An Enterprise Modeling Framework for Complex Software Systems

Dr. Paolo Donzelli
Office of the Prime Minister

This article presents a goal-oriented, agent-based Enterprise Modeling Framework where advanced requirements engineering techniques are combined with software quality modeling approaches to provide an environment within which the stakeholders and the analysts can easily cooperate to discover, verify, and validate the requirements for a new software system. The framework assists and drives the stakeholders to an early definition of the desired system functionality and quality attributes, while supporting the redesign of the encompassing organization to better exploit the new system's capabilities. Although beneficial to a wider class of software systems, the framework has been applied here to improve the requirements engineering process for synthetic environments, which are complex software-intensive systems that typically comprise distributed interactive simulations of real-world systems. They are increasingly used to support vitally important operational, political, and economic decisions in a variety of industrial and governmental settings.

Systems requirements define what the system is required to do and the circumstances under which it is required to operate. The branch of software engineering concerned with all the activities involved in discovering, documenting, validating, and maintaining system requirements is *requirements engineering* (RE). Since difficulties with requirements are still a major contributory factor to project failures, leading not only to late and over budget deliveries but often also to systems significantly different from the stakeholders' expectations, RE is one of the most crucial steps in system development.

As technologies advance, they allow designers to envision systems that are increasingly becoming integral parts of the encompassing organizational processes. As this occurs, attention is being more and more focused on the very early phases of RE. The development of a successful system, that is a system able to address the stakeholders' real needs and suitable to evolve to meet ever-changing organizations' demands, relies on a firm understanding of the organizational context in which the system has to function. In other words, the system and its context need to be treated as a larger social-technical system, whose overall needs are the ones to be fulfilled [1].

Consequently in RE, appropriate process modeling techniques are typically advocated [2, 3] to help understand the organizational context (as it exists), envision possible solutions (as they could exist with the new system in place), and compare feasible alternatives. Within such a perspective, this article introduces an Enterprise Modeling Framework (EMF) explicitly designed to support discovery, verification, and validation of both *user-oriented* and *organization-oriented* system requirements [4, 5] by assisting dialogue between the analysts and the stakeholders and negotiation among the stakeholders.

In the remainder of this article, the EMF is introduced and its main characteristics are briefly described. Next comes some extracts from a practical application, and the article concludes by discussing some of the observed benefits.

> *"As technologies advance, they allow designers to envision systems that are increasingly becoming integral parts of the encompassing organizational processes."*

The EMF

The EMF is designed to allow analysts to deal with the *what* and *how* of the organizational context, i.e., the tasks performed by the organization, and the way in which they are performed. It also allows the analysts to model explicitly the why, i.e., the underlying reasons, expressed in terms of organizational goals. This enables the analysts and the stakeholders to focus on the *right* system for a given context, to design (or redesign) the encompassing process that fully exploits the system's capabilities [6], and to improve their capacity to identify, justify, and validate the system requirements [2, 3, 7].

In particular, the modeling effort is tackled by breaking the activity down into more intellectually manageable components, and by adopting a combination of different approaches on the basis of a common conceptual notation: *Agents* are used to model the organization, whereas *goals* are used to model agents' interactions.

According to the nature of a goal, we distinguish between *hard* goals and *soft* goals [2, 3]. For a hard goal, the achievement criterion is sharply defined (e.g., "buy a computer"); for a soft goal, it is up to the goal originator, or an agreement between the involved agents, to decide when the goal is considered to have been achieved (e.g., "buy a fast computer"). In comparison to hard goals, soft goals are highly subjective and strictly related to a particular context (what is meant by "fast computer?"). The EMF, therefore, spawns three interrelated modeling efforts: the organizational modeling, the hard goal modeling, and the soft goal modeling (see Figure 1 on page 24). In this way, separating goal modeling from organizational modeling helps reduce the problem complexity [4].

During *organizational modeling*, the organizational context is modeled as a network of interacting agents (any kind of active entity, e.g., teams, humans, and machines, one of which is the target system) [2, 8], collaborating or conflicting in order to achieve both private and organizational goals. Once identified, goals are translated and implemented through continuous refinement into operational terms as tasks with constraints.

