A goal-driven and agent-based requirements engineering framework*

Article *in* Requirements Engineering · February 2004

DOI: 10.1007/s00766-003-0170-4 · Source: DBLP

1 author:

Paolo Donzelli
Government of Italy
62 PUBLICATIONS **620** CITATIONS

SEE PROFILE

Requirements Eng (2004) 9: 16–39
DOI 10.1007/s00766-003-0170-4

ORIGINAL ARTICLE

Paolo Donzelli

A goal-driven and agent-based requirements engineering framework*

Received: 15 October 2001 / Accepted: 28 February 2003 / Published online: 12 July 2003

Abstract The paper presents a requirements engineering framework (REF), where advanced requirements engineering techniques are combined with software quality modelling approaches to provide an environment within which the stakeholders and the analysts can easily cooperate to discover, organise, reconcile and validate the requirements for a new system. By adopting a basic and essential graphical notation, and a clear top-down methodology, REF allows for an effective involvement of the stakeholders, assisting and driving them to an early definition of the desired system's functionalities and quality attributes, while supporting the redesign of the encompassing organisational context to better exploit the new system's capabilities. As a case study, REF is applied to support the requirements engineering process for a complex software-intensive simulation system. Results demonstrate the feasibility of REF and the benefits it offers to the requirements engineering process, but also to the subsequent system development phases. As illustrated through the case study REF can, in fact, be usefully applied as a forerunner for unified modelling language (UML)-based approaches.

Keywords Agents · Non-functional requirements · Quality models · Requirements engineering · Soft and hard goals · UML

*Part of this work was completed while the author was Senior Research Fellow with the Computing Information Systems Engineering Group, at the Royal Military College of Science, Cranfield University (UK)

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

1 Introduction

As technologies advance, empowering engineers to envision systems that increasingly tend to become integral parts of the encompassing organisational processes, attention is being more and more focused on the very early phases of requirements engineering (RE). The successful implementation of a new software system relies on a firm understanding of its application context and, above all, relies on our ability of envisioning its impact on its application context [1, 2, 3].

Envisioning the impact that a software system will have on its application context is a difficult task. Organisations are made of people, and the goals, the expectations and the needs of all the stakeholders, directly or indirectly involved (e.g., the final users, customers, business and information technology managers, procurements people, maintainers, developers, etc.) have to be explicitly addressed, analysed and captured from the outset of the project. The software system and its application context form a larger social-technical system that has to be treated and analysed as a whole: the overall needs of such social-technical systems are the ones that have to be fulfilled [4]. Consequently, appropriate organisation modelling techniques are typically advocated [2, 3, 4, 5, 6, 7, 8, 9]:

- to allow the navigation of the (social-technical) solution space, and to formulate the solution in social-technical terms; the requirements engineering process outcome may affect both the organisation (e.g., the internal structure, personnel and existing systems), as well as the investigated 'target' system.
- to allow reasoning about dependencies in the social-technical system: that is, establishing a clear link between the enterprise needs and constraints, and the software system properties.

In such a perspective, the paper proposes a requirements engineering framework (REF) explicitly designed to support discovery, refinement, definition,

reconciliation and early validation of both user-oriented and organisation-oriented requirements [8, 10, 11, 12], by assisting the negotiation among the stakeholders and between the analysts and the stakeholders and by providing reconcilable user-oriented views of the application context. REF combines advanced requirements engineering techniques (i.e., agents [13, 14, 15] and goals [16, 17]) with software quality modelling approaches [18, 19], to capture the stakeholders' perception of quality from the beginning of a new project, and to produce agreed-upon and implementable functionalities and constraints. While gathering requirements, the analysts build models [19] of the relevant system quality attributes, thus capturing, formalising and eventually reconciling the different stakeholders' points of view on product and process characteristics. This supports validating not only the functionality of the system, but also the negotiated quality attributes, according to the customers' needs and the developers' constraints.

This paper is organised as follows: the next section introduces REF, by briefly describing its main characteristics, capabilities and basic notations, while the following section reports on the background upon which REF is based and highlights the main novelties of the approach. Next, a REF application is presented, with extracts from an articulated case study, where REF has been applied to support the requirements engineering process for a synthetic environment (SE). SEs are a specific class of software-intensive simulation systems, increasingly used to support operational, political and economic decisions, in a variety of industrial and governmental settings. The next section shows how REF can support further system development phases, in particular how the proposed framework can act as a forerunner for unified modelling language (UML)-based development approaches [20]. The last section concludes the paper and outlines plans for future work.

2 The REF

REF is designed to allow analysts to deal with the 'what' and the 'how' of the organisational context (i.e., the tasks performed by the organisation, and the way in which they are performed), but also to explicitly model the 'why' (i.e., the underlying reasons) expressed in terms of organisational goals. This enables the analysts and the stakeholders to focus on the 'right' system for a given context, to design (or re-design) the encompassing organisational process that fully exploits the system's capabilities [21], and which improves the analysts' and stakeholders' capacities to identify, justify and validate the system requirements.

REF tackles the modelling effort 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.

The key modelling abstractions are those of agents and goals. The organisational context is modelled as a network of interacting agents (both humans and machines), collaborating or conflicting in order to achieve both individual and organisational goals. Any agent may generate its own goals, may operate to achieve goals on the behalf of some other agents and may decide to collaborate with other agents on a specific goal, while clashing on some other goals. According to the nature of a goal, a distinction is made between hard goals and soft goals [5, 15, 16]. A goal is classified as hard when its achievement criterion is sharply defined (e.g., 'buy a computer'). For a soft goal, instead, it is up to the goal originator, or to 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 can be highly subjective and strictly related to a particular context; they enable the analysts to highlight quality issues (e.g., the concept of a 'fast computer') from the outset, making explicit the semantics assigned to them by the stakeholders. While a hard goal will lead to a set of functions (functional requirements) that the software system will have to provide, a soft goal will be refined into more precise subordinate soft goals to reduce its initial fuzziness, until, eventually, it can be expressed as a set of hard goals with the associated constraints, non-functional requirements [2].

Distinguishing goal modelling from organisational modelling, and then, further distinguishing between hard goal modelling and soft goal modelling helps in reducing the complexity of the modelling effort. REF, therefore, tackles the modelling effort by supporting three interrelated activities (see Fig. 1): the organisational modelling, the hard goal modelling and the soft goal modelling. Organisation modelling, hard goal modelling, and soft goal modelling, as shown in Fig. 1, do not exist in isolation; rather they are different views of the same modelling effort, linked by a continuous flow of information, schematised as development, elicitation and validation flows. Elicitation and validation flow shows where interaction with the stakeholders occurs, whereas the development flow shows how information discovered in one model may feed the others, in a continuous loop.

REF allows the analysts to handle and control a large amount of information [22], while building a model of the organisational context at the desired level of detail. Through continuous interaction with the stakeholders, supported by the different models, the analysts will deal first with the high level organisational structure, and then will descend step by step into the details of the application context of the new system, until the focus is placed upon the single agents and their role within the organisation (which includes the system itself).

As shown in Fig. 1, REF adopts a cyclical development flow. In particular, once an initial model of the organisation is built (in the start-up phase), the REF process evolves in a cyclical way through two main

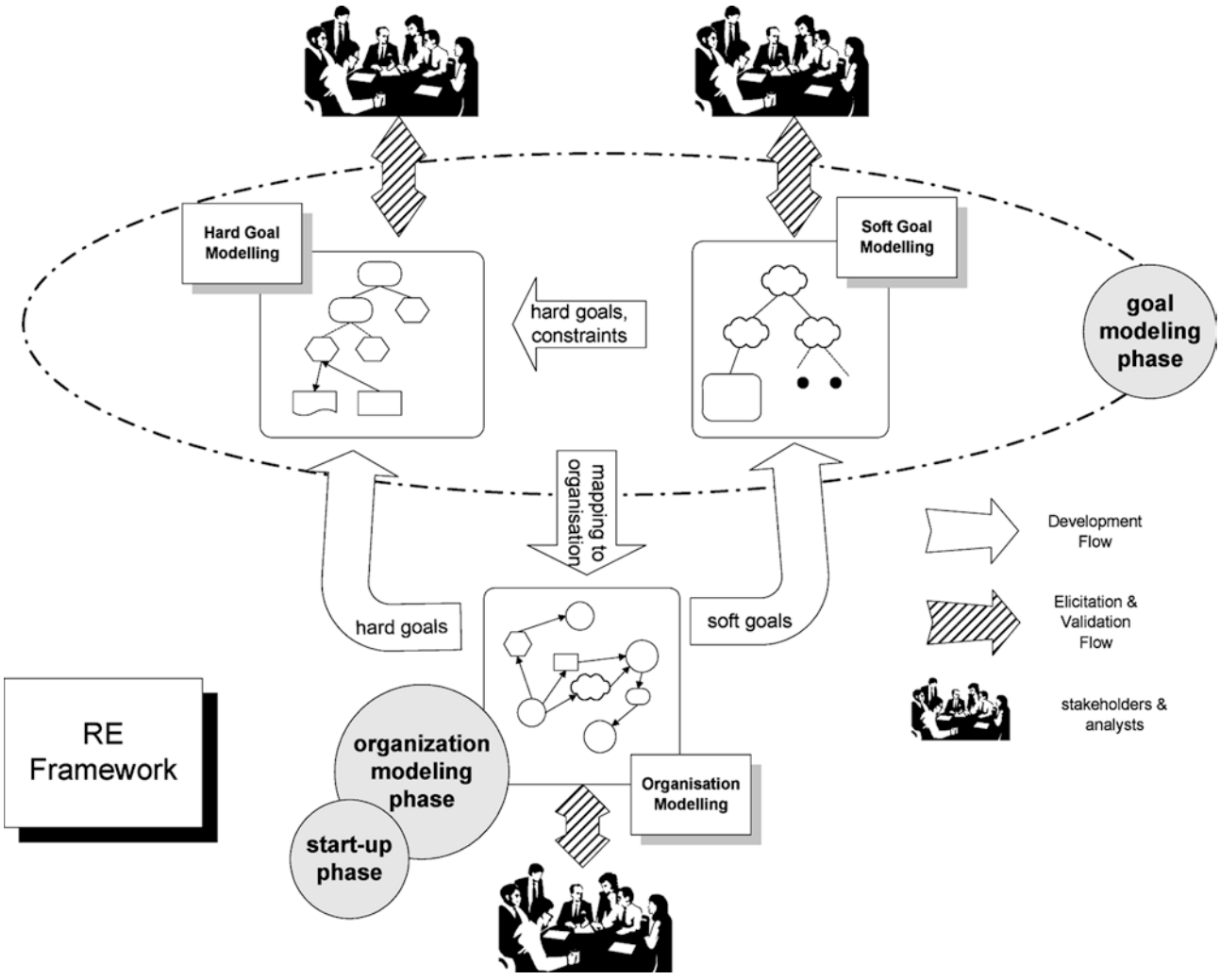


Fig. 1 An overview of the REF

phases: a) the goal modelling phase, during which the soft and hard goals discovered during organisation modelling are refined; and b) the organisation modelling phase, during which the analysts use the information gained during the previous step (indicated by the 'mapping to the organisation' arrow) to enrich and extend the initial organisational model: i.e., to replace the goals with their models, and to introduce the new agents identified as relevant to achieve those goals. New agents usually lead to new goals, triggering the goal modelling phase again. Such a cycle is continued until the desired (and needed) level of details is reached.

2.1 The start-up phase (modelling the organisation)

During organisation modelling, the organisational context is analysed and the agents and their interactions are identified. The basic elements from which an organisation model can be built are: agents, soft and hard goals, tasks, resources and dependency links (see Fig. 2).

An agent may represent any kind of active entity within the organisation: a human operator, a team, a process or a machine (e.g., the target system). Depending on the agent, or on the abstraction level the analysts are dealing with (or are interested in), the agent can be modelled as a complex agent, i.e., an agent that contains

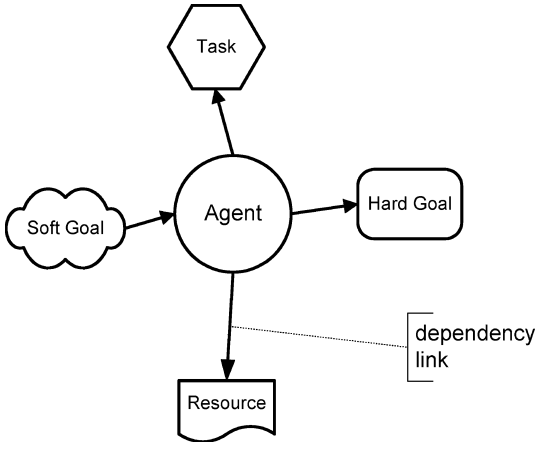


Fig. 2 Basic elements of an organisation model

other agents, or a simple agent, i.e., an elementary processor in the organisation model.

Agents interact through goals, tasks and resources: a goal (hard and soft) defines a desired status within the organisation, without specifying how to achieve it; a task is a well-specified activity, leaving little room for initiative; a resource represents any entity in the organisation, both physical and informational, that can be used by an agent.

Goals, tasks and resources can be linked to agents by dependency links. A dependency link (see Fig. 2) is a directed link that goes from the depender to the dependee; it can connect an agent to a hard or soft goal, a task, a resource, or vice-versa. In particular, an agent is linked to a goal, a task or a resource when it depends, in some way, on that goal to be achieved, that task to be performed or that resource to be provided; a goal/task/resource is linked to an agent when it depends on that agent to be achieved/performed/provided.

For example, in Fig. 2, the soft goal depends upon the agent in order to be achieved. At the same time, the agent needs the hard goal to be achieved, the task to be performed and the resource to be provided in order to fulfill its purpose. By combining dependency links, two agents may be connected (indirectly) to indicate that a dependency [23] exists between them. As an example, a simple organisation model is depicted in Fig. 3. Here, the agent 'Tom' needs a fast computer to be bought and depends upon the agent 'Fred' to achieve such a goal.

