Integrating Use Cases and Organizational Modeling

Victor F.A. Santander [*] Jaelson F. B. Castro

Universidade Federal de Pernambuco – Centro de Informática
Cx. Postal 7851, CEP 50732-970, Recife-PE, BRAZIL
Phone: (+55 81) 3271-8430, Fax: (+55 81) 3271-8438
{vfas,jbc}@cin.ufpe.br

Abstract

The object oriented development paradigm has attracted many supporters in the Software Engineering community. One of the most important advances was the Unified Language Modeling (UML), a standard for visual modeling. Use Case Diagrams have been used for capturing system functional requirements. However, the system development occurs in a context where organization processes are well established. Therefore, we need to capture organizational requirements to define how the system fulfils the organization's goals, why it is necessary, what are the possible alternatives, what are the implications to the involved parts, etc. Unfortunately, UML is ill equipped for modeling organizational requirements. We need other techniques, such as i, to represent these aspects. Nevertheless, organizational requirements must be related to functional requirements represented as Use Cases. In this paper we present some guidelines to assist requirement engineers in the development of Use Cases from the organizational models represented by the i* technique.*

KeyWords: Requirements Engineering, Organizational Modeling, Use Case.

1. Introduction

System development occurs in a context where organizational processes [4] are well established. However, as discovered in empirical studies, the primary reason for software system failure is the lack of proper understanding of the organization by the software developers. Therefore, we need to capture organizational requirements to define how the system fulfils the organizational goals, why it is necessary, what are the possible alternatives, what are the implications to the involved parts, etc. Unfortunately, the dominant object oriented modeling technique, UML, is ill equipped for organizational requirement modeling. We need others techniques, such as i* [19] to represent these aspects. We argue that the i* framework is well suited to represent organizational requirements that occur during the early-phase requirements capture, since it provides adequate representation of alternatives, and offers primitive modeling concepts such as softgoal and goal. These early activities would enable an understanding of how and why the requirements came about.

Nevertheless, organizational requirements must be related to functional requirements represented with techniques such as Use Cases. However, Use Case development demands great experience of the requirement engineers. The heuristics presented in the literature to develop Use Cases [3] [12] [18] are not sufficient to allow a systematic development. Indeed, they do not consider relevant organizational aspects such as goals and softgoals. In this work, we propose some guidelines to support the integration of i* and Use Case modeling. In this paper we present guidelines that allow requirement engineers to develop Use Cases (and associated scenarios) from organizational modeling described in i* framework. This paper is

[*] Partially Supported by CNPq Grant No. 147192/1999-4. On leave from Universidade Estadual do Oeste do Paraná.

an evolution of a preliminary version [17]. We have improved the earlier guidelines and have applied it to a new case study. This paper is organized as follows. Section 2 introduces theconcepts used by i* framework to represent organizational requirements and early requirements. In Section 3, we review Use Case modeling. In Section 4, we present the benefits of our approach as well as describe the guidelines to integrate i* organizational models and Use Case diagrams. In Section 5, we present a brief case study to show the viability of our proposal. Section 6 presents the related works and our conclusions.

2. The i* Modeling Framework - SD

When developing systems, we usually need to have a broad understanding of the organizational environment and goals. The i* framework [19] provides understanding of the reasons ("Why") that underlie system requirements. This technique offers a modeling framework that focuses on strategic actor relationships. i* allows the description of the intentions and motivations involving actors in an organizational environment. It offers two models to represent these aspects: The Strategic Dependency (SD) Model and the Rationale Dependency (SR) Model.

2.1. The Strategic Dependency Model

This model focuses on the intentional relationships among organizational actors. It consists of a set of nodes and links connecting them, where nodes represent actors and each link represents the dependency between two actors. The depending actor is called Depender and the actor who is depended upon is called Dependee. Hence, this model consists of a set of relationships among actors, capturing intentions and motivations among them. The i* framework defines four types of dependencies among actors: goal dependency, resource dependency, task dependency and softgoal dependency. In a Goal Dependency, an actor depends on another to fulfil a goal, without worrying how this goal will be achieved. In a resource dependency, an actor depends on another to provide a physical resource or information. In a task dependency, an actor depends on another to realize some sequence of activities. Finally, in a softgoal dependency an actor depends on another to fulfil a fuzzy goal. The softgoal dependency is a different dependency because it represents a goal not precisely defined. In requirement engineering, a softgoal represents non-functional requirements.