The *hard goal modeling* seeks to determine how to achieve hard goals by decomposing them into more elementary subordinate hard goals and tasks, where tasks are well-specified activities that someone or something within the organization has to perform. So, for example, the hard goal "buy a computer" will be translated into a set of actions (tasks) necessary to procure a computer: organize a

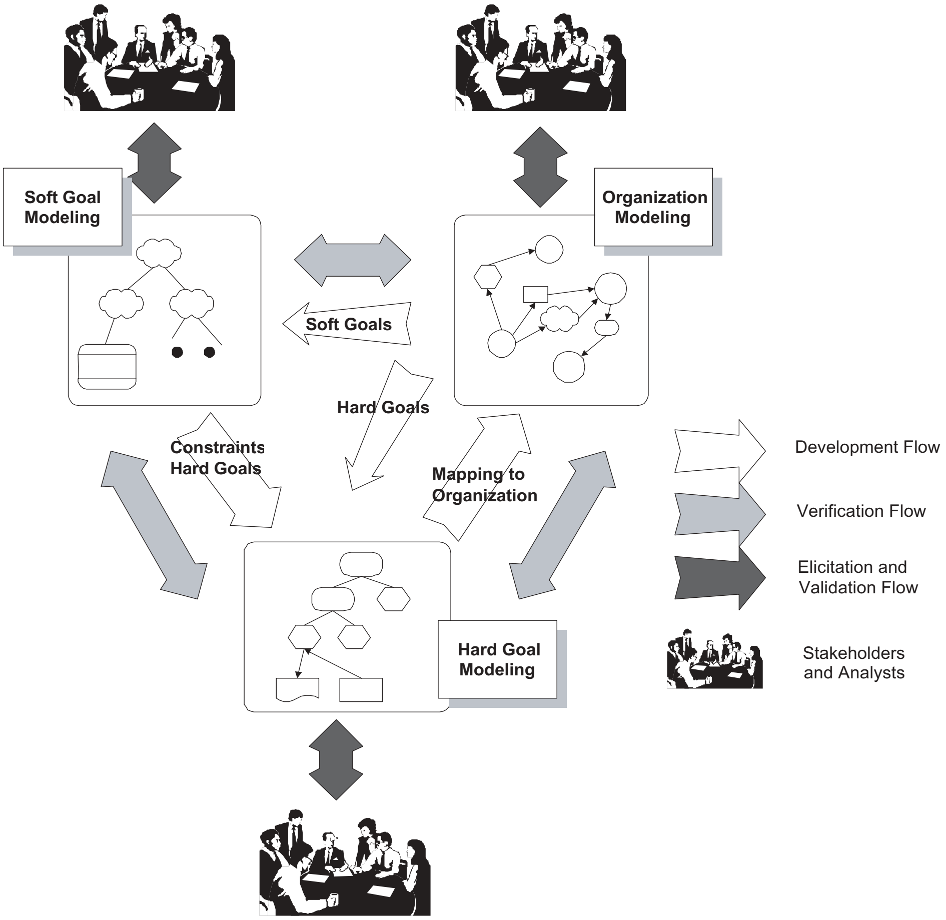
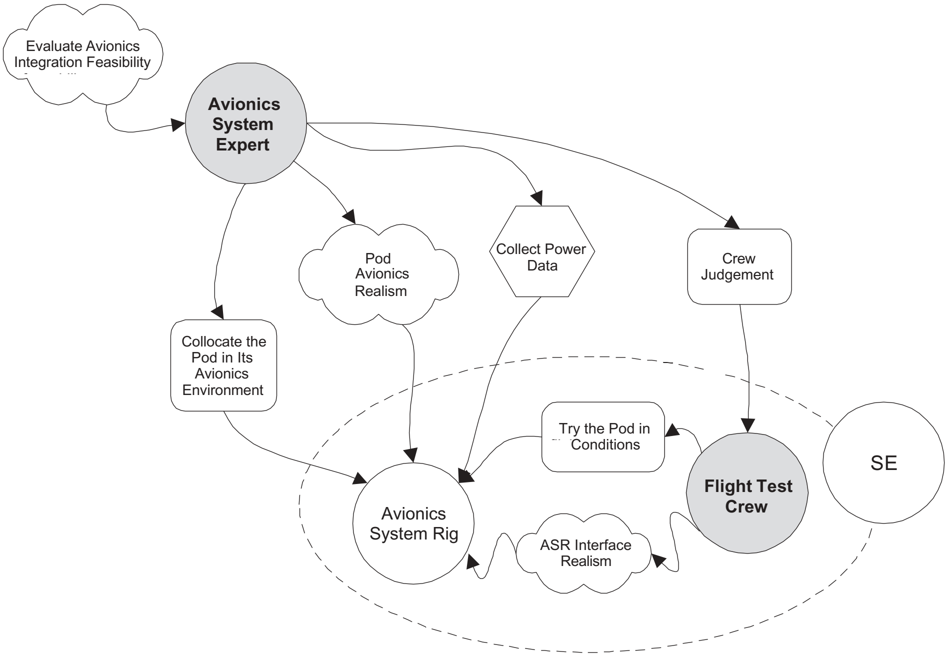