2.2 Goal modelling phase

During goal modelling, all the goals revealed during organisation modelling (both soft and hard goals) are refined, through interaction with the involved agents, until an implementable set of functions and constraints is reached.

As example, the soft goal 'buy a fast computer' will be modelled, which provides us the opportunity to illustrate both soft and hard goal modelling.

2.2.1 Modelling a soft goal

Soft goal modelling aims at producing operational definitions of the soft goals, sufficient to capture, and make explicit, the semantics that are usually assigned implicitly by the user [5, 12], and to highlight the system quality issues from the outset. Modelling a soft goal will try to reduce its initial fuzziness by refining it into more precise subordinate soft goals until, eventually, it can be expressed as a set of implementable hard goals, tasks and constraints, that are in a set of actions that have to be performed, and in a set of constraints characterising those actions.

Fig. 3 An example organisation model

An initial model for the soft goal 'buy a fast computer' is depicted in Fig. 4. Here the soft goal 'buy a fast computer' spawns the hard goal 'buy a computer' and a set of associated constraints having to do with the computer (e.g., CPU clock > 0.6 GHz, memory size = 256 MB, multi-level cache, a RISC architecture, etc.), which, by specifying the meaning of the computer's quality attribute 'fast', will 'constrain' our possibilities to achieve the hard goal 'buy a computer'. In other words, the agent 'Fred' has to buy a computer, but not just any computer; a computer that satisfies all the constraints specified by the agent 'Tom'.

As in an organisation model, a arrowhead line indicates a dependency link. A soft goal depends on a subordinate soft goal, hard goal, task or resource when it requires that goal, task or resource to be achieved, performed, or provided in order to be achieved. A dependency link tells us that some kind of dependency exists, but it does not go any further. For example, it does not allow the modeller to specify alternative or collaborating refinement paths. Although this abstraction level is sufficient at the early stage of the modelling effort, when little is known about the application context, more precise refinement links could turn out to be useful at a later (more detailed) modelling stage. For this reason, information can be added to a dependency link to specify whether it represents an AND or an OR refinement. As shown in Fig. 4, the 'A' annotation on each dependency link indicates that all the corresponding tasks must be satisfied (the 'AND' refinement). The 'O' annotation on each dependency link, instead, is used to indicate that at least one of the corresponding goals or tasks must be satisfied (the 'OR' refinement).

To proceed in the analysis, at this point, we have to turn to the hard goal 'buy a computer', which can be refined according to the hard goal modelling approach described below.

2.2.2 Modelling a hard goal

The hard goal modelling seeks to determine how to achieve the hard goals: a hard goal model allows the analyst to express a combination of subordinate hard goals, tasks and resources needed to achieve an initial hard goal (by definition of hard and soft goals, no soft goals can emerge from a hard goal refinement). Of course, the granularity of refinement of a hard goal into subordinate hard goals and tasks depends on the level of capability and autonomy of the agent upon which the goal is placed. So, for example, while the hard goal 'buy a computer' could be clear enough for a human agent, it might need to be translated into a set of actions (tasks)

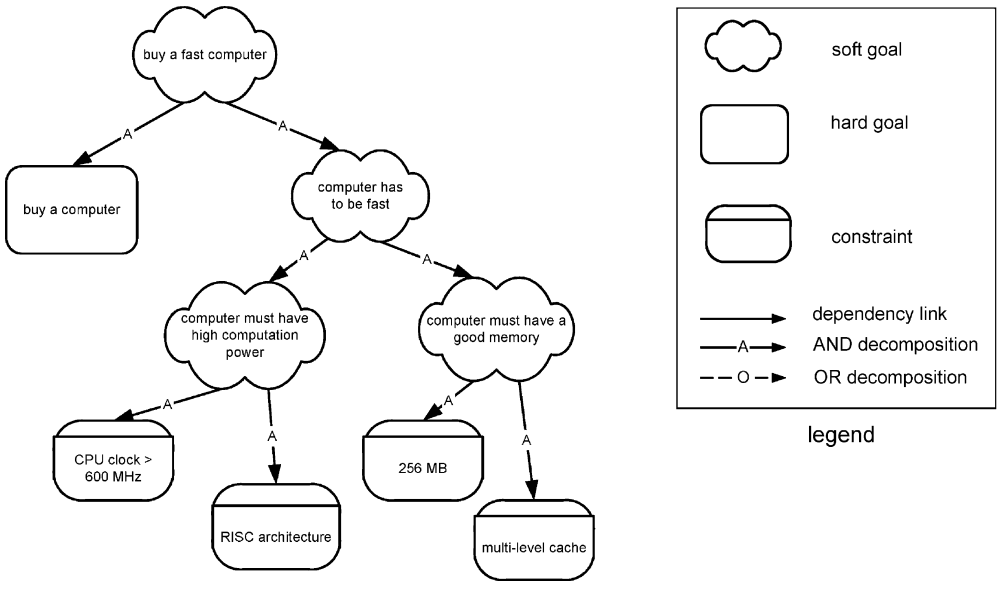


Fig. 4 The 'buy a fast computer' soft goal model

to be enacted by an organisation: 'organise a competitive tender', 'inform the bidders', etc.

An example model for the goal 'buy a computer' is depicted in Fig. 5. According to such a model, in order to buy a computer, it will be necessary first to write the specifications for the computer to be bought, then to choose between two possible options: to organise a tender, which implies informing the bidders and to select the winner, or to buy the computer directly from a trusted provider.

Again, a arrowhead line indicates a dependency link (and AND/OR refinements). A hard goal depends on a subordinate hard goal, task or resource when it requires that goal, task or resource to be achieved, performed or provided in order to be achieved. Similarly, a task depends on a subordinate hard goal, task or resource, when it requires that goal, task or resource to be achieved, performed, or provided.

2.2.3 Completing the soft goal model

Once completed, the hard goal model in Fig. 5 tells us how to buy a computer, but nothing more about how to buy a fast computer. However, it can be used to complete the model of the soft goal 'buy a fast computer'.

As a first step, we have to replace the hard goal in Fig. 4 with its model, as shown in Fig. 6, and then we have to use the emerging constraints. The constraints that emerge during soft goal modelling, and that capture part of the goal softness, in fact, cannot live in isolation, but must be linked to the hard goal, the task or the resource where they will be dealt with. Constraints, in other terms, simply specify quality attributes of entities or actions. In particular, in our example, the constraints emerging from the 'buy a fast computer' soft goal model specify the quality characteristics the computer must have in order to be considered as a fast computer. Given

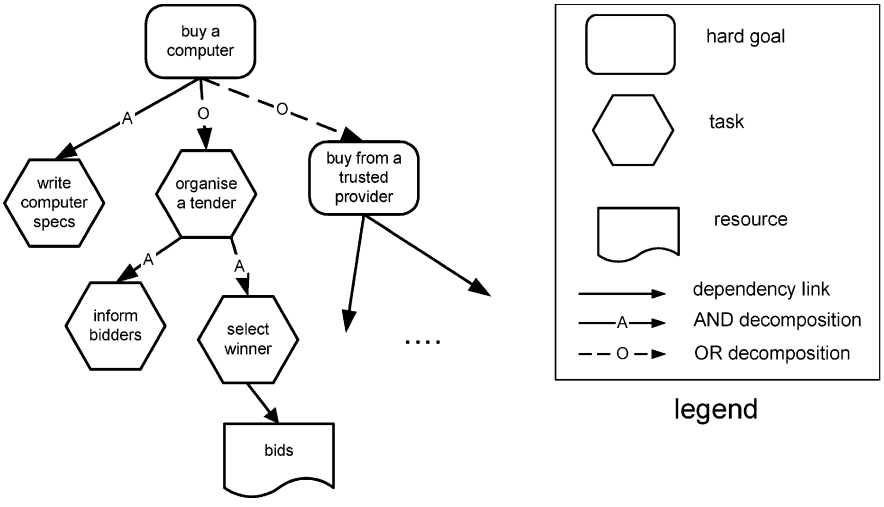


Fig. 5 The 'buy a computer' hard goal model

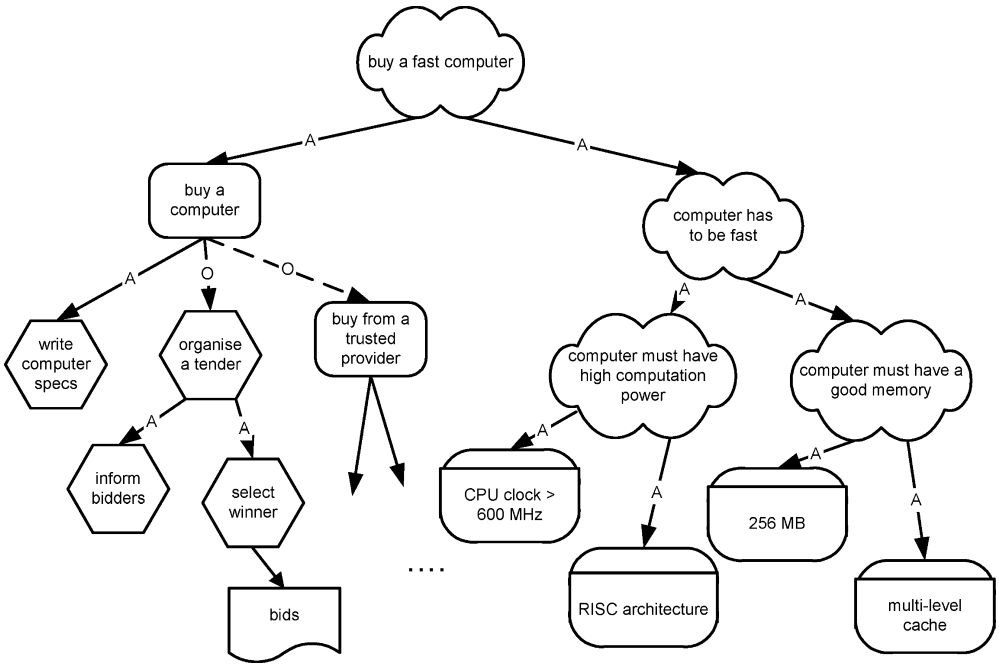


Fig. 6 Building the 'buy a fast computer' soft goal model

that, according to the hard goal model, the computer characteristics have to be described in the specification document (to be used either for organising a tender or for buying it from a trusted provider), we can assume that such constraints will be dealt with while writing the specification document.

This solution is expressed in Fig. 7, where the constraints are linked to the 'write the computer specification' task. Again, a simple dependency link is used to link the constraint to the hard goal or task where the constraint has to be dealt with: i.e., the constraint will depend upon that goal or task in order to be fulfilled.

2.3 Organisation modelling phase

While modelling the goals, the analysts gather information about the agents' behaviour, and in general, about the organisation functioning. All this information may be used to enrich and complete the initial organisational

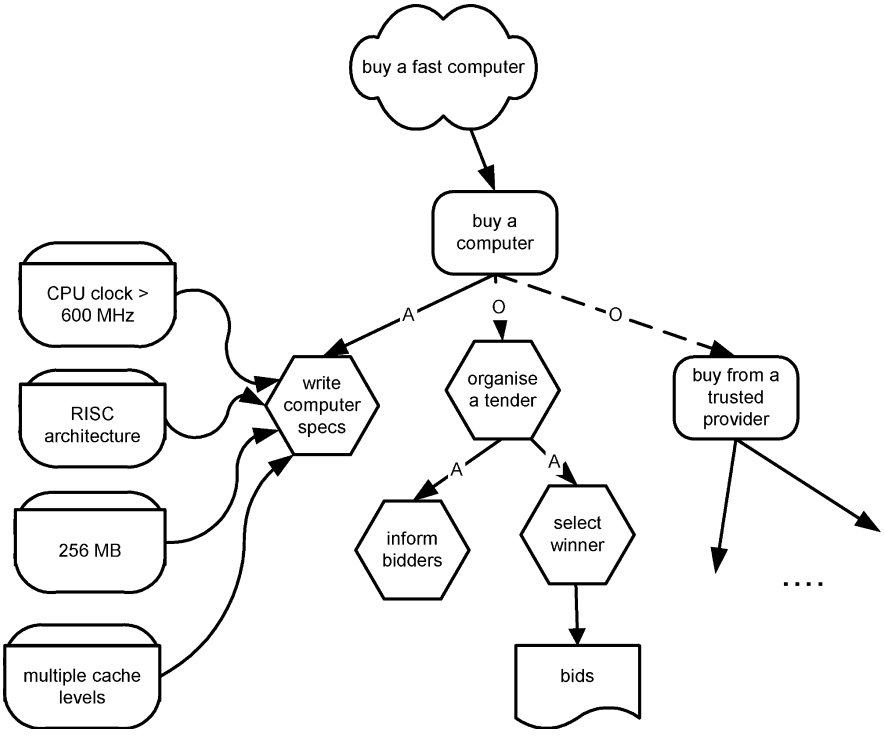


Fig. 7 The final 'buy a fast computer' soft goal model

model (e.g., by replacing the initial model's goals with their models), but, above all, to drive the analysis further, until the desired level of detail is reached. While reasoning with an agent about how to achieve a goal, in fact, that agent may decide to depend upon a new agent. In this way, new agents may have to be taken into account, and new agents usually lead to new goals, triggering a new development loop (see Fig. 1).

However, this is not the case in our simple example. By replacing the soft goal of Fig. 2 with its model, we obtain the organisation model of Fig. 8 that we can assume to be at the right level of detail: it clearly shows, in fact, how the agent 'Fred' will have to operate in order to buy a fast computer according to the needs of the agent 'Tom'.

3 The REF background

REF combines advanced requirements engineering techniques (i.e., agents [13, 14, 15] and goals [16, 17]) with software quality modelling approaches [18, 19]. The basic idea is to exploit the advantages provided by modelling mechanisms such as those of agent, goal and dependency, which have been recognised as suitable mechanisms to support a smooth transition from high-level organisational needs to system requirements, and the advantages provided by quality modelling techniques, which allow capturing the stakeholders' perception of quality at the beginning of a new project, by producing agreed-upon and implementable functionalities and constraints.