Figure 1 shows an Strategic Dependency (SD) Model for an e-commerce example [21]. The *Medi@* software system is viewed as full-fledge actor in this model. *Customer* depends on *Media Shop* to buy media items while *Media Shop* depends on *Customer* to increase market share and make them happy (with *Media Shop* service). *Media Supplier* is expected to supply *Media Shop* with media items in a continuous way, depending on the latter for continuing business. It can also use *Medi@* to determine new needs from customers, such as media items not available in the catalogue while expecting *Media Producer* to provide her with quality packages. *Media Shop* depends on *Medi@* for processing internet orders and on *Bank Cpy* to process business transactions. *Customer*, in turn, depends on *Medi@* to place orders through the internet, to search the database for keywords, or simply to browse the on-line catalogue. With respect to relevant qualities, *Customer* requires that transactions services be secure and available, while *Media Shop* expects *Medi@* to be easily adaptable (e.g., catalogue enhancing, item data-base evolution, user interface update...). Finally, *Medi@* relies on internet services provided by *Telecom Cpy* and on secure on-line financial transactions handled by *Bank Cpy*.

A number of schemes have been proposed to express this type of knowledge about system environment to various degrees of formality. The models are interpreted prescriptively, i.e., the system (here the Medi@) and other agents (customer, media shop, media supplier, media producer, telecom cpy and bank cpy) are supposed to conform to the model. In contrast, the i* framework supports a descriptive view for developing a deeper understanding about the organizational environment in which the proposed system is to be embedded, and is aimed at a phase in requirements engineering before the prescriptive requirements are arrived at. One aims to identify who will be affected (stakeholders), and how their strategic interests would be affected by changes in the work processes associated with the proposed system.

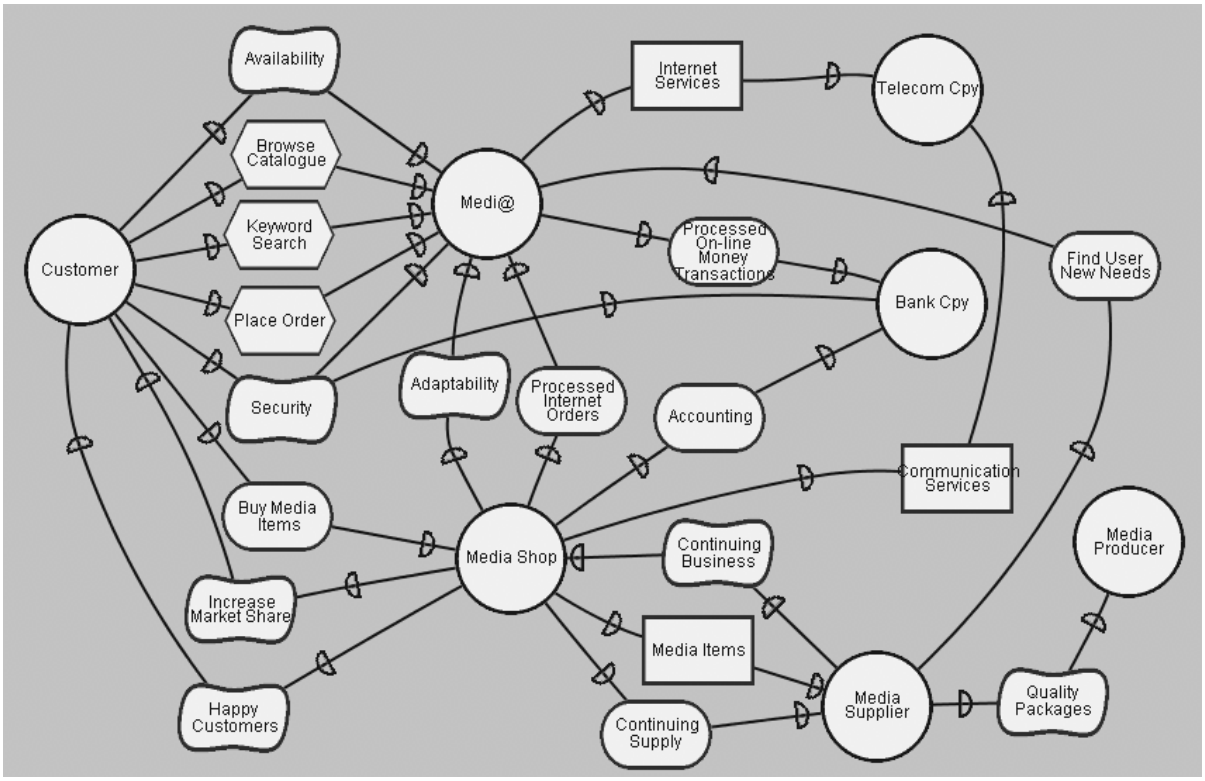

Figure.1. Strategic Dependency Model for a Media Shop.

2.2. The Strategic Rationale Model - SR

The Strategic Rationale (SR) model is a supplementary model to the Strategic Dependency (SD) model. This model allows modeling of the reasons associated with each actor and their dependencies. The strategic rationale model assists in requirements engineering by allowing process elements and the rationales behind them to be expressed. During early requirements engineering, the SR model can be used to understand how systems are embedded in organizational actors' routines, to generate alternatives, and to model and support actors' reasoning about the alternatives.

