). M3 objects relevant to FTA must be on the level of concretion normal to FTA.

Criterion 5 (Binary fault states). M3 objects relevant to FTA must, when instantiated as fault tree events, have the binary fault states `non-failed` and `failed`.

3.2 Creating the metamodel

Using the M3 objects (entities and relations) thus selected, we build a metamodel for the use of FTA in MODAF. This entails a final delimitation:

Criterion 6 (Connectedness). M3 objects relevant to FTA must not be solitaire with respect to the whole set of such M3 objects. For event objects, this means that at least one arc object has to point to it. For arc objects, this means that it has to connect two event objects (not necessarily distinct).

The metamodel created by application of criteria 1 through 6 is illustrated in Fig. 1. Objects removed by criterion 6 are listed to the left in the figure.

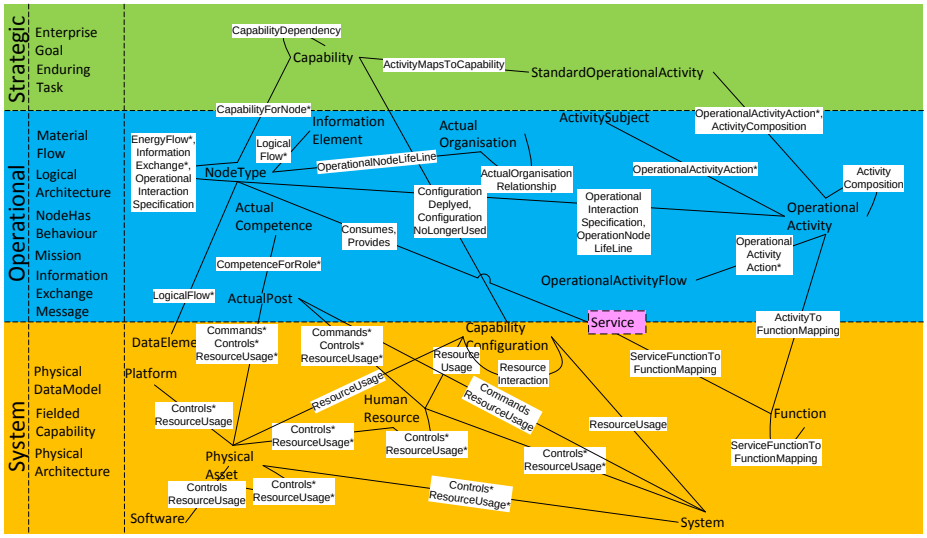

Fig. 1. A metamodel for the use of FTA in MODAF. The objects listed to the left are the solitaires excluded by criterion 6. Note the singleton `Service` element.

3.3 From metamodel to fault tree

Given a MODAF model and the metamodel illustrated in Fig. 1, the following algorithmic procedure allows the creation of a fault tree for dependency analysis.

Events Starting from an existing MODAF model, identify the entities that are instantiations of entities found in the FTA metamodel (Fig. 1). We now have an event set.

Arcs Similarly, identify the relations that are instantiations of relations found in the FTA metamodel (Fig. 1). We now have an arc set.

Gates For each relation connecting two entities, take the causally descendant entity and consider together the whole set of arcs connecting it to antecedent entities. Partition this set into OR or AND gates. The degenerate partitioning of the whole set into one single gate is allowed, as is partitionings including singletons, i.e. gates corresponding to the identity relation. The procedure is iterated until all arcs in the arc set have been mapped to gates.

Of course, this procedure cannot do away with the requirement of expertise and acquaintance with the system modeled. It does, however, provide a template and a coherent process for the generation of fault trees from MODAF models.

4 Dependency analysis of Search and Rescue capability

This section presents a practical example of how FTA formalism can be employed for dependency analysis of an existing MODAF model. We use a Search and Rescue (SAR) example created by the VEGA Group under contract to the UK Ministry of Defence [10], illustrating a maritime search and rescue operation at sea. The scenario involves a monitoring unit, a yacht in distress, a Command and Control (C2) center, a helicopter, and a lifeboat.