3.1 Ideas from RE techniques

REF is strongly inspired by i*, the modelling framework suggested by Eric Yu [23] for RE [13, 16] and business analysis and re-engineering [24]. However, REF introduces a clear methodology to drive the process of requirement discovery, definition, refinement and reconciliation, and adopts the i* notational ingredients at a basic and essential level [8, 10]. REF, in fact, introduces some simplifications and tends to adopt a more pragmatic approach in order to obtain a greater and more active involvement of the stakeholders during the requirements discovery, elicitation and formalisation process. The basic idea is to enable the stakeholders to easily and quickly grasp the notation in order to become more active participants from the beginning of the project, as found during the case study.

As a first step towards this objective, REF adopts a strict top-down approach (see the previous section), during which the analysts, in close cooperation with the stakeholders, drill through the organisation, until reaching the desired level of detail, by using three different (but similar) modelling tools: organisation models, soft goal models and hard goal models.

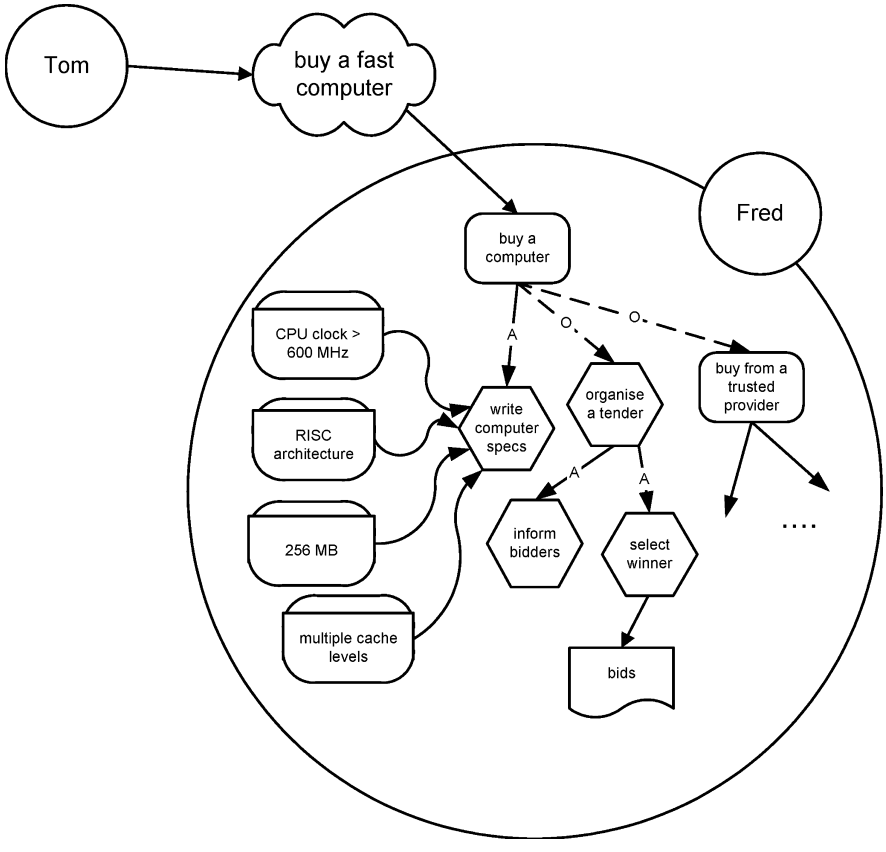


Fig. 8 The final organisation model

Then, with respect to i*, REF opts for the following simplifications:

1. The adoption of only one kind of agent (so that no distinction is made between agent, role and position).
2. The adoption of only one kind of link (i.e., the dependency link, a simple arrow!) both for modelling the dependencies in the organisation models (no distinction is made between resource, task, hard goal and soft goal dependencies) as well as for modelling the goal decomposition dependencies in the soft and hard goals models (no distinction is made between task-decomposition and the mean-ends links)

In the latter case, the direction of the arrow is reversed with respect to that used in i*, allowing for a more intuitive reading of the REF goals models (corresponding now to the direction to the analysis flow, i.e., top-down).

Such choices not only allow a better decoupling between agents, on the one hand, and goals, tasks and resources, and on the other hand, making it easier to model common situations when more agents depend upon a goal (task or resource), or vice-versa, but such choices also introduce a more natural and more smooth flow of dependencies: from dependencies between actors in the organisation models, to the decomposition dependencies in the hard and soft goals models.

Finally, unlike i*, REF emphasises the operational role that soft goals can play in deriving the requirements of a new system. Soft goals, in fact, beyond playing an important role in supporting reasoning between alternatives (as in i*), from the early stages of a project, can also provide a systematic and very pragmatic way of handling non-functional requirements [25, 26] , while dealing with functional ones. REF, therefore, clearly recognises the need to explicitly 'resolve' soft goals, to turn them into more manageable constraints. In such a way, soft goal models become one of the main modelling efforts and also a powerful tool in analysts' hands to detect and resolve clashing requirements, through an active and informed stakeholder participation.

3.2 Ideas from quality modelling techniques

To refine a soft goal, the analysts have first to specify the actions that are usually implied, and sometimes hidden, in the goal (e.g., to buy a computer in the soft goal 'buy a fast computer'), and then they have to make operational its softness (e.g., in identifying the constraints that describe the concept of fast for a computer). In refining such a softness, i.e., to make it operational, the analysts perform a two-step process:

1. First, they build a quality model [18, 19] of the soft (quality) attribute, by identifying the characteristics that the stakeholders assume to be important in judging it. For example, for judging a computer speed, we can identify as relevant computer characteristics the CPU clock, the memory size, the number of levels of the cache memory, the type of CPU architecture, and so on.
2. Then, they populate and freeze the quality model, by specifying for each characteristic the desired values (or range of values), according to the corresponding measurement scale [25]. In other words, for each characteristic, the analysts specify a constraint. So, for example, for the CPU clock (ratio scale) a range of real numbers can be used (e.g., CPU clock $>$ 0.6 GHz) for the CPU architecture (nominal scale) a specific type has to be specified (e.g., a RISC architecture) and for the number of level of the cache memory (absolute scale) a number or a range of possible numbers have to be assigned.

The quality model populated with the final result of the desired characteristics' is a comparable non-functional requirement. The relevant characteristics of the final system (e.g., the computer), indicated by the quality model, in fact, can be measured and compared against the corresponding specified values, to assess whether or not the system meets the non-functional requirement [2, 25, 26] (e.g., to be fast). Such a process is schematised in Fig. 9 with the support of the example soft goal 'buy a fast computer' discussed in the previous section.

To perform quality modelling, the analysts may turn to quality models already available in literature, such as the McCall [27], the Bohem [28], or the IEEE standard [29], or may adopt more sophisticated empirical measurement identification methods, such as the goal question metric (GQM) approach proposed by Basili [18] and its extensions [19].

Quality models available in literature suggest, for the high level quality attributes we are more likely to want to measure (e.g., usability, reliability, etc.), measurable characteristics by which they may be expressed. So, for example, the McCall factor criteria metric model first recognises some high level product quality factors (e.g., maintainability, efficiency), then for each factor identifies some lower level criteria (e.g., modularity) and then finally for each criterion proposes some actual metrics (e.g., module coupling). The analysts can use such models (as a whole or in part) as example templates or guidelines to refine the soft quality aspect of interest.

However, quality models show little flexibility and may, as a result, be difficult to customise to the specific context's and agents' needs. On the contrary, empirical measurement identification methods recognise that quality issues cannot live in isolation but have to be derived from, and linked to, their operational context (i.e., stakeholders, problem domains, underlying reasons, etc.)

The most classical of the empirical methods is the GQM approach, based on the idea that measures have to be identified starting out from the analysis of the goal that has to be achieved. The goal has to be precisely

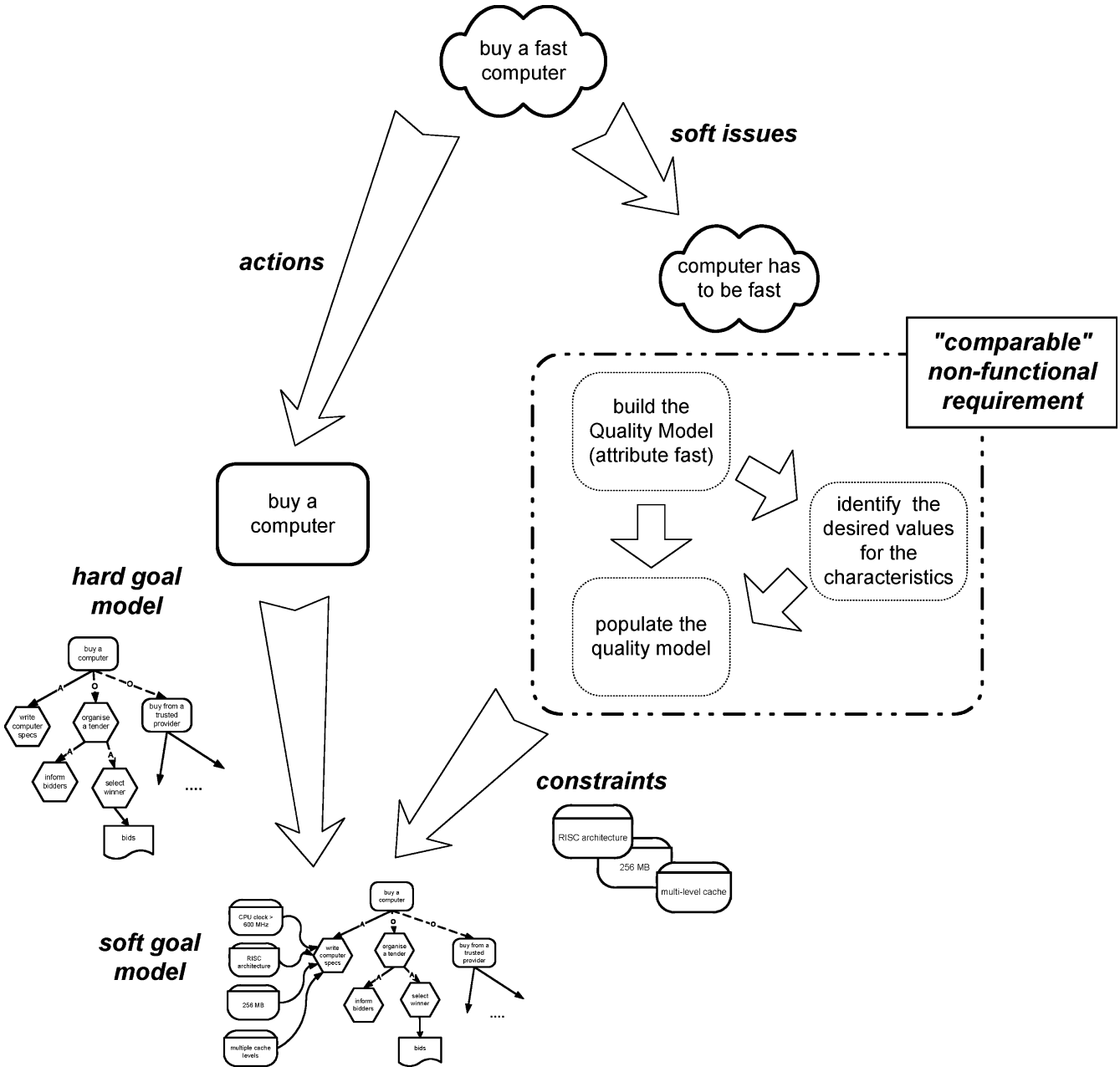


Fig. 9 Quality modelling in soft goal modelling

stated, by specifying the object of study, the purpose (the underlying reason, e.g., to understand, to improve, to monitor, etc.), the quality focus (the attribute of interest), the viewpoint and the context; then, it is refined into several questions that break down the issue into its major components, and then each question is refined into metrics. Basically, measures are obtained by applying a basic question-answer mechanism: asking which 'questions' we should be able to answer in order to achieve the 'goal', and then asking what 'metrics' we should collect to be able to answer those questions.

The GQM approach represents a very natural and at the same time pragmatic way of dealing with soft issues, providing both the analysts and the stakeholders with a simple tool for reasoning about soft goals. In such a perspective, a GQM-like approach has been applied for goal refinement during soft goal modelling. For each soft goal, in fact, REF clearly identifies the target object (the object of study), the reasons (the purpose), the soft attribute (the quality focus), the stakeholders (the viewpoint) and the application context (the context), providing the basis upon which a multi-step GQM-like question-answering mechanism has been adopted as an elicitation technique to gather and formalise knowledge from the agents. So, goal fuzziness has been reduced by repeatedly applying a question-answer approach, i.e., by repetitively asking the agents what they needed to know, to perform, to have delivered or to have performed in order to consider the soft goal as achieved.

So, for example, let us turn to the refinement of the soft goal shown in Fig. 4. To achieve the soft goal 'the computer has to be fast' (object of study: computer; purpose: has; quality attribute: fast; viewpoint: agent 'Tom'; context: business office), we could ask a question such as 'What characteristics affect the computer speed?', leading to the subordinate soft goals 'the computer must have high computational power' (object of study: computer; purpose: has; quality attribute: high computational power; viewpoint: agent 'Tom'; context: business office), and 'computer must have a good memory' (object of study: computer; purpose: has; quality attribute: good memory; viewpoint: agent 'Tom'; context: business office). From here, again, we could ask questions such as 'what kind of CPU is suitable for the applications run by Tom?', and 'what kind of memory structure is state-of-the-art for office automation?', from which metrics such as 'CPU clock', 'type of CPU architecture', 'memory size', and 'number of cache levels' could be identified as significant to formulate an answer.

4 The case study

SEs are complex software-intensive systems [8], which usually combine real equipments and real (human) operators with computer-based simulation models to provide greater flexibility and capability in supporting various aspects of a business enterprise. The classical application for SEs is training, where they offer potential cost and flexibility benefits. However, they have also been exploited in a number of other application areas: from equipment procurement (to inform the various life-cycle phases, from feasibility to disposal), to operational support, to policy and plan formulation.