Two news links are added to previous notation:

- Means-ends: This link indicates a relationship between an end - which can be a goal to be achieved, a task to be accomplished, a resource to be produced, or a softgoal to be satisficed - and a means for attaining it. The means is usually expressed in the form of a task, since the notion of task embodies how to do something. This is done by way of describing the elements (components) of a task.
- Task-decomposition: A task is modeled in terms of its decomposition into its sub-components. These components can be goals, tasks, resources, and/or sofgoals. The

distinction among these have been introduced in the Strategic Dependency model, which they appear as dependums in the strategic relationships between actors. These same distinctions are useful in elaborating on the makeup of a task.

In Figure 2, we present an example of the Strategic Rationale (SR) model.

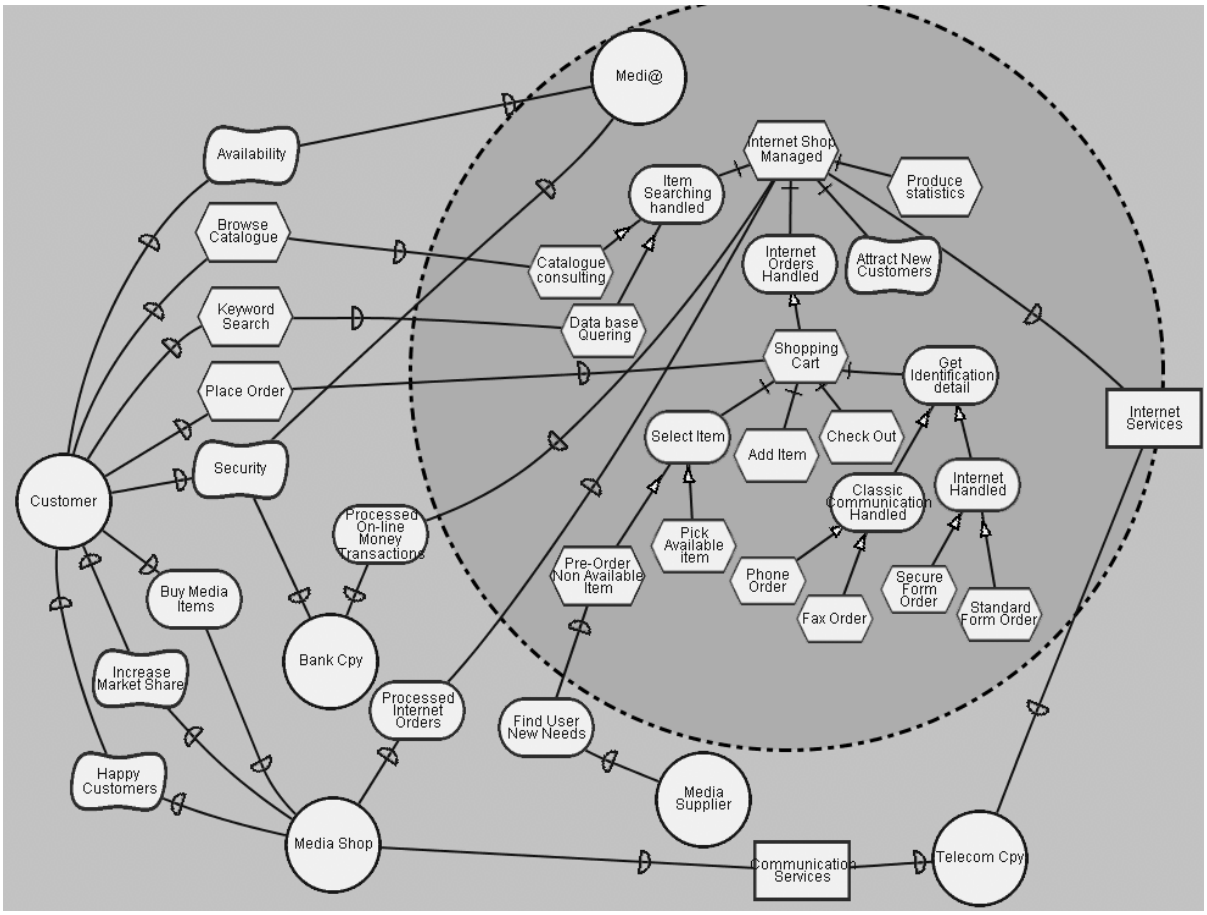

Figure 2. Strategic Rationale (SR) Model for Medi@.