Figure 1: *The Enterprise Modeling Framework*

competitive tender, inform the bidders, etc. The EMF then draws upon these [7], although a more informal approach based on a combination of natural language and a semiformal graphical notation is preferred [2].

The *soft goal modeling* aims at producing operational definitions of the soft goals sufficient to capture explicitly the semantics that are usually assigned implicitly by the user [9], and highlight the system quality issues from the outset. A soft goal is refined in terms of subordinate soft goals, hard goals, tasks, and constraints. Constraints are associated with hard goals and tasks to specify the corresponding quality attributes. So, for example, the soft goal “buy a fast computer” will spawn the hard goal “buy a computer” and a set of associated constraints, e.g., CPU speed, memory size, cache characteristics, etc., that specify the computer quality attribute *fast* according to the stakeholders’ perception.

Figure 2: *The Organizational Model to Perform the Integration Study*



Applying the EMF

To investigate the feasibility of equipping an aircraft with a new avionics subsystem, a particular kind of ground-based equipment capable of providing the characteristics of the aircraft and of its components is usually adopted. Such equipment, or Avionics System Rig (ASR), is an example of a synthetic environment (SE) or of a subcomponent. In fact, a SE would typically consist of a mixture of real and simulated equipment [10], and would even encompass human operators. Furthermore, a SE can have different degrees of complexity, depending on the kind of information it provides (i.e., on the decision-making process that it has to inform).

In [5], EMF is applied to support the requirements engineering process for a hypothetical ASR that is needed to investigate the feasibility of providing an aircraft with a thermal pod. A thermal pod is an infrared device that is, for example, normally used on aircraft and helicopters dedicated to *search and rescue* and *anti-fire* roles. Although hypothetical, the case study is firmly based upon a real application where SE's have been used for a similar purpose [10]. For brevity in this article, the presentation is limited to a few extracts from this example application.

Shown in Figure 2 is a simple organizational model, within which the ASR is employed. Circles represent agents, and dotted lines are used to bind the internal structure of complex agents; i.e., agents containing other agents. Consequently, the *avionics system expert* is a simple agent that interacts with the SE to collect information necessary to assess the feasibility of equipping the aircraft with the thermal pod.

The SE agent in Figure 2, instead, is a complex agent encompassing the agents ASR and *flight test crew*. As stated earlier, it is typical of SE's to encompass human operators who have to interact with the simulated environment, but who are not direct stakeholders of the process that the SE is designed to support. In this case, for example, the flight test crew is part of the overall SE, so that the effects of the thermal pod integration on crew performance can be determined. However, it is the avionics system expert who is the primary (feasibility study) process owner.

Agents interact by exchanging goals and tasks. Clouds represent soft goals, rounded-rectangles represent hard goals, and hexagons represent tasks. Thus in Figure 2, the avionics system expert receives from the enclosing domain the soft goal of "evaluate avionics integration feasibility" for the new pod. Goals, tasks, and agents are connected by dependency-links, represented by arrowhead lines. An agent is linked to a goal when he/she needs that goal to be achieved; a goal is linked to an agent when he/she depends on that agent to be achieved. Similarly, an agent is linked to a task when it wants the task to be performed; a task is linked to an agent when the agent has to perform the task. By combining dependency-links, we can establish a dependency among two or more agents [2].