SEs are becoming vitally important in a variety of industrial and government settings, as knowledge acquisition and decision support tools, leading to a growing awareness of the need to define and validate them adequately [8, 30]. It is crucial, in fact, that the RE process for a SE is able to ensure that the SE is suitable for its intended use. In other words, it is crucial that the type, the amount and the quality of the information provided by an SE reflect the specific needs of the business process it supports. Being so intertwined with its application context, an SE can be considered as a suitable test bed for the introduced REF.

In the following section, REF is applied to support the RE process for a hypothetical SE, needed to investigate the feasibility of providing an aircraft with an infrared pod: an infra-red camera which is normally used on aircraft and helicopters dedicated to 'search and rescue' and 'anti-fire' roles to improve the air-crew vision capability.

Although hypothetical, the case study is firmly based upon a real application, where SEs have been used for a similar purpose. The original project was concerned with a strict military application and it is referred to by various works [31, 32], where different aspects, from the system architecture and project complexity to the adopted software development approach, are described. The project lasted for 10 months over a calendar time of 15 months, involved about twelve persons, organised in specialised teams (i.e., avionics experts, software experts and crew representatives), required about 11 K line Ada code to be written and led to a full evaluation of the avionics system integration feasibility. In this context, the REF was applied to support the whole RE process for the required SE. In particular, REF eased interaction with the final system users (crew members and technicians), and helped to evaluate and validate as early as possible critical choices regarding functionalities and components (e.g., which components had to be simulated) characterised by a possible high impact, in terms of cost and time, over the whole project. Partly because of the original application domain, and partly because of the complexity of the target system, in this paper, we preferred to turn to a hypothetical application, suitable enough, however, to convey the REF capabilities.

4.1 Applying REF

In applying REF, the cyclical development flow described in Fig. 1 is adopted: starting with the initial organisation model (the start-up phase), the case study will evolve through the goal modelling and the organisation modelling phases.

4.1.1 Start-up phase (modelling the organisation)

The initial organisational context, or, in other words, the organisation model representing the initial problem is shown in Fig. 10.

As seen in Section 2, circles represent agents, whereas dotted lines are used to bound the internal structure of complex agents; that is, agents containing other agents. Consequently, in Fig. 10, the feasibility study is a complex agent, modelling the business process that has to investigate the feasibility of providing an aircraft with the infrared pod, whereas the project leader is a simple agent, acting within the feasibility

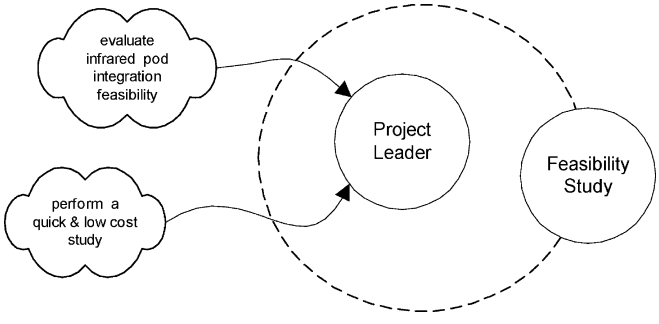


Fig. 10 The initial organisational model to perform the feasibility study

study. Agents interact by exchanging goals, tasks and resources. Thus, in Fig. 10, the project leader, which is the agent in charge of the feasibility study, receives, from the enclosing context (for example, the research and development unit, out of the scope of this analysis), the soft goals 'evaluate the infrared pod integration feasibility' and 'perform a quick and low cost study'. These goals can be seen as the result of the decision, made at a higher organisational level, of performing a feasibility study to gather more information before proceeding in the project.

4.1.2 Goal modelling phase

Once built, the initial model, to carry on with the analysis of the discovered goals, has to be refined through interaction with the involved agent, i.e., the project leader. Each agent, in fact, works as a goal transformer. Having received a goal, an agent will operate according to its own experience, knowledge or position within the organisation, in trying to achieve the goal. The agent 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. Such a transformation (i.e., the agent's behaviour) can be analysed through and captured by goal modelling. Figure 11 shows the 'evaluate infrared pod integration feasibility' soft goal model, representing how the Project Leader will act to achieve the corresponding goal.

In particular, Fig. 11 shows how the soft goal 'evaluate infrared pod integration feasibility' spawns three subordinate soft goals: 'evaluate integration risks', 'evaluate final integration time & cost', and 'evaluate technical feasibility'. These subordinate soft goals are further decomposed. The 'evaluate technical feasibility' soft goal leads to further subordinate soft goals such as 'evaluate avionics integration feasibility', and 'evaluate software integration feasibility'. The soft goal 'evaluate integration risks' leads to the soft goal 'evaluate compatibility with other projects', indicating the need of assessing this project in the wider contexts of all the activities aimed at improving the aircraft. Finally, the soft goal 'evaluate final integration time and cost' leads to the hard goals 'estimate final software integration time and cost' and 'estimate final avionics integration time and cost', together with the associated constraint, stating that a 20% margin of error (for a feasibility study) can be tolerated. As already described in a previous section, the emerging constraints are linked to the soft goals by a simple dependency link to express the fact that it depends upon them in order to be fulfilled. As a whole, the soft goal model shows how the abstract concept of 'feasibility' can be reidentified, making it relevant to the specific context, captured and formalised [22] (and possibly reusable in similar projects).

In Fig. 11, the bold outline has been used to highlight the items that the project leader will pass out, having decided to depend on other agents for their achievement. For such a reason, they are not further analysed; instead they will be refined as a further agreement between the project leader and the agent that will be appointed. These may be items for which the project leader thinks it would be better to get some support, or which may be out of the scope of his responsibility. So for example, the project leader can decide to ask an avionics expert for some support (e.g., to 'evaluate avionics integration feasibility'), to get help from a software expert (e.g., to 'evaluate software integration feasibility'), and to signal to the external context the necessity to 'evaluate the compatibility with other projects'.

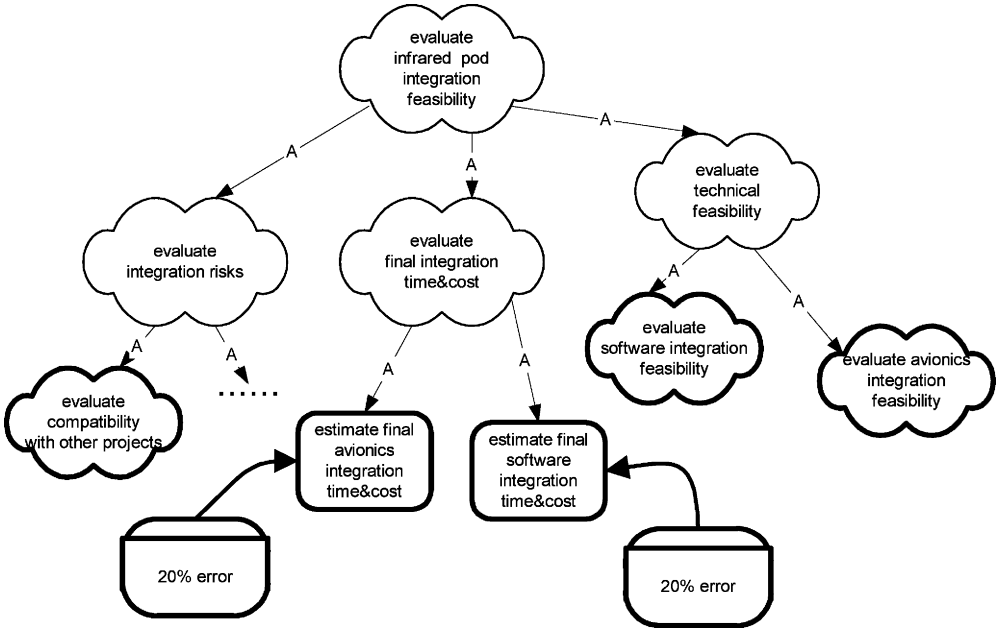


Fig. 11 The 'evaluate infrared pod integration feasibility' soft goal model

4.1.3 Organisational modelling phase

All the information acquired by modelling the goal may now be used to enrich and extend the initial organisation model. In other words, the results of the analysis of Fig. 11 allow us to enrich the organisation model in Fig. 10, leading to the model in Fig. 12. Here two new agents have been introduced: the avionics system expert and the airborne SW expert.

In particular, Fig. 12 shows how the project leader has decided to depend upon the avionics system expert for the soft goal 'evaluate avionics integration feasibility' and the hard goal 'estimate the final avionics integration time and cost' (with a constraint stating that a 20% margin of error can be tolerated), and has decided to depend on the airborne SW expert for the soft goal 'evaluate the software integration feasibility'. In addition, it is also shown how a new goal is emerging from the feasibility study, i.e., 'evaluate compatibility with other projects', placing new responsibilities upon the encompassing context.

4.1.4 Goal modelling phase

The organisation model in Fig. 12 shows that the avionics system expert will have to achieve the soft goal 'evaluate avionics integration feasibility' and the hard goal 'estimate final avionics integration time and cost', received by the project leader. At this point, the analysis can be carried on by analysing the behaviour of the avionics system expert: in other words, the analysts will have to deal with the avionics system expert to understand how she or he will try to achieve the received goals. The resulting goal models are shown in Figs. 13 and 14.

Figure 13 depicts the model of the soft goal 'evaluate avionics integration feasibility'. In order to achieve the received soft goal, the avionics system expert will need to 'observe the infrared pod in its avionics environment', and to 'evaluate the crew workload'. The first soft goal will spawn some precise goals. That is, the hard goal 'collocate the infrared pod in its avionics environment', with the associated soft goal 'high realism of the infrared pod avionics environment', and the hard goal 'monitor avionics system behaviour', leads, among others, to the task 'report data about avionics bus load'. To achieve the second one, instead, the avionics system expert will require the crew to provide a judgment in operative conditions, i.e., to achieve the soft goal 'crew judgment in operative conditions'. As in Fig. 11, and also in Fig. 13, the bold outline is used to highlight those goals and tasks that the avionics system expert will pass out, and they are not further refined at this stage. For example (see Fig. 15), the agent thinks that these goals can be accomplished only through the availability of a SE.

Figure 14, instead, shows the model of the hard goal 'estimate final avionics integration time and cost'. In order to achieve the received hard goal and the corresponding constraint '20% margin of error', the avionics

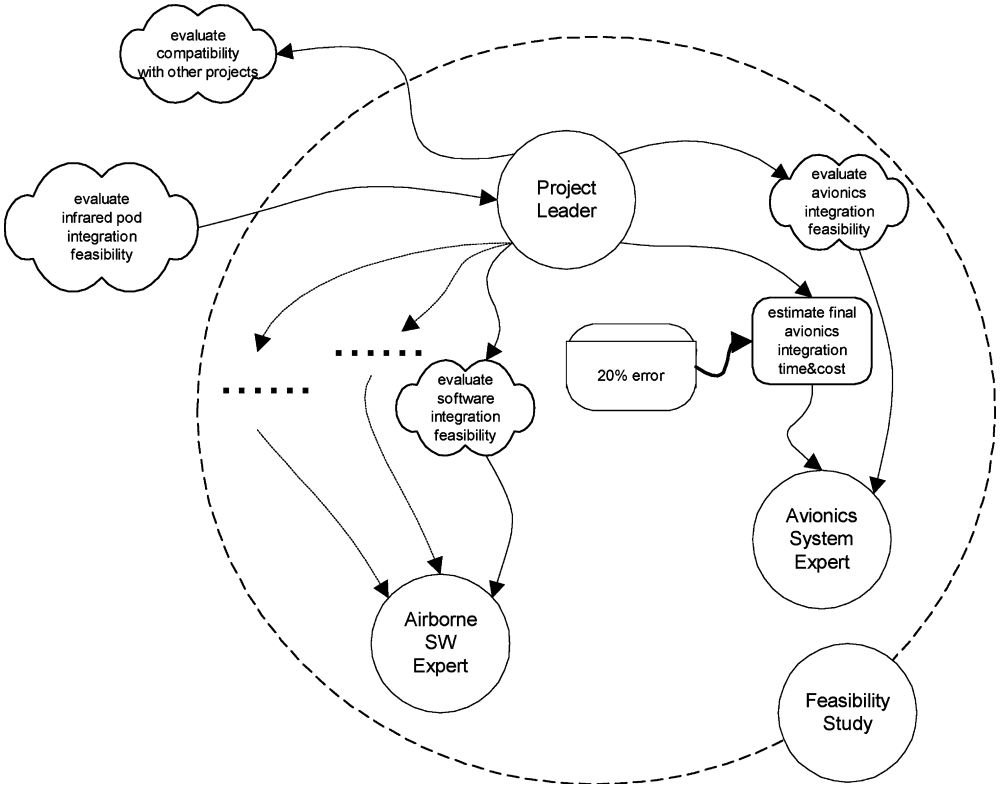


Fig. 12 The evolving organisational model to perform the feasibility study

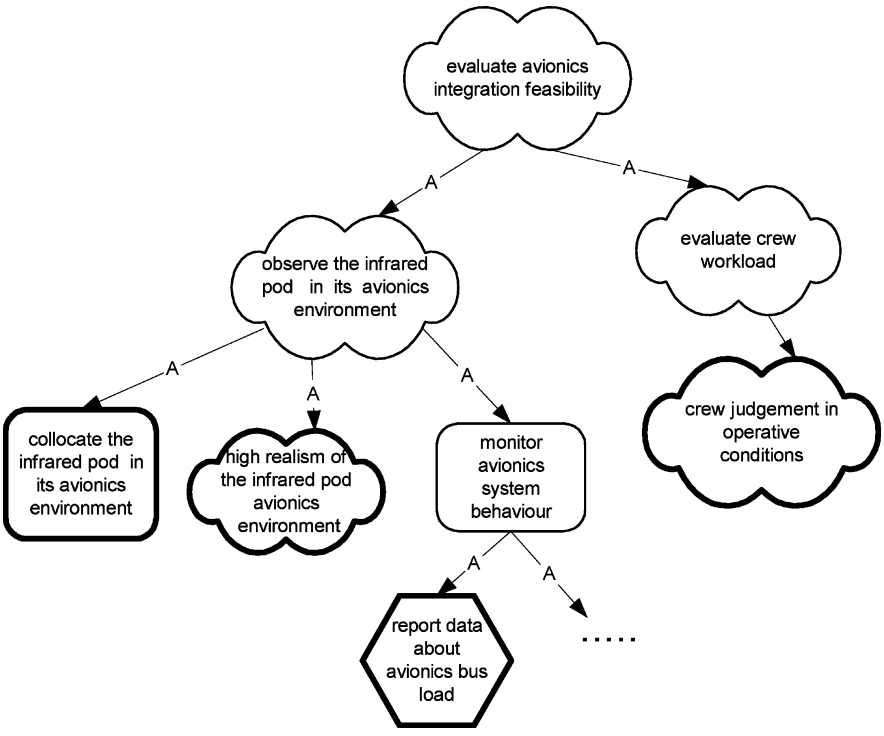


Fig. 13 The 'evaluate avionics integration feasibility' soft goal model

system expert will decide to forecast the time and cost for each project sub-phase, leading to three sub-tasks, i.e., forecast time and cost for the integration phase, for the on-ground certification phase and for the in-flight certification phase. In order to perform the first two tasks, the avionics system expert may decide to rely on data coming from similar projects (and in this way to avoid asking support from other agents), or asking another agent (for example, the production unit) to provide an estimation. To estimate the in-flight certification time and cost, instead, the avionics system expert decides it needs support from another agent (for example, the flight test unit), to which it has to provide the plan for the necessary flights (express as a resource).