We use the SR notation to detail the *Medi@* actor. Due to space limitation, we do not refine the other actors.The Figure 2 postulates a root task *Internet Shop Managed*. This task is firstly refined into goals *Item Searching Handled* and *Internet Order Handled*, sofgoal *Attract New Customer* and task *Produce Statistics*. *Internet Order Handled* is achieved through the task *Shopping Cart*, which is decomposed into *Select item*, *Add Item*, *Check Out*, and *Get Identification Detail*. These are the main process activities required to design an operational on-line shopping cart. The latter (*Get Identification Detail*) is achieved either through sub-goal *Classic Communication Handled* dealing with phone and fax orders or *Internet Handled* managing secure or standard form orderings. To allow for the ordering of new items not listed in the catalogue, *Select Item* is also further refined into two alternative sub-tasks, one dedicated to select catalogued items, the other to preorder unavailable products. The goal *Item Searching Handled* might alternatively be fulfilled through tasks *Database Querying* or *Catalogue Consulting* with respect to customers' navigating desiderata, i.e., searching with particular items in mind by using search functions or simply browsing the catalogued products.

3. Use Cases in UML

Scenario-based techniques have been used by the software engineering community to understand, model and validate users requirements [10] [11] [15] [16] [18]. Among these

techniques, Use Cases have received a special attention in the object oriented development community. In fact, Use Case diagrams are part of the Unified Language Modeling (UML), a standard for visual modeling.

Use Cases in UML [3] are used to describe the use of a system by actors. An actor is any external element that interacts with the system. A Use Case is a description of a set of sequences of actions, including variants, that a system performs that yields an observable result value to an actor. It is desirable to separate main (primary scenario) versus alternative (secondary scenario) flows because a Use Case describes a set of sequences, not just a single sequence, and it would be impossible to express all the details of an interesting Use Case in just one sequence.

In order to cope with increasing complexity of Use Cases description, UML caters for three structuring mechanism: inclusion, extension and generalization. Note that the generalization mechanism can also be used to relate actors, i.e. one actor can be a specialization of another actor. For further details see [3].

Several heuristics to aid requirement engineers in the Use Case development are presented in [3] [12] [18]. However, the discovery and description of Use Cases is not so simple, because, in most of the situations, it demands a certain degree of experience from requirement engineers. The first aspect is to discover the actors that interact with the system. Another difficulty is to consider the behaviour that each actor expects or requires the system to provide. One of the alternatives to assist this process is indicated in [8]. Cockburn argue that stakeholders possess goals in relation to the system to be developed. These goals can be of higher or lower level and can originate Use Cases for the intended system.

However, the main challenge is related to how system goals can be initially discovered. Traditionally, requirement engineers accomplish this work, using mainly their experiences to discover these goals and then to describe Use Cases. In most situations, Use Cases are developed without considering the organizational requirements previously defined by the organization. In this sense, we believe that a viable alternative is to begin the Use Case discovery process investigating the goals and other elements represented in the **i* organizational models**. The use of the defined information in organizational models can facilitate the Use Case development as well as turn this process more systematic. Thereby, in this paper we propose to develop Use Case starting from the observation and analysis of organizational goals and other elements represented through **i***.

4. Deriving Use Cases from Organizational Modeling.

In this section we argue how our approach can improve the Use Case development. In section 4.1 we outline the main benefits accomplished by approach and in section 4.2 we describe it in detail.

4.1. Benefits of I* and Use Case Integration

We have shown that i* provides an early understanding of the organizational relationships in a business domain. As we continue the development process, we need to focus on the functional and non-functional requirements of the system-to-be, which will support the chosen alternative among those considered during early requirements. As a first step in the late requirements phase we can adopt Use Case notation to describe functional requirements of the system. We argue that the Use Case development from organizational modeling using i* allows requirement engineers to establish a relationship between the functional

requirements of the intended system and the organizational goals previously defined in the organization modeling. Besides, through a goal-oriented analysis of the organizational models, we can derive and map goals, intentions and motivations of organizational actors to the main goal of Use Case. We assume, that for each Use Case we have associated a main goal, which represents what the user aims to reach as a result of the execution of the Use Case. In our proposal, the Use Case scenario description is based on organizational models, which are well known and understood by all stakeholders. Note that our approach can be used for any type of system.