Each agent works as a goal transformer. Having received a goal, an agent will operate according to his/her own experience, knowledge, or position within the organization in attempting to achieve the goal. He/she will decide how to achieve the goal in terms of tasks and subordinate goals, and may choose to depend on other agents by passing out some of these tasks and subordinate goals. For example, the soft goal model in Figure 3 explains the behavior of the avionics system expert.

In order to achieve the received goals, the avionics system expert will need to "observe the pod in its avionics environment" and will require a *crew judgement*. The first soft goal will spawn some precise goals that the SE agent has to satisfy. These include the hard goal "collocate the pod in its avionics environment" with the associated soft goal "pod avionics environment realism," and the hard goal "monitor avionics system behavior," which leads to the task "collect power data."

The latter will require that a new agent, named flight test crew, will have to be included in the SE to operate with the ASR. In Figure 3, the A annotation on each decomposition line indicates that all these goals must be satisfied (and refined), whereas the goals and tasks in bold outline are those that the avionics system expert will pass out (see Figure 2) and are not further refined. To be able to express its opinion, the flight test crew in Figure 2 will require the possibility of *flying* the new pod, which places two other goals on the ASR: a hard goal "try the pod in flight conditions," and a soft goal "ASR interface realism."

By analyzing and refining the goals imposed on the SE agent, the final requirements for the ASR can be obtained. For example, by modeling the soft goals "pod avionics environment realism" and "ASR interface realism" (identified in Figure 2), a clear idea of the needs of the *avionics system expert* and *flight test crew* agents can be gained and translated into *functional* and *nonfunctional* requirements for the ASR.

Figure 4 provides this soft goal model for the ASR. It shows how the soft goal "pod avionics environment realism" from the avionics system expert spawns the following soft goals: "sensors and aircraft model realism," "avionics system realism," and the same "ASR interface realism" as was generated by the flight-test crew.

The soft goal "sensors and aircraft model realism" leads to a well defined set of constraints, which are represented by rounded-rectangles with one horizontal line and form nonfunctional requirements regarding both the sensors' tolerance and the aircraft model's capabilities. In this way, the flight test crew will be able to fly quite realistic low-level missions to test the pod.

The soft goal "avionics system realism" will translate into constraints that define what kind of equipment should be used. For example, the avionics system expert wants to be able to interface the pod with the real on-board computer.

Finally, the soft goal "ASR interface realism" will refine into both hard goals and constraints by mutual agreement of the two agents that have independently generated this same soft goal.

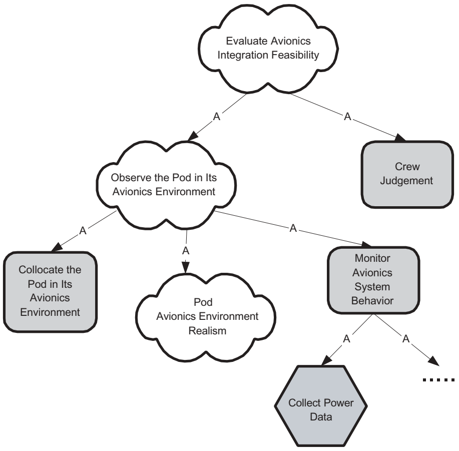

Evaluate Avionics Integration Feasibility
A
A
Observe the Pod in Its Avionics Environment
Crew Judgement
A
A
A
Collocate the Pod in Its Avionics Environment
Pod Avionics Environment Realism
Monitor Avionics System Behavior
A
A
Collect Power Data
.....