In Fig. 14, it is also shown how the constraint placed upon the initial goal will affect the subordinate tasks and

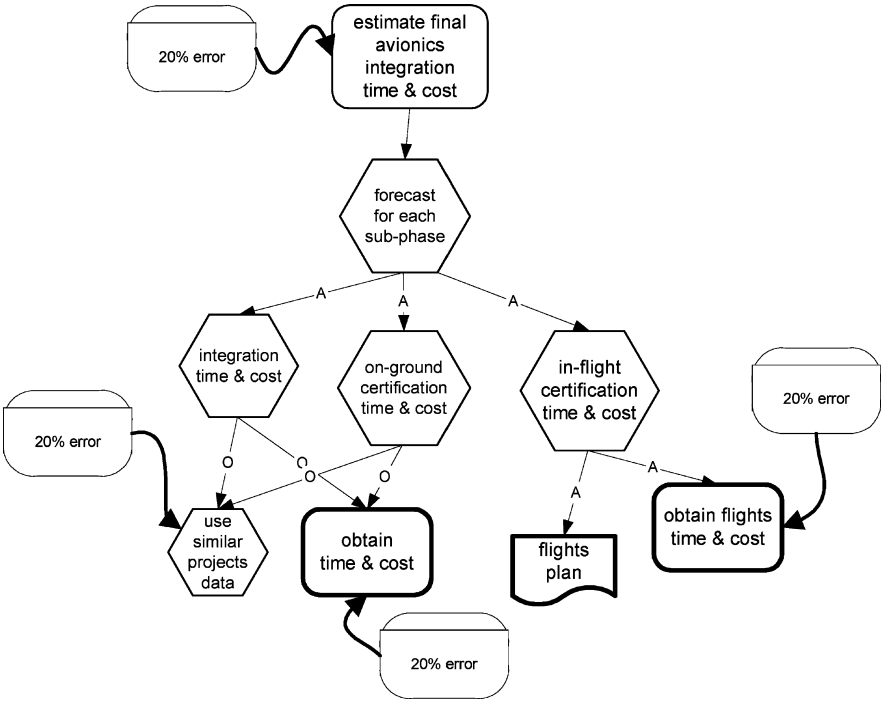


Fig. 14 The 'estimate final avionics integration time and cost' hard goal model

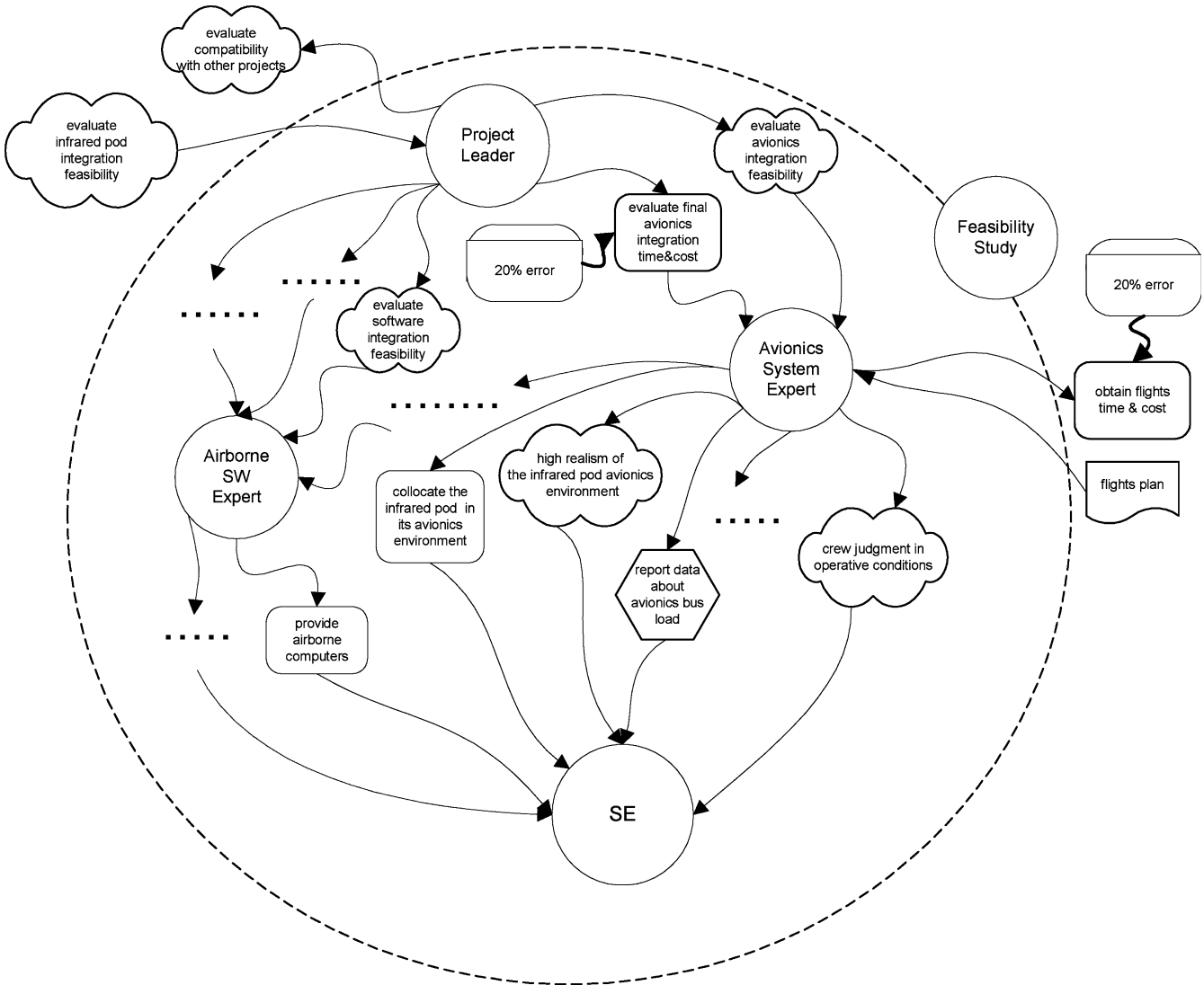


Fig. 15 The evolving organisational model to perform the feasibility study

goals. Again, as in the previous goals models, the items in bold outline are those that the agent will pass out.

4.1.5 Organisation modelling phase

The results of the analysis in Figs. 13 and 14 (and of a similar one performed for the airborne SW Expert) allow the analysts to further extend the organisation model, leading to the one illustrated in Fig. 15.

In Fig. 15 a new agent, SE, has been added, and it is shown how the project leader, the avionics system expert and the airborne software expert interact with each other and with the new agent to collect the information necessary to assess the feasibility of equipping the aircraft with the infrared pod. In addition, Fig. 15 also shows how the avionics system expert has decided to ask for support from an external agent (for example, the flight test unit) to estimate the in-flight certification time and cost, for which it will provide the necessary flights plan. The avionics system expert has instead decided to use similar project data (see Fig. 14) to estimate integration and on-ground certification time and cost.

4.1.6 Goal modelling phase

At this point of the analysis, the focus turns upon the SE, and again, the analysis has to start from the goals and the tasks placed upon it. However, in this case, it is evident that some initial decisions about its structure can be made: the soft goal 'crew judgment in operative conditions', in fact, clearly suggests that a new agent, i.e., a crew, will have to be taken into account. It is typical of SEs 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 crew is part of the overall SE, so that the effects of the infrared pod integration on crew performance can be determined, but it is the project leader who is the primary owner of the process (i.e., the feasibility study).

4.1.7 Organisational modelling phase

From this initial assessment of the goals placed upon the SE, it is possible to observe that the SE may be modelled as a complex agent, encompassing other two simple agents, the flight test crew and the avionics system rig (i.e., a particular kind of ground-based equipment, capable of simulating the characteristics of the aircraft and of its components). In general, a complex agent would be further refined into a subordinate organisation model, suitable to satisfy all the goals placed upon it, or, in other words, to enact the corresponding goals models. The resulting organisation model is shown in Fig. 16.

4.1.8 Goal modelling phase

We can now focus on the soft goal 'crew judgment in operative conditions', placed upon the flight test crew. In particular, in order to be able to express its opinion, the flight test crew will require the possibility of 'flying' the new pod, which places two other goals on the avionics system rig: a hard goal, 'try the pod in flight conditions', and a soft goal, 'flight condition realism'. The resulting soft goal model is shown in Fig. 17, where the goals that flight test crew will pass out are marked with bold outlines.

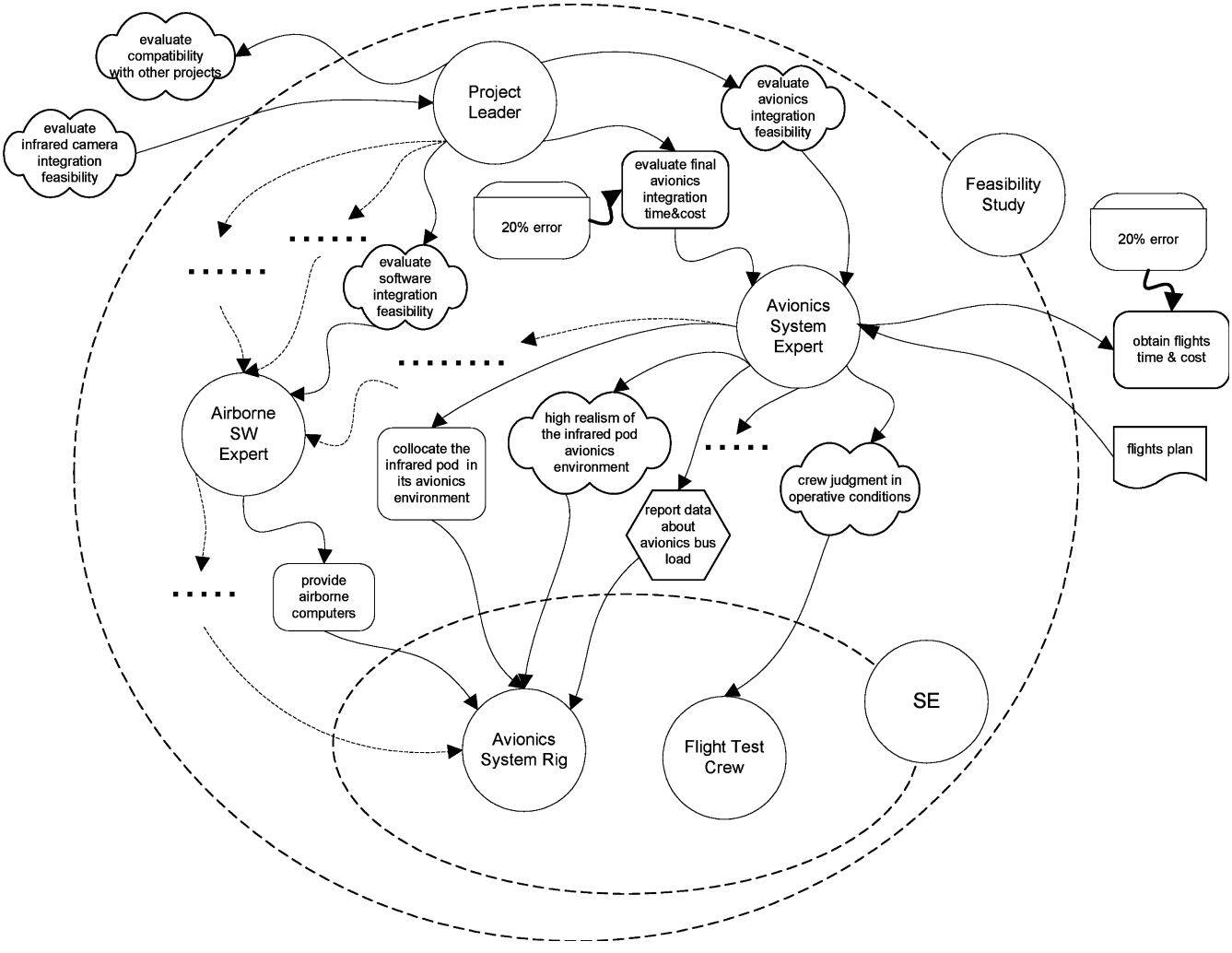
4.1.9 Organisation modelling phase

Again, the results of the goal modelling phase can be used to enrich the organisation model. By adding the goals generated by the flight test crew, the organisation model of Fig. 18 can be obtained.