We can mention other important benefits obtained using our approach, such as:

- Many researchers [1] [5] [7] [9] [16] [19] [20] have considered goals in a number of different areas of Requirements Engineering. Goal-oriented approaches to requirements acquisition may be contrasted with techniques that treat requirements as consisting only of processes and data, such as traditional systems analysis or "objects", such as the object-oriented methods, but which do not explicitly capture why and how relationships in terms of goals.
- The relationships between systems and their environments can also be expressed in terms of goal-based relationships. This is partly motivated by today's more dynamic business and organizational environments, where systems are increasingly used to fundamentally change businesses process [20]. Deriving Use Cases from i* relationships allows traceability and evaluation of the impact of these changes into the functional requirements of the intended system;
- Some of the Use Case pitfalls and drawbacks described in [12], can be partially solved using our approach. For instance, Use Cases are written from the actor's (not the system's) point of view. We derive Use Cases from actors dependencies defined explicitly in i*. Another positive aspect is the ability to define the essential Use Cases for the intended system. This avoids defining too many Use Cases and allows managing the appropriate granularity of Use Cases. Finally, the integration between requirements engineers and customers during the organizational model development also allows customers (actors) to better understand the Use Cases originated from these models;
- To elicit and specify system requirements observing the actor's goal in relation to the system-to-be, is a way of clarifying requirements [20]. From i* we can derive these goals, associate them with system actors and then refine and clarify the requirements into Use Cases.

4.2. Proposed Approach

To guide the mapping and integration process of i* organizational models and Use Cases, we have defined some guidelines. These guidelines must be applied according to the steps represented in Figure 3.

In this Figure, steps 1, 2 and 3 represent the discovery of system actors and its associated Use Case diagrams and descriptions. The input for the integration process are the Strategic Dependency (SD) and Strategic Rationale (SR) models developed through i* framework. In steps 1 and 2, the input is the Strategic Dependence (SD) Model. The description of scenarios for Use Cases (step 3) is derived from elements represented in the Strategic Rationale (SR) Model. The results of the integration processes are Use Case diagrams for the intended system and scenario textual descriptions for each Use Case.

In the sequel we suggest heuristics for the Use Case development from organizational modeling with i*.

1º Step: Discovering System Actors.

Guideline 1*: *every actor in i* should be considered for a possible mapping for ***actor in Use Case***; *For instance, we can analyze the Customer actor in Figure 1.*

***Guideline 2*:** the actor considered in i* should be external to the intended software system; *For example, the Customer actor is external to the system.*

Guideline 3: if the actor is external to the system, it should be guaranteed that the actor in i* is a candidate actor in the Use Case diagram. For this purpose, the following analysis is necessary:

> ***Guideline 3.1:*** the actor dependencies in i* must be relevant from the point of view of the intended system; this can be verified if there is at least one dependency between the analyzed actor and the intended system. *For instance, the Customer actor in i* can be mapped to Use Case actor, considering that dependencies associated with it, characterizes it as important in an interaction context with the Medi@ system.*

Guideline 4: actors in i*, related through the IS-A mechanism in the organizational models and mapped individually for actors in Use Cases (applying guidelines 1, 2 and 3), will be related in the Use Case diagrams through the <<generalization>> relationship.

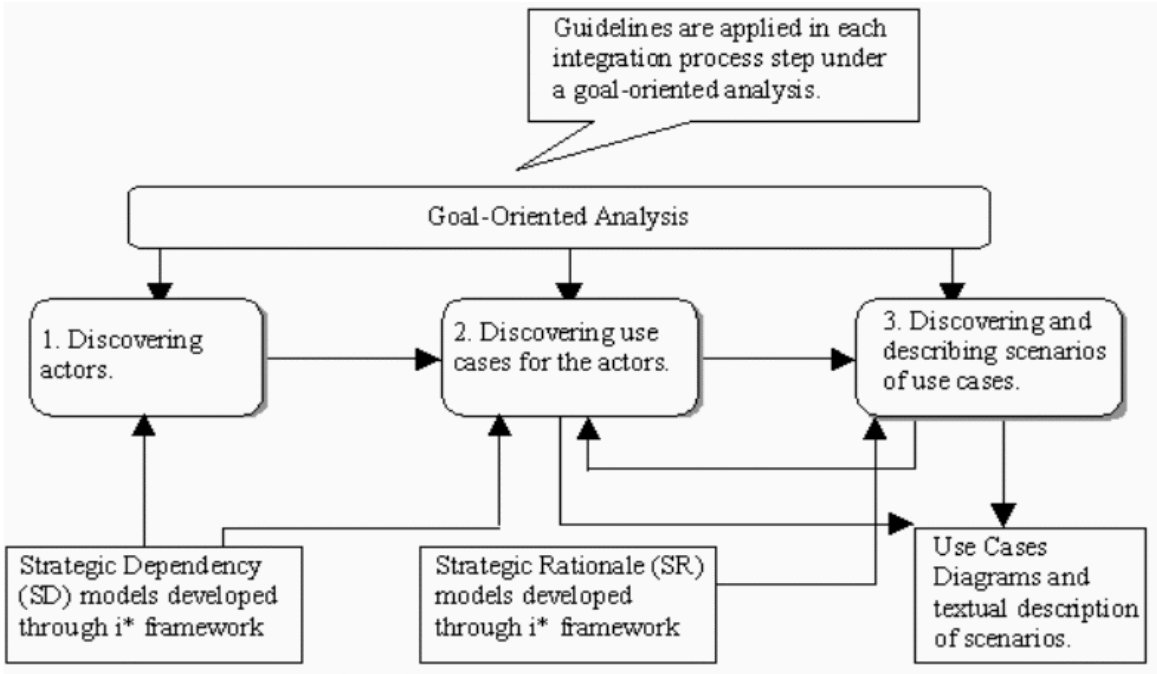

Figure 3. Steps of the integration process between i* and Use Cases in UML

2º Step: Discovering Use Cases for the Actors.