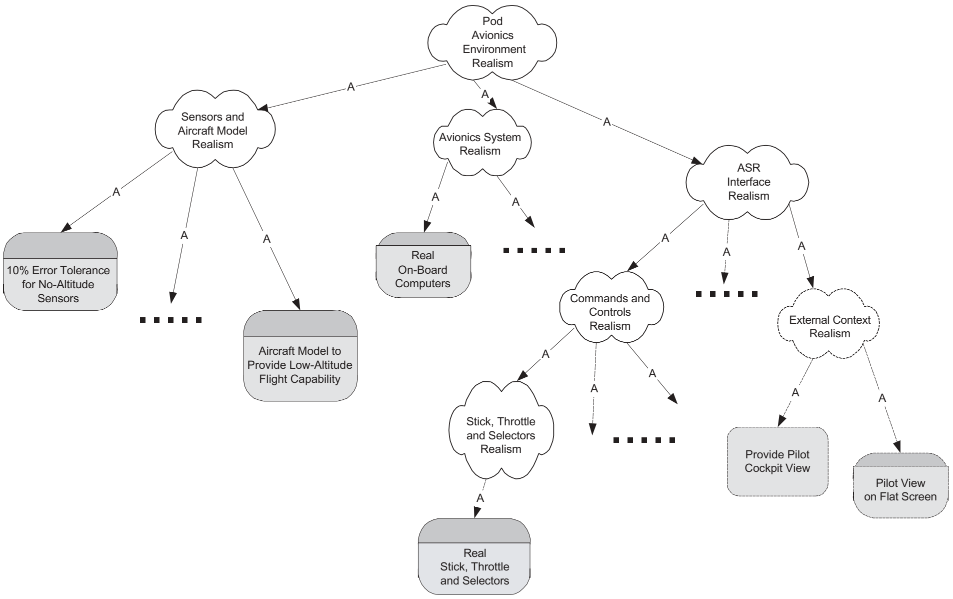
Figure 3: *The Soft Goal "Evaluate Avionics Integration Feasibility"*

In particular, resolving requirements clashes in this area result in the ASR providing the pilot cockpit view (a functional requirement) on a flat screen (a nonfunctional requirement). Importantly, this last goal demonstrates that different agents may have different opinions [11], and that *soft goal models allow the analysts to detect clashing requirements* at the early stages of a new system development and simultaneously provide a way to resolve them.

Conclusions

The application example described demonstrates the feasibility of the suggested approach. It also shows the benefits offered during the early phases of RE. This is the time when analysts and stakeholders have to cooperate to understand and reason about the organizational context within which the new system

Figure 4: *Soft Goals Imposed on the ASR*


Pod Avionics Environment Realism
A
A
A
Sensors and Aircraft Model Realism
Avionics System Realism
ASR Interface Realism
A
A
A
A
A
A
A
A
10% Error Tolerance for No-Altitude Sensors
.....
Aircraft Model to Provide Low-Altitude Flight Capability
Real On-Board Computers
.....
Commands and Controls Realism
.....
External Context Realism
A
A
A
A
A
Stick, Throttle and Selectors Realism
.....
Provide Pilot Cockpit View
Pilot View on Flat Screen
A
Real Stick, Throttle and Selectors

has to function. They must identify and formalize not only the system requirements, but also the organizational setting that better exploits the new system's capabilities.

In particular, benefits can be observed in terms of requirements discovery and early validation. Discovery and validation are improved because of the visibility of decisions made by the stakeholders as a result of explicit organizational and goal modeling. Each type of EMF model provides a specific knowledge representation vehicle that the analyst can use to interact with the stakeholders to capture requirements, reason about them, and eventually reach an accepted formulation.

Soft goal models force the stakeholders to reason about their own concepts of quality (for example, the concept of *realism* in Figure 4). Hard goal models allow the stakeholders to understand and validate their role within the organization, whereas organizational models provide management with a clear view of how the business process will be changed or affected by the introduction of the new system (see Figure 2).

The resulting models also suggest that EMF offers potential benefits in the *post-deployment phase*. The clear links established between organizational goals and system requirements, in fact, allow the analysts to quickly identify the influence of organizational changes on the final system requirements, supporting both *system maintenance* and reuse in different application contexts.

Although EMF addresses the early stages of the RE process, the possibility of combining its outcome with techniques more suitable for dealing with further system development phases has been investigated [12]. For example, initial results suggest that EMF can be usefully applied as a forerunner to object-oriented approaches such as those based upon the unified modeling language [13].