4.1.10 Goal modelling phase

The final target of our analysis is the avionics system rig, so the analysis may be carried out by modelling the goals placed upon it by the other agents. Let us take into account the soft goals 'high realism of the infrared pod avionics environment' and 'flight conditions realism', placed upon the avionics system rig by the avionics

Fig. 16 The evolving organisational model to perform the feasibility study



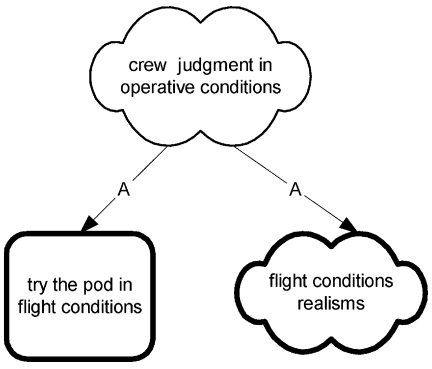


Fig. 17 The 'crew judgment in operative conditions' soft goal

system expert and the flight test crew, respectively. These two goals, in particular, provide the occasion to show how soft goals can act as a useful tool to detect and resolve clashing requirements, as discussed in the following section.

Fig. 18 The evolving organisational model to perform the feasibility study

4.2 Soft goals as a mean to detect and resolve clashing requirements

During soft goal modelling the analysts and the stakeholders tend to clarify very early in the project concepts that are usually left blurred until implementation forces them to make a choice. Soft goals become a knowledge representation vehicle that:

1. encourages the interaction between the analysts and the stakeholders, and among the stakeholders themselves
2. leads towards a common terminology
3. supports reasoning about tradeoffs
4. allows the freezing of temporary solutions, and the formalising of final decisions

Soft goal models therefore allow the analysts to detect clashing requirements early, which usually are hidden behind generic and left-implicit assumptions, and, at the same time, provide an operational way to resolve them, by reconciling the different stakeholders' points of view. As an example of this capability, let us proceed with the analysis of the case study by modelling the soft goals 'high realism of the infrared pod avionics envi-

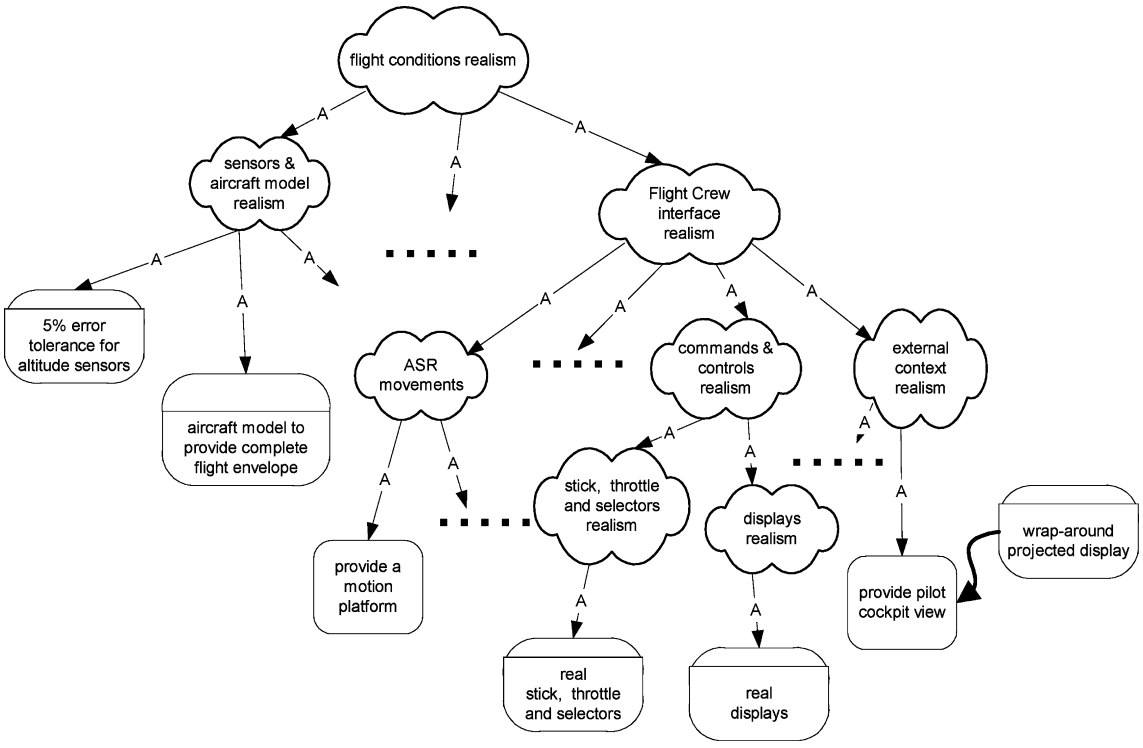


Fig. 19 'Flight conditions realism' for the Flight Test Crew

ronment' and 'flight conditions realism', placed upon the avionics system rig by the avionics system expert and flight test crew, respectively.

In order to model the soft goal 'flight conditions realism' placed upon the avionics system rig by the flight test crew, we have to understand what the flight test crew means by 'flight conditions realism', and to see if this is compatible with the view of the other agents, for example

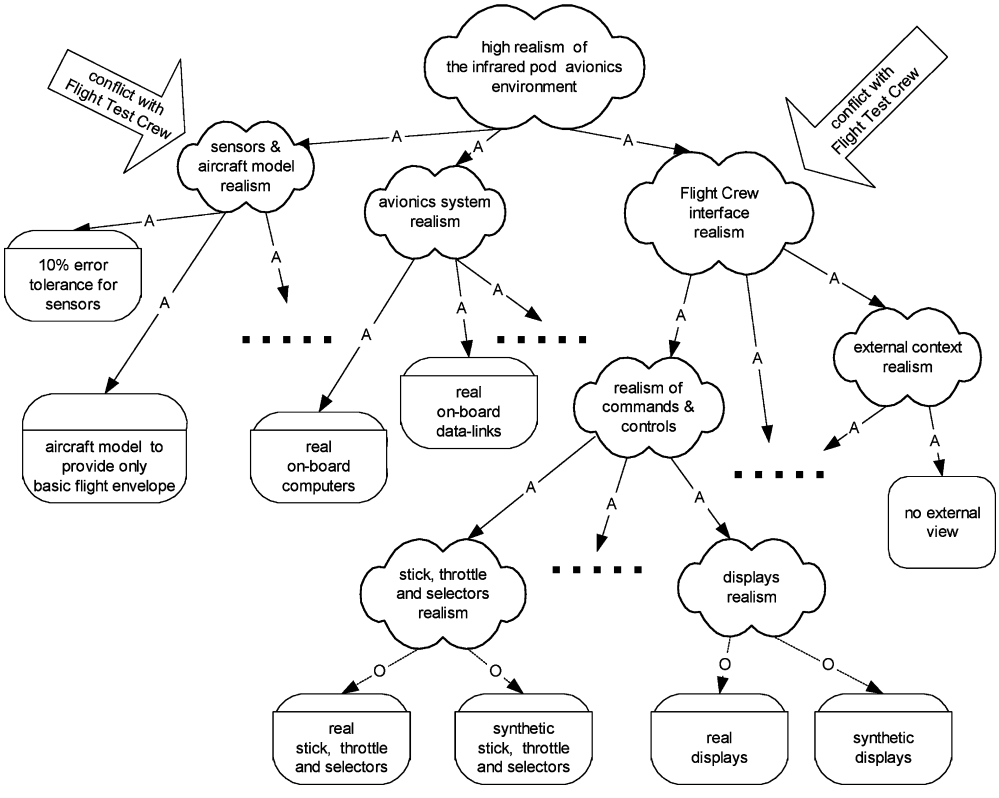


Fig. 20 'High realism of the infrared pod avionics environment' for the avionics system expert

the avionics system expert. The answer may be found by analysing the soft goal models in Figs. 19 and 20.

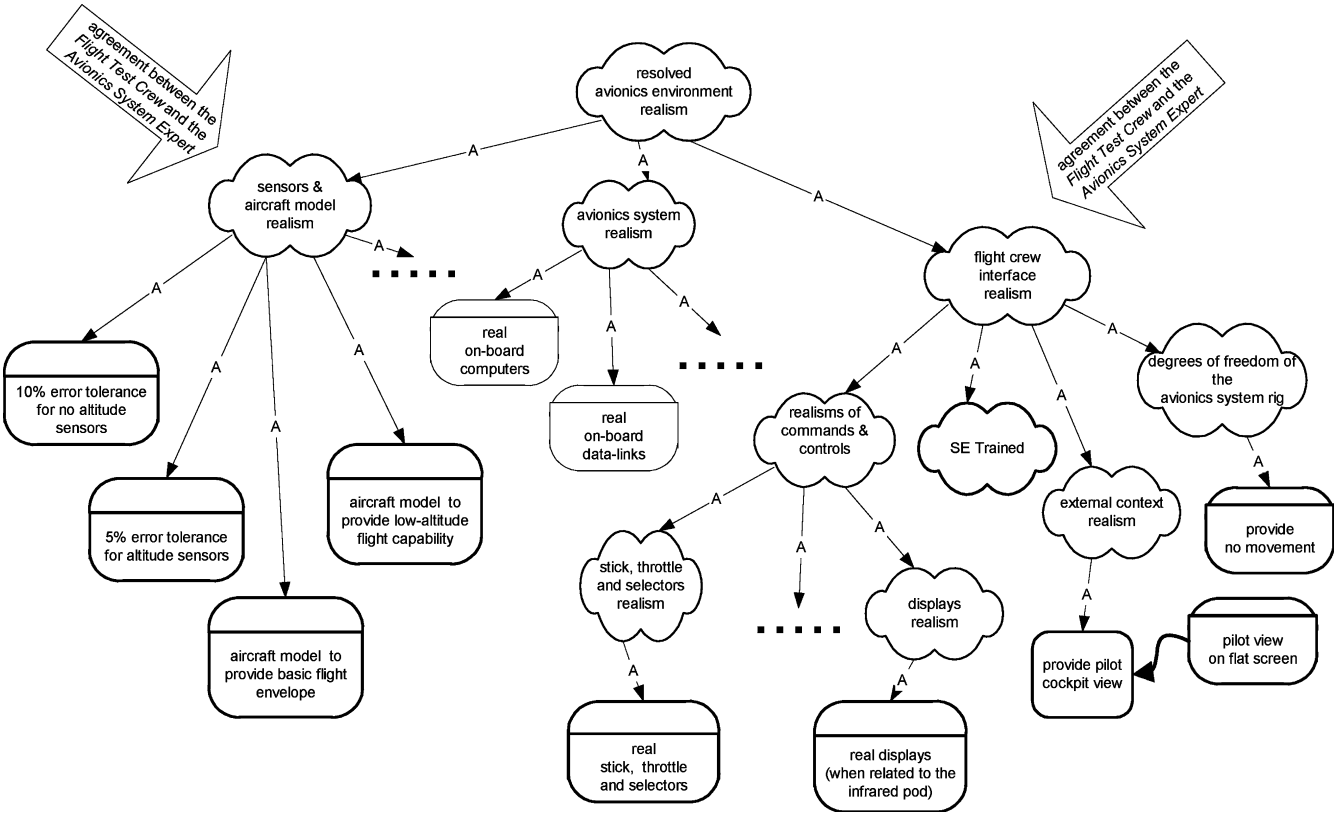
Figure 19 illustrates what the flight test crew means when it requires realistic flight conditions. Here, for the sake of the example, only two of the possible related aspects (subordinate soft-goals) are taken into account: the realism of the adopted sensors and aircraft simulation models, and the realism of the flight crew interface. For the first point, the flight test crew requires being able to fly the complete aircraft flight envelope, and to employ quite precise altitude sensors, being aware of the relevance of the new system in low-altitude missions. For the second point, an extensive use of real equipment is required. For example, the crew will require the avionics system rig to have 'real stick, throttle and selectors' and 'real displays', to have a 'wrap-around projected display' to show the external environment and to have a motion platform.

Figure 20, instead, illustrates what the avionics system expert means when he/she requires a realistic infrared pod avionics environment. In this example, only three subordinate sub-goals are analysed. One, the 'avionics systems realism', is a matter of concern to only the avionics system expert, so no conflicts with the flight test crew arise. The other two, instead, lead to problems, as highlighted in both the figures by bold arrows. In particular, the avionics system expert does not assume the availability of a complete flight envelope as relevant for the project, and thinks that a 10% margin of error for the sensors is more than sufficient. Furthermore, he or she is open to various implementation solutions for what concerns the crew interface, at different levels of cost and complexity. For example, for the avionics system expert, there is no difference in using a real stick or a synthetic one (e.g., a computer joystick).

In this situation, it is therefore clear how soft goals models not only help in identifying possible conflicts, but also provide a pragmatic way of reasoning about tradeoffs, and formalising results. The 'resolved avionics environment realism' soft goal model is illustrated in Fig. 21, where the two bold arrows indicate where the agreement between the avionics system expert and the flight test crew has been reached.

Thus, for example, for what concerns the crew interface, we can assume that, due to cost limitations (see the initial 'perform a quick and lost cost study' soft goal of Fig. 10) an interface less complex than the one required by the flight test crew would be implemented: to employ real displays only when they are strictly related with the use of the infrared pod, to use a flat screen (rather than a wrap-around projected one) to show the external environment and to avoid proving a motion platform. This choice, although saving on the equipment, would lead to the need of employing only crews specially trained in the use of SEs. Such a need is captured by the subordinate soft goal 'SE trained' that will be placed upon the flight test crew (see Fig. 22). By further refining this goal, specific constraints and sub-ordinate goals defining the necessary crew skill can be obtained.