Guideline 5*:** for each discovered ***actor of the system (step 1), we should observe all the dependencies (dependum) between the system-to-be and the actor in which the discovered actor is a ***dependee***, looking for Use Cases for the actor; Initially, we recommend to create a table containing the discovered actors and the information about the dependencies for the actor from the point of view of a dependee. We also suggest including in this table the dependencies of the discovered Use Case actor now from the point of view of a ***depender*** related to the system-to-be (dependee). This information will be useful applying guideline 6. Moreover, you can include which guideline(s) to be used to analyze each dependency (dependum) (see table 1).

Guideline 5.1*:** goal dependencies - goals in i* can be mapped to Use Case goals; *For instance, in Figure 1, the goal dependency Processed On-line Money Transactions between Medi@ (Depender) and Bank Cpy (Dependee) can be mapped to the* ***Processed On-line Money Transactions *Use Case, which will contain the several steps accomplished by Bank Cpy dealing with on-line money transactions.*

Actor	Dependency	Type of Dependency	Guideline to be used
Bank Cpy	Processed On-line Money Transactions	Goal	(G5.1)

Table 1. Gathered information from SD Models to aid requirement engineers to derive Use Cases.

Guideline 5.2: task dependencies - if an actor depends on another actor for the accomplishment of a task, it should be investigated if this task needs to be decomposed into other sub-tasks. *For example, for the task dependency Place Order between Customer and Medi@ actors (see Figure 1), we must considering if the execution of this task requires several steps (later mapped to Use Case steps) such as select item, add item to shopping cart and check out. Thus, from the Place Order task we can generate the Use Case called* ***Place Order*** *for the Customer actor. (Notice that in this example, the Customer actor is observed as depender and the Medi@ actor is observed as dependee. This situation is explained in the guideline 6).*

Guideline 5.3*:** resources dependencies - if an actor depends on another actor for obtaining a resource(s), why is it required? If there is a more abstract goal, this goal will be candidate to be the goal of the Use Case for the actor. *For instance, for the resource dependency Internet Services associated with the Telecom Cpy actor (see Figure 1), we conclude that Telecom Cpy provides internet services to be used by the Medi@ system. We could consider that to provide internet services involve several interaction steps between Telecom Cpy and Medi@ which could be defined in one Use Case called* ***Internet Services *for the Telecom Cpy.*

Guideline 5.4: sofgoal dependencies - Typically, the sofgoal dependency in i* is a non-functional requirement for the intended system. Hence, a softgoal does not represent a Use Case of the system but a non-functional requirement associated with a specific Use Case of the system or with the system as a whole. (This guideline will be improved in the future versions of this work). *For instance, the softgoal Increase Market Share between Media Shop and Customer actors can be mapped into a non functional requirement associated with the Use Case Place Order. This indicates that the Use Case Place Order can contribute to "satisfice" the non-functional requirement Increase Market Share for the Media Shop.*

Guideline 6: analyze special situations, where an actor discovered (following the step 1), possess dependencies in relation to an actor in i* that represents the system-to-be or part of it. These dependencies usually generate Use Cases. It is important to notice that in this situation the derived Use Case is associated with the **depender** actor in the relationship. This occurs due to the fact that the dependee is a software system and the depender (Use Case actor) must interact with the system to achieve the goal associated with the generated Use Case. *For instance, the goal dependency Processed Internet Orders between Media Shop and Medi@ system in the Figure 1, points out for the definition of the Use Case* ***Processed Internet Orders*** *for the Media Shop actor, which represents the use of the system by the actor, describing the details to process internet orders.*

Guideline 7: classify each Use Case according to the type associated to its goal (business, summary, user goal or subfunction). This is based on classification proposed by Cockburn [8].

A business goal represents a high level intention, related to business processes, that the organization or user possesses in the context of the organizational environment. An example could be the goal “the media shop wishes to increase market shares”. A summary goal represents an alternative for the satisfaction of a business goal. One Alternative could be the goal, “increase market shares by using an e-commerce application”. An user goal results in the direct discovery of a relevant functionality and value for the organization actor using a software system. An example could be the goal, "the customer wishes to place order”. Finally, subfunction-level goals are those required to carry out user goals. An example could be the goal, “the customer browse catalogue”. To aid requirement engineers to identify new Use Cases and to better understand them, we recommended to generate a table containing the actor name, the Use Case goal and the goal classification (see table 2).