Now, we employ the FTA usage method from the previous section. Figure 2 illustrates the process of going from the initial MODAF model (1. in the figure), viz. a top to bottom dependency diagram from [10], via the FTA metamodel (2.) developed in the previous section, to a full blown fault tree (3.).

The `Maritime SAR` capability is described by `Search`, `Assistance`, and `Rescue` sub-capabilities, connected to the top event, i.e. failure of the maritime SAR capability, through an OR gate. OR gates are used for faults acting independently of each others, AND gates are used for failures acting in concert. For brevity, we have left the `Assistance` and `Rescue` capabilities undeveloped further down the tree. Instead developing the `Search` capability into sub-events, we proceed down through the Operations and System views until we reach basic events.

This example illustrates the process described in the previous section. The fault tree generated, as depicted in Fig. 2, enables not only qualitative dependency analysis, as does standard MODAF models, but also quantitative analysis. If the fault distributions of the components whose failures constitute the basic events are known, the impact of these distributions on the maritime SAR capability can be calculated. Decisions on procurement and maintenance of low level technical systems can thus be optimized with regard to the actual capability they are to support, rather than an intermediate proxy. Not the least, this allows rational prioritization when it comes to trade-offs between different systems.

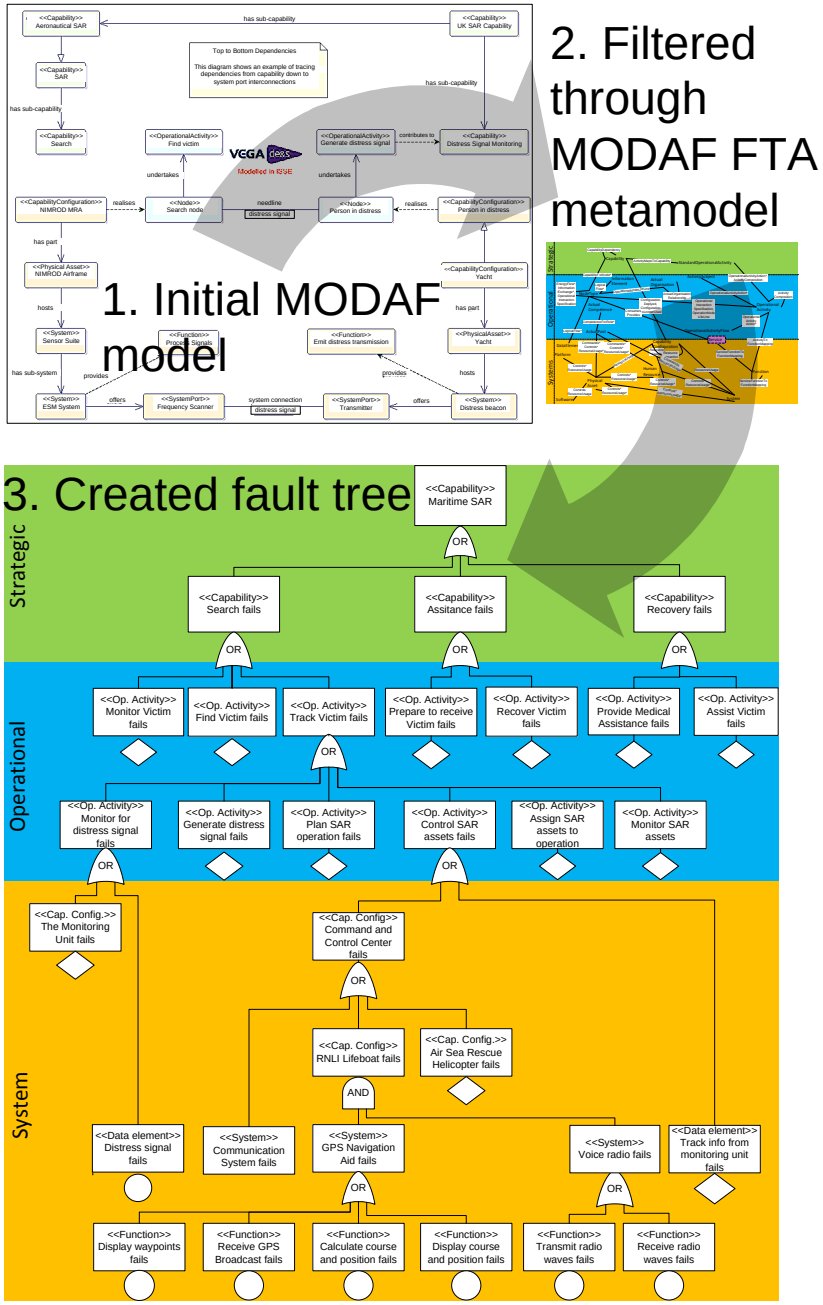

Fig. 2. The process of generating a fault tree interconnecting the System, Operational, and Strategic viewpoints of MODAF. The initial MODAF model is taken from [10].

5 Discussion and conclusions

MODAF was developed to support the modeling needs of the UK Ministry of Defence, while Fault Tree Analysis is a well-defined mathematical hazard analysis technique. The present contribution attempts to bridge this gap in several ways.

First, there is a gap of abstraction: the MODAF Meta Model is a metamodel, whereas FTA is generally performed on concrete instances – models – of technical systems. This is bridged by the creation of a smaller metamodel aimed specifically at FTA usage. Second, there is a gap of expressive power: FTA is much less expressive as a language than is MODAF. This is bridged by the algorithmic procedure for creating a fault tree out of an existing MODAF model.