Fig. 21 The resolved 'avionics system rig realism' soft goal model



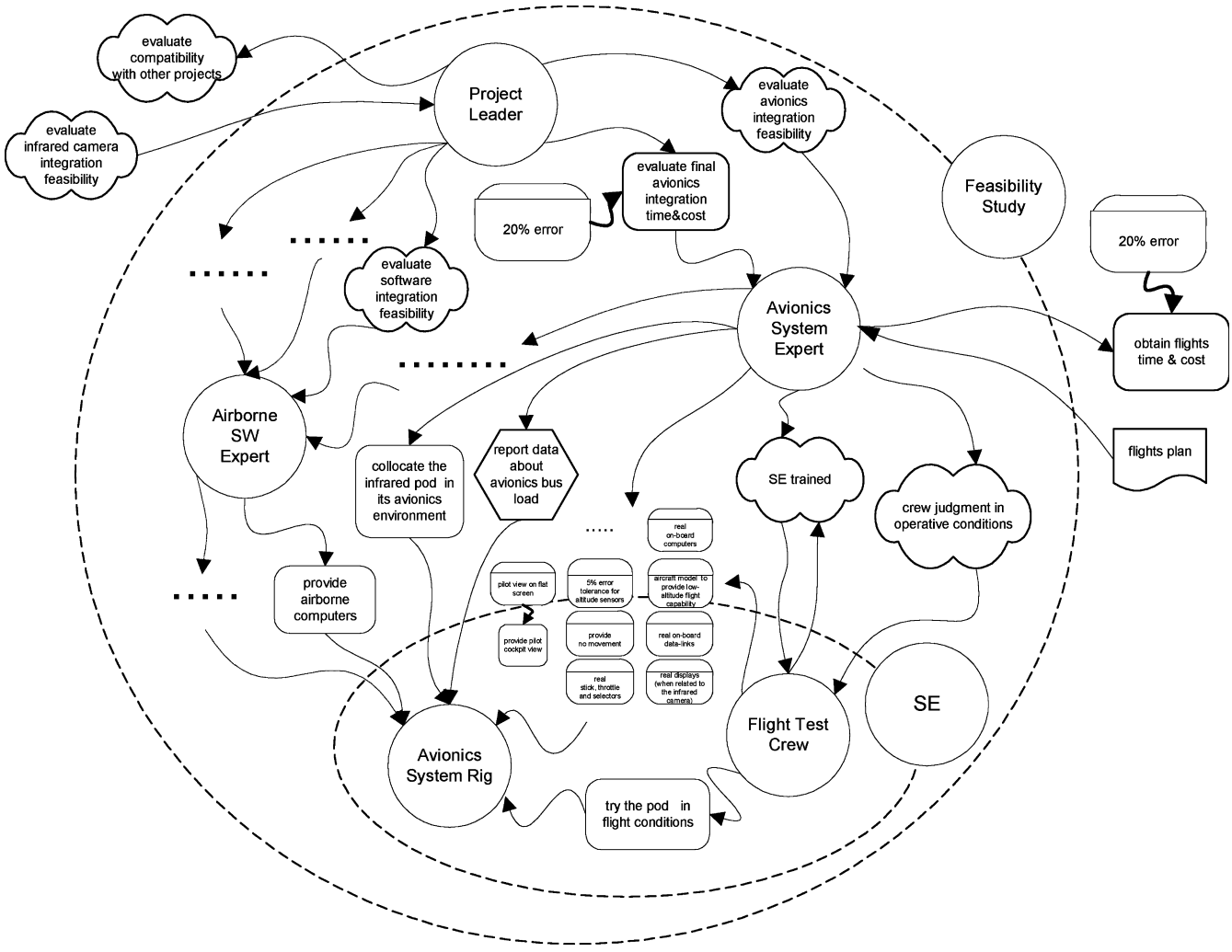


Fig. 22 The 'final' organisation model to perform the feasibility study

For what concerns the realism of the sensors and of the aircraft model, instead, the importance of performing low-altitude flights expressed by the flight test crew has been recognised by the avionics system expert, yielding, for example, to a stricter constraint for altitude sensors (with a 5% margin of error tolerance), and to simulate a low-altitude flight capability. Such an example shows how soft goals, and in particular, soft goal modelling, allows the analysts to clarify very early in the RE process what is really needed (i.e., meant) by the stakeholders, so as to avoid late and costly conflict resolution.

4.2.1 Organisational modelling phase

The information obtained by modelling the last two soft goals allows a further extension of the organisation model. In particular, the results of the analysis in Fig. 21 lead to the organisation model in Fig. 22. Here, for the sake of simplicity, all the constraints emerging from Fig. 21, and placed upon the avionics system rig by the avionics system expert and the flight test crew, have been schematically represented together as the hard goal 'provide pilot cockpit view' and its associated constraint 'pilot view on flat screen'. For what concerns the soft goal 'SE trained', it is worth noticing that, as it is shown in Fig. 22, it is placed upon the flight test crew and originated by both the avionics system expert and the flight test crew itself.

Figure 22 can be considered as the last step of our analysis. Although simplified, for the sake of clarity, Fig. 22 shows in fact the operational context within which the final system (the avionics system rig) and the other agents act (and interact) in order to achieve the set high-level organisational goals. In particular, it highlights the hard goals, the tasks and the constraints placed upon the avionics system rig by other agents. Collectively, these hard goals, tasks and constraints form the initial set of requirements for the avionics system rig.

At this point, the analysts will have to decide whether to carry on requirements refinement with REF or to adopt a different (and maybe more suitable) technique, as discussed in the next section.

5 REF as a forerunner of further system development

REF allows the analysts to model the application context at a social-technical level, by providing a systematic approach to deal with agents, soft goals, hard goals and their incremental refinement. Starting with the high-level organisational goals, the main requirements for the target software system (e.g., the avionics system rig) can be obtained and expressed as a collection of hard goals, tasks and constraints placed upon it by other agents of the organisation.

As an example, Fig. 23 illustrates some of the hard goals, tasks and constraints found during the case study of the previous section. In particular, Fig. 23 shows the hard goals, the tasks and the constraints placed upon the avionics system rig by the avionics system expert and the flight test crew.

Once at this point, the analysts have to deal with a purely technical system. Although it would be possible to carry forward the REF social-technical modelling approach within it, which would eventually result in a network of agents performing a set of tasks, the possibility of incorporating techniques more suitable for dealing with the final, and purely technical, system requirements has been investigated.

Initial results show, in fact, that REF can be usefully applied as a forerunner to UML-based [20] approaches. REF output, in fact, as a set of hard goals and constraints placed upon the target software system by well-defined agents not only is entirely consistent with 'use case modelling' (i.e., the front-end activity of any UML-based approach), but also addresses the need, clearly recognised by both the research community and the business world, that use case analysis can benefit substantially from supporting methods that help to identify both the actors, which interact with the system, and their motivating goals [33].

Although details can be found in [10, 30], two main points about the suggested approach are worthy of being noticed here. Firstly, UML use cases offer a systematic and intuitive means of capturing functional requirements, so UML use case modelling results in a very effective hard goals refining strategy [30] to translate hard goals placed upon the system into functionalities it has to provide. A hard goal can in fact lead to one or more use cases, where a use case is a collection of possible sequences of interactions (scenarios) between the actor and the system. Importantly, the possibility of delaying hard goal refinement at the REF analysis stage, through use cases at this level, helps to reduce the detail that must be captured at the social-technical level,

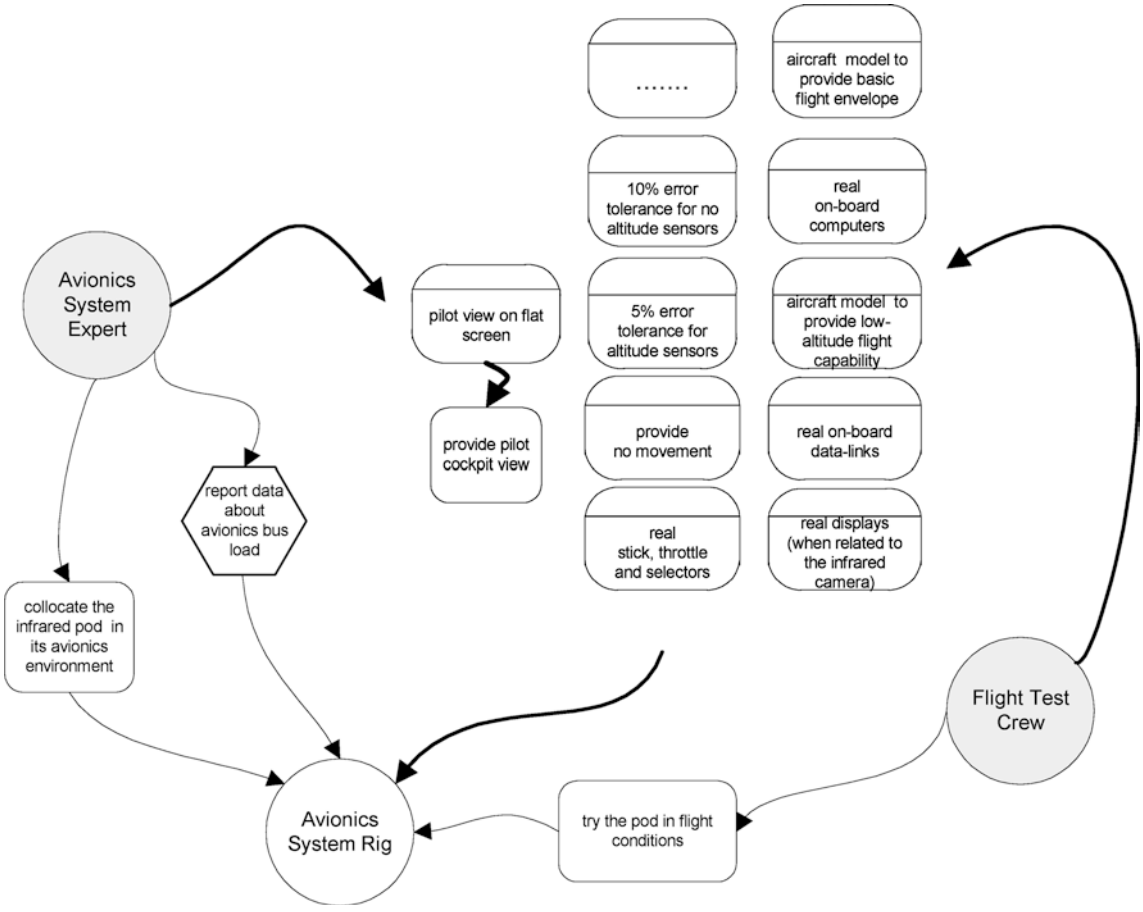


Fig. 23 The RE framework outcome: a set of hard goals, tasks and constraints

thereby providing greater clarity and abstraction. Secondly, the constraints emerging from the framework allow the enrichment and completion of use cases, by providing not only a systematic and intuitive means of capturing and formalising non-functional requirements, but also providing a very effective means for combining them with functional requirements. Enriched with constraints and clearly linked to goals and their originators, UML use case models result therefore in conceptual models suitable to combine functional and non-functional system aspects, and to better support subsequent development stages.

Some extracts from the avionics system rig use case diagram, developed by using the SELECT CASE tool, are illustrated in Fig. 24, where only the avionics system expert agent and his or her related goals and constraints have been taken into account. Here, by superimposing the REF notation (dotted lines) onto the UML notation, we can illustrate how use cases are related to hard goals and constraints. In particular, dotted boxes are used to group together use cases contributing to the same hard goal (linked by a 'spark'); dotted arrows are used to link constraints to use cases within which they should be dealt with, and tasks are simply put close to the use cases that will take their place.

Firstly, Fig. 24 shows how the whole set of use cases 'connect pod to the avionics sub-systems', 'perform pod on-ground operations' and 'perform pod in-flight operations' contribute to the achievement of the avionics system expert hard goal 'collocate the pod in its avionics environment'. Then, how the use case 'perform pod in-flight operations', on its own, aims to achieve the hard goal 'provide pilot cockpit view', and how the use case 'collect data' is simply another view of the task 'report data about avionics bus load'. Finally, Fig. 22 illustrates how the constraints are associated with the various use cases. For example, the constraints 'real on-board computers' and 'real on-board data links' will affect the way in which the use case 'connect pod to avionics system' will be performed.