Actor	Use Case Goal	Goal Classification
Media Shop	Processed Internet Orders	Summary Goal

Table 2. Use Case goal classification.

3º Step: Discovering and Describing Scenarios of Use Cases.

Guideline 8: analyze each actor and its relationships in the Strategic Rationale (SR) model, to extract information that can lead to the description of the Use Cases scenario for the actor. It is important to remember that SR models represent the internal reasons associated with the actor goals. Therefore, we must consider internal elements that are used by the actor to achieve goals and sofgoals, to perform tasks or obtain resources. The actor has the responsibility to satisfy these elements and the decomposition in SR shows how the actor will be performing this. Typically, the dependencies associated with the actor are satisfied internally through two types of relationships used in SR: **means-ends and task-decomposition.** These relationships must be observed to derive scenario steps for the Use Cases. Sub-components in the task-decomposition link usually can be mapped to steps (activities) of Use Case scenarios associated with the task. Note that if the task being decomposed fulfils some dependency (with other actors) previously mapped for Use Case, the sub-components are mapped to activities (steps) of the Use Case primary scenario. On the other hand, a **mean** for attaining an end (which can be a goal to be achieved, a task to be accomplished, a resource to be produced, or a softgoal to be satisficed) through mean-end link represents alternatives to achieve the end. If this end is a goal or task that fulfils some dependency previously mapped to Use Case, these alternatives (means) are described as extends in the Use Case scenario description. Additionally, we also can associate sofgoals represented in the SR model with Use Cases. If a sub-component in a task-decomposition relationship is a sofgoal and the decomposed task fulfils some dependency mapped to Use Case, this sofgoal is to be associated with the Use Case as special requirement (non-functional requirement) in the primary scenario (an example for this case is presented in the Processed Internet Orders Use Case scenario description in the section 5).

For instance, let us observe the Strategic Rationale (SR) Model in Figure 2. From the Medi@ actor point of view, we verify that the task Shopping Cart is used by Medi@ to achieve (satisfy) the goal dependency Place Order for the Customer actor. Hence, Shopping Cart is decomposed into sub-tasks Select Item, Add Item, Check Out, and Get Identification Detail, which are the main process activities required to design an operational on-line shopping cart and satisfy the place order goal. We could adopt that these activities are the necessary high-level steps to place order (Use Case ***Place Order*** *defined for the Customer actor). Thus, this Use Case could contain the steps (the primary scenario description) required to place order*

by customers. Additionally, as Select Item task can be achieved (via mean-end link) by two alternatives sub-tasks, one dedicated to select catalogued items and other to preorder unavailable products, both sub-tasks (means) can be mapped to extends descriptions in the Use Case scenario description.

5. Case Study

In this section, we follow the steps proposed in Figure 3 and apply the appropriate guidelines to the example described in the previous section (Figure 1 and 2). Recall that Figure 1 described a Strategic Dependency (SD) model for the media shop while Figure 2 represents the Strategic Rationale (SR) model. Hence, these organizational models are used to discover and des+cribe Use Cases in UML for the Medi@ system. We begin deriving the Use Case actors from the SD model. We then find the Use Cases for the actors observing the actors dependencies in SD model. Next, the primary scenario for two derived Use Cases are described from the SR model. Last but not least, a version of the Use Case diagram in UML for the Medi@ system is generated.

- From Figure 1, we can find candidates actors for the Use Case development. According to the guidelines in the ***1st step*** of the proposal, we conclude that one of the analyzed actors does not follow guideline 2. The Medi@ actor is a system, i.e. the software to be developed. Therefore, this i* actor cannot be considered as a Use Case actor. The Media Producer also is not considered Use Case actor because it does not follow the guideline 3, i.e., the Media Producer actor i* dependencies are not relevant from the point of view of interaction context with the Medi@ system. The other i* actors are considered appropriate because their strategic dependencies refer to relevant aspects for the Medi@ system (guideline 3) development. So, the list of candidates Use Cases actors includes: ***Media Shop, Media Supplier, Bank Cpy, Telecom Cpy and Customer.***

The next step is to discover and relate Use Cases for each actor according to the guidelines presented in the ***2º Step*** *(Discovering Use Cases for the Actors*).