Finally, the general principles upon which the EMF is based allow it to be deployed for a larger class of computer-based information systems beyond SE's. For example in [14], EMF has been applied to define the requirements for a workflow-based document management system. Whereas in [15], it has been adopted to analyze the organizational impact and advantages of introducing a corporate smart card as enabling platform for accessing and using different e-services delivered by the organizational information technology system (e.g., digital signature, certified e-mail, documents management systems, etc.).◆

References

1. Fickas, S., and B. R. Helm. "Knowledge Representation and Reasoning in the Design of Composite Systems." IEEE Transactions on Software Engineering 18.6 (June 1992).
2. Yu, E. Why Agent-Oriented Requirements Engineering? Proc. of 3rd Workshop on Requirements Engineering for Software Quality. Barcelona, Catalonia, June 1997.
3. Loucopulos, P., and V. Karakostas. System Requirements Engineering. UK: McGraw Hill, 1995.
4. Donzelli, P., and M. R. Moulding. Developments in Application Domain Modeling for the Verification and Validation of Synthetic Environments: A Formal Requirements Engineering Framework. Proc. of the Spring '99 Simulation Interoperability Workshop. Orlando, Fla., Mar. 1999.
5. Donzelli, P., and M. R. Moulding. "A Unified Approach to the Verification, Validation and Accreditation of Synthetic Environments: A Requirements Engineering Framework." Cranfield University Technical Report Ver. 1. SE027E/TR1. Dec. 1999.
6. Hammer, M., and J. Champy. Reengineering the Corporation: A Manifesto for Business Revolution. London: Nicholas Brealey Publishing, 1995.
7. Dardenne, A., A. Van Lamsweerde, and S. Fickas. "Goal-Directed Requirements Acquisition." Science of Computer Programming 20 (1993).
8. D'Inverno, M., and M. Luck. Development and Application of an Agent-Based Framework. Proc. of the First IEEE International Conference on Formal Engineering Methods. Hiroshima, Japan, 1997.
9. Cantone, G., and P. Donzelli. "Goal-Oriented Software Measurement Models." European Software Control and Metrics Conference. Herstmonceux Castle, East Sussex, UK, Apr. 1999.
10. Donzelli, P., and R. Marozza. Laser Designation Pod on the Italian Air Force AMX Aircraft: A Prototype Integration. Proc. of the NATO/RTO SCI Joint Symposium on Advances in Vehicle Systems Concepts and Integration. Ankara, Turkey, Apr. 1999.
11. Van Lamsweerde, A., R. Darimont, and E. Letier. "Managing Conflicts in Goal-Driven Requirements Engineering." IEEE Transactions on Software Engineering 24. 11 (Nov. 1998).
12. Donzelli, P., M. R. Moulding. Application Domain Modeling for the Verification and Validation of Synthetic Environments: From Requirements Engineering to Conceptual Modeling. Proc. of the Spring 2000 Simulation Interoperability Workshop. Orlando FL., Mar. 2000.
13. Rumbaugh, J., I. Jacobson, and G. Booch. The Unified Modeling Language Reference. Addison-Wesley, 23 Dec. 1998.
14. Antonelli, C., P. Donzelli, Masroianni, R. Setola, S. Vinti, and S. Tucci. A Web-Based Workflow Solution to Support the Italian Government Agenda Definition Process. Proc. of the Italian Automatic Computation Association 2000 Conference – Le Technologie dell'Informazione e della Communcazione come motore di sviluppo del Paese. Taomina, Italy, 27-30 Sept. 2000.
15. Donzelli, P., and R. Setola. Putting the Customer at the Center of the IT System – A Case Study. Proc. of the EuroWeb 2001 Conference – The Web in the Public Administration. Pisa, Italy, 18-20 Dec. 2001.

About the Author

Paolo Donzelli, Ph.D., is an advisor with the Department of Informatics of the Office of the Prime Minister in Rome, Italy. A former engineering officer with the Operational Testing Center of the Italian Air Force, Dr. Donzelli was a senior research fellow with the Computing Information Systems Engineering Group, Royal Military College of Science, Cranfield University, United Kingdom. He has a variety of interests in the software engineering area, and his doctorate thesis was in software process quality modeling.

Presidenza del Consiglio
dei Ministri
Ufficio per l'Informatica,
la Telematica e la Statistica
Via della Stamperia 8, 00187
Roma, Italy
E-mail: p.donzelli@governo.it