The contribution of the present paper is three-fold: (i) The feasibility of expanding Fault Tree Analysis to strategic level enterprise architecture objects, usually considered too abstract for FTA, has been demonstrated. (ii) A metamodel for FTA use in MODAF has been proposed. (iii) A method for generation of fault trees, starting from the FTA metamodel in conjunction with an existing MODAF model, has been provided, thus allowing MODAF users to perform dependency analysis using the FTA technique.

References

1. Johnson, P., Lagerström, R., Närman, P., Simonsson, M.: Enterprise architecture analysis with extended influence diagrams. Information Systems Frontiers **9**(2) (May 2007)
2. Lagerström, R., Johnson, P.: Using architectural models to predict the maintainability of enterprise systems. In: Proceedings of the 12th European Conference on Software Maintenance and Reengineering. (April 2008)
3. Ullberg, J., Lagerström, R., Johnson, P.: A framework for service interoperability analysis using enterprise architecture models. In: IEEE International Conference on Services Computing. (July 2008)
4. Ministry of Defence: MOD Architecture Framework version 1.2.003. Technical report, Ministry of Defence, UK (September 2008)
5. Ericson, C.: Fault tree analysis – a history. In: 17th International System Safety Conference. (1999)
6. Codetta-Raiteri, D.: Extended Fault Trees Analysis supported by Stochastic Petri Nets. PhD thesis, University of Torino, Torino, Italy (2005)
7. McKelvin, M., Pinello, C., Kanajan, S., Wysocki, J., Sangiovanni-Vincentelli, A.: Model-based design of heterogeneous systems for fault tree analysis. In Rodney J. Simmons, Ph. D., C., Gauthier, N.J., eds.: 24th International System Safety Conference, System Safety Society (August 2006) 400–409
8. Pai, G.J., Dugan, J.B.: Automatic synthesis of dynamic fault trees from uml system models. In: Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE'02). (2002)
9. Chen, B., Avrunin, G.S., Clarke, L.A., Osterweil, L.J.: Automatic fault tree derivation from little-jil process definitions. In: SPW/ProSim. (2006) 150–158
10. VEGA Group (contracted by UK MOD): Search and Rescue Example. Available on `http://www.modaf.org.uk/vExamples/163/search-and-rescue-example`, accessed November 14, 2008 (Crown Copyright 2004-2008)