Use cases may be described in a tabular form [20]. For example, Table 1 gives the descriptions for one of the use cases introduced in Fig. 24. The rows with a dark background contain the basic use case information; that is, the use case name, the actor, the description and the specific intent behind it. Here, for the sake of brevity, only very short descriptions have been given. The other rows represent additional information, including a

Fig. 24 The use case diagram for the avionics system expert

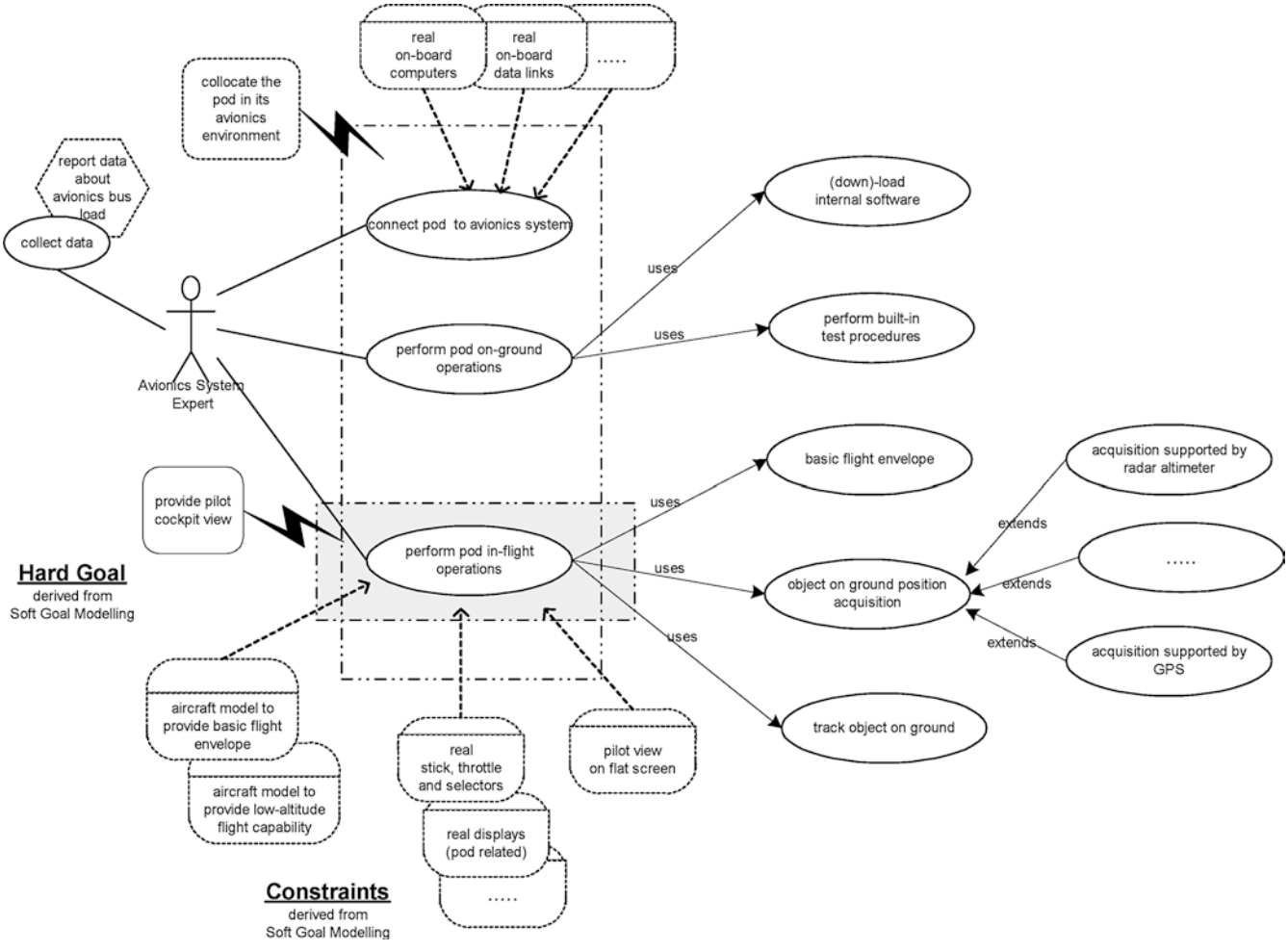


Table 1 Tabular description of the use case 'connect pod to avionics system'

Use case feature	Feature description	
Name	V	
Actor	Avionics system expert	
Description	-	Connect power cable
	-	Connect discrete signal links
	-	Connect bus data link
Intent	To connect the pod to the avionics system	
High-level Goal	Collocate the pod in its avionics environment	
Pre-condition	Pod available	
Post-condition	Pod connected to avionics system	
Non-functional requirements	-	ASR to employ real on-board computers
	-	ASR to employ real data/power cables
	-	

specification of the hard goal that the use case contributes towards achieving, and the non-functional requirements derived from the constraints.

The further system development according to UML is outside the scope of this work. However more details on how UML can be used as a modelling technique to produce a conceptual (analysis) model of the avionics system rig are in [10,30].

6 Conclusions

The paper has introduced a goal-oriented and agent-based REF explicitly designed to support reasoning about socio-technical systems and to enable the analysts to transform high-level organisational needs into system requirements, while re-engineering the organisational structure itself.

REF adopts requirements engineering concepts, such as those of agent, goal and intentional dependency that have been recognised as crucial for a homogeneous and natural requirements engineering process, smoothly moving from organisational analysis to system requirements. In addition, REF encompasses quality modelling techniques in order to capture the stakeholders' perception of quality as early as possible, and to produce agreed-upon and implementable functionalities and constraints.

REF is strongly based on diagrammatic notations which immediately convey the dependencies among different agents and allow for a detailed analysis of the goals upon which the actors depend, through a top-down goal decomposition process. The adopted graphical notation is widely inspired by the i* framework. However REF introduces also a clear methodology to drive the process of requirement discovery, definition, refinement and reconciliation, and adopts the i* notation at a very essential level; this choice, although apparently constraining, results in quite a success in practical terms. Several case studies [34, 35, 36] demonstrate, in fact, that the simplified notation facilitates the acceptance of the diagrammatic language by the stakeholders, and contributes to a quicker introduction of the methodology in the RE process. For example, in [36], it has been applied to define the requirements for a workflow-based [37] document management system, whereas in [35] it has been adopted to analyse the organisational impact and advantages of introducing a corporate smart card as an enabling platform for accessing and using different e-services delivered by the organisational information technology system (e.g., the digital signature, certified e-mail, document management systems, etc.).

The underlying concepts and the adopted notations make REF a very effective and usable tool, able to tackle complex real case situations, while remaining simple enough to allow a concrete and effective stakeholder involvement. The availability of simple representational tools during RE is a key factor to guarantee the involvement of stakeholders, facilitating their understanding and participation.

The described application example demonstrates the feasibility of the suggested approach, and the benefits it offers during the early phases of the requirements engineering process, when the analysts and the stakeholders have to cooperate to understand and reason about the organisational context within which the new system has to function, in order to identify and formalise not only the system requirements, but also the organisational setting that better exploits the new system's capabilities.

REF benefits can be observed in terms of requirements discovery and validation, mainly because of the visibility of decisions made by the stakeholders, resulting from explicit organisation and goal modelling. Each part of the framework model provides, in fact, a specific knowledge representation vehicle that the analyst can use to interact with the stakeholders, in order to capture requirements, reason about them and eventually reach an accepted formulation. In other words, soft goal models force the stakeholders to reason about their own concepts of quality (for example, the concept of 'realism' as shown in Fig. 21) and hard goal models allow the stakeholders to understand and validate their role within the organisation, whereas organisation models provide the management with a clear view of how the business

process will be affected by the introduction of the new system (see Fig. 18).

Once the project is over and the system deployed, the knowledge acquired during the RE process does not extinguish its role. On the contrary, made easily accessible and reusable through the different models into which it has been captured, such a knowledge offers potential benefits also in the post-deployment phases, supporting system maintenance and evolution. The clear links established (through the different models) between organisational goals and system requirements, in fact, can provide support to the analysts and the stakeholders in identifying the effects that changes in the organisational goals or new technology trends may have upon the system requirements. The possibilities offered within our context by a new display technology, for example, could be easily evaluated by observing the corresponding goal models (i.e., Figs. 19, 20 and 21): the new technology will be judged valuable for the system (and the organisation) when it is able to overcome some of the limits found during the previous analysis, or, in other words, to enable the stakeholders to better achieve their goals. At the same way, a change in an organisational goal may easily be translated into requirements changes.

The framework models also improve and support knowledge transfer and sharing across different projects [22]. In particular, fragments of knowledge, pieces of information or system components may be reused in different application contexts. For example, the outcome of the framework, i.e., the requirements expressed as refinements of high-level goals, can be easily generalised and exported to different projects, contributing to improve the sharing and reuse of the knowledge captured, produced and formalised during the very early stages of the system development.

Finally, REF can be usefully applied as a forerunner to UML-based approaches. As illustrated through the case study, the REF outcome does not only encompass the basic information upon which to start a UML analysis, but it is suitable to enrich and complete such information by providing a clear picture of the system's application context and the reasoning tools to represent and deal with non-functional requirements while operating with functional ones.

References

1. van Lamsweerde A (2000) Requirements engineering in the year 00: a research serspective. In: Proceedings of the International Conference on Software Engineering (ICSE), Limerick, Ireland, 4-11 June, 2000
2. Loucopulos P, Karakostas V (1995) System requirements engineering. McGraw-Hill, London
3. Mylopolous J, Castro J (2000) Tropos: a framework for requirements-driven software development. In: Brinkkemper J, Solvberg A (eds) Information system engineering: state of the art and research themes, Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York
4. Fickas S, Helm BR (1992) knowledge representation and reasoning in the design of composite systems. IEEE Trans Soft Engin 18(6):470–482
5. Yu E (1993) Modeling organizations for information systems requirements engineering. In: Proceedings of the 1st IEEE International Symposium on Requirements Engineering, San Diego, CA, January, 1993
6. Yu, E. Towards modeling and reasoning support for early-phase requirements engineering. In: Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE' 97), Washington D.C., 6–8 January, 1997
7. Kirikova M, Bubenko JA (1994) Software requirements acquisition through enterprise modeling. In: Proceedings of the Software Engineering Knowledge Engineering (SEKE) Conference, Jurmala, Latvia, 21-24 June, 1994
8. Donzelli P, Moulding MR (1999) Developments in application domain modeling for the verification and validation of synthetic environments: a formal requirements engineering framework. In: Proceedings of the Spring '99 Simulation Interoperability Workshop, Orlando, FL, March, 1999
9. Bresciani P, Perini A, Giorgini P, Giunchiglia F and Mylopoulos J (2001) A knowledge level software engineering methodology for agent oriented programming. In: Proceedings of the 5th International Conference on Autonomous Agents, Montreal, Canada, May 2001
10. Donzelli P, Moulding MR (1999) A unified approach to the verification, validation and accreditation of synthetic environments: a requirements engineering framework. Cranfield University Technical Report SE027E/TR1 Version 1, December, 1999
11. Donzelli P, (2002) Agents, goals and quality in a structured requirements engineering framework—a case study. In: Proceedings of the Agent-Oriented Information System Workshop, 14th Conference on Advanced Information Systems Engineering (CAISE), Toronto, Canada, 27–31 May, 2002
12. Donzelli P, Rus I and Cantone G Integrating quality modeling with requirements engineering. In: Proceedings of the European Software Control and Metrics Conference (ESCOM 2001), London, UK, April, 2001
13. Yu, E. Why agent-oriented requirements engineering. In: Proceedings of the 3rd Workshop on Requirements Engineering For Software Quality, Barcelona, Spain, June, 1997
14. D'Inverno M, Luck M (1997) 'Development and application of an agent based framework. In: Proceedings of the First IEEE International Conference on Formal Engineering Methods, Hiroshima, Japan, 12–14 November, 1997
15. Giorgini P, Perini A, Mylopoulos J, Giunchiglia F and Bresciani P (2001) Agent-oriented software development: a case study. In: Proceedings of the Thirteenth International Conference on Software Engineering and Knowledge Engineering (SEKE), Buenos Aires, Argentina, 13-15 June, 2001
16. Yu E, Mylopoulos J (1998) Why goal-oriented requirements engineering. In: Proceedings of the 4th International Workshop on Requirements Engineering: Foundations of Software Quality, 8–9 June, 1998, Pisa, Italy
17. Dardenne A, Van Lamsweerde A and Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program20(1-2)3–50
18. Basili VR, Caldiera G and Rombach HD (1994) 'The goal question metric approach. In: Marciniak JJ (ed) Encyclopedia of software engineering, Wiley, New York
19. Cantone G, Donzelli P (2000) Production and maintenance of goal-oriented software measurement models. Int J Soft Engin Knowl Engin 10(5):605–626
20. Rumbaugh J, Jacobson Iand Booch G The unified modeling language reference manual. Rational Software Corporation, Addison Wesley, London
21. Hammer M, Champy J (1995) Reengineering the corporation: a manifesto for business revolution. Brealey, London
22. Donzelli P, Setola R (2002) Handling the knowledge acquired during the requirements engineering process. In: Proceedings of the Fourteenth International Conference on Knowledge Engi-

neering and Software Engineering (SEKE), Ischia, Italy, 15-19 July, 2002
23. Yu E (1995) Modeling strategic relationships for process reengineering. Dissertation, University of Toronto
24. Yu E, Mylopoulos J (1996) Using goals, rules, and methods to support reasoning in business process reengineering. Int J Intellig Sys Acct Finan Manage 5(1):1–13
25. Chung L, Nixon B, Yu E and Mylopoulos J (2000) Non-functional requirements in software engineering. Kluwer, Norwell, MA
26. Fenton NE, Pleeger SH (1997) Software metrics: a rigorous and practical approach—second edition. International Thomson Computer Press, London
27. McCall J, Richards P and Walters G (1977) Factors in software quality. Technical Report 77CIS02, Rome Air Development Center
28. Boehm BW, Brown JR, Kaspar JR et al. (1978) Characteristics of software quality. North Holland, Amsterdam
29. IEEE Std 1061–1992, IEEE Standard for a Software Quality Metrics Methodology
30. Donzelli P, Moulding MR (2000) Application domain Modeling for the verification and validation of synthetic environments: from requirements engineering to conceptual modeling. In: Proceedings of the Spring 2000 Simulation Interoperability Workshop, Orlando, FL, March, 2000
31. Donzelli P, Marozza R (1999) Laser designation pod on the Italian Air Force AMX aircraft: a prototype integration. In: Proceedings of the NATO/RTO SCI Joint Symposium on Advances in Vehicle Systems Concepts and Integration, Ankara, Turkey, April, 1999
32. Donzelli P (2002) Tailoring the software maintenance process to better support complex system evolution projects. J Soft Maint Evol Res Pract 14:1–14
33. Cockburn A (1997) Structuring use cases with goals. JOOP 10(5):35–40 and 10(7):56–62
34. Antonelli C, Donzelli P, Mastroianni R, Setola S, Vinti S and Tucci A (2000) A web-based workflow solution to support the Italian government agenda definition process. In: Proceedings of the 2000 Conference of the Italian Automatic Computation Association, Taormina, Italy, 27–30 September, 2000
35. Donzelli P, Bresciani P (2003) Goal oriented requirements engineering: a case study in e-government. In: Proceedings of the 15th Conference on Advanced Information Systems Engineering (CAISE 2003) Klagenfurt/Velden, Austria, 16–20 June, 2003
36. Donzelli P, Setola R (2001) Agents and goals in the requirements engineering process—a case study. In: Proceedings of the 2002 Conference of the Italian Automatic Computation Association, Pisa, Italy, 18–20 December, 2001
37. Bussler C (1999) Enterprise-wide workflow management. IEEE Concurrency 7(3):32–43