- Initially, following guideline 5 and observing the SD model presented in Figure 1 we can generate the table:

Actor	Dependency	Type of Dependency	Guideline to be used
Media Shop	Processed Internet Orders	Goal	(G 6, G5.1)
Media Shop	Adaptability	Sofgoal	(G 6, G5.4)
Media Supplier	Find User New Needs	Goal	(G 6, G5.1)
Bank Cpy	Processed On-line Money Transactions	Goal	(G5.1)
Telecom Cpy	Internet Services	Resource	(G5.3)
Customer	Browse Catalogue	Task	(G 6, G5.2)
Customer	Keyword Search	Task	(G 6, G5.2)
Customer	Place Order	Task	(G 6, G5.2)
Customer	Availability	Softgoal	(G 6, G5.4)
Customer	Security	Softgoal	(G 6, G5.4)

Table 3. Gathered information from SD Models to derive Use Cases for the Medi@ System.

- Thus, for the Media Shop actor, observing this actor as Dependee (guideline 5), no dependencies exist between this actor and Medi@ system. However, observing the actor as

depender and Medi@ system as dependee (guideline 6), we find the Processed Internet Orders goal dependency (see table 3). This dependency originates the ***Processed Internet Orders*** Use Case considering that several steps are necessary to achieve this goal including activities such as to handle item searching, handle internet order and produce statistics.

- To find Use Cases candidates for the Media Supplier actor, we should follow the same guidelines (*2º Step)* used for Media Shop actor. Observing this actor as Dependee (guideline 5), no dependencies exist for the actor. However, observing Media Supplier as depender and Medi@ system as dependee (guideline 6), the Find User New Needs goal dependency (see table 3) is verified. The Media Supplier depends on Medi@ system to find user new needs. To achieve this goal we can include several aspects (sub-tasks) such as to verify unavailable items searched by customers, to get suggestions from customers through internet and to pre-order unavailable items. Thus, the Find User New Needs goal can generates the Use Case called **Find User New Needs** for the Media Supplier actor which includes the necessary steps (scenario description) on interaction between Medi@ and Media Supplier to find user new needs.
- For the Bank Cpy actor, observing this actor as Dependee (guideline 5), we can verify the Processed On-line Money Transactions goal dependency (see table 3). This dependency can be mapped to the **Processed On-line Money Transactions** Use Case, which will contain the several steps on interaction between Bank Cpy and Medi@ for dealing with on-line money transactions.
- For the Telecom Cpy actor, observing this actor as Dependee (guideline 5), we can verify the Internet Services resource dependency (see table 3). Following guideline 5.3, we conclude that the main goal of obtaining of Internet Services resource is to provide Medi@ with internet services. We could consider that in this process, several interactions steps between the system and the Telecom Cpy actor could be defined in one Use Case called **Internet Services** for the Telecom Cpy actor.
- For the Customer actor we can indicate some Use Cases originated from the actor dependency relationships (guideline 5). Initially, observing this actor as Dependee (guideline 5), we do not find dependencies between this actor and Medi@ system. However, observing Customer as depender and Medi@ system as dependee (guideline 6), we have three task dependencies that can originate Use Cases: Browse Catalogue, Keyword Search and Place Order (see table 3). The Browse Catalogue dependency can be mapped to the **Browse Catalogue** Use Case, which will contain the several interaction steps required to browse catalogue. The Keyword Search dependency can be mapped to the **Keyword Search** Use Case, which will contain the several interaction steps required between Customer and Medi@ to search a keyword. Finally, the Place Order dependency can be mapped to the **Place Order** Use Case, which will contain the several interaction steps between Customer and Medi@ to place an order through internet.

  It is important to note that usually, the description of the primary scenario (to be accomplished later) for the discovered Use Cases, includes other user goals that can originate new Use Cases for the system. The mapped Use Cases in this step represent the essential Use Cases for the Medi@ system.
- Next, following the guideline 7 we can classify each discovered Use Case goal, as shown in the table 4.

Thereby, after we have used the proposed guidelines (*2º Step*), we have discovered **Processed Internet Orders** Use Case for the Media Shop actor, **Find User New Needs** Use Case for the Media Supplier actor, **Processed On-line Money Transactions** Use Case for the

Bank Cpy actor, **Internet Services** Use Case for the Telecom Cpy and **Browse Catalogue**, **Keyword Search** and **Place Order** Use Cases for the Customer actor. Therefore**,** we can begin the description of the primary and secondary scenarios and the Use Cases relationships (*3º Step*). At this point, the Strategic Rationale (SR) model is used as source of information for the scenario description and the Use Case relationships.

Actor	Use Case Goal	Goal Classification
Media Shop	Processed Internet Orders	Summary Goal
Media Supplier	Find User New Needs	User Goal
Bank Cpy	Processed On-line Money Transactions	Summary Goal
Telecom Cpy	Internet Services	User Goal
Customer	Browse Catalogue	Subfunction
Customer	Keyword Search	Subfunction
Customer	Place Order	User Goal

Table 4. Goal Classification for the Medi@ System.

For example, the ***Place Order*** Use Case discovered for the Customer actor represents the use of the system by Customer to place an order. In the SR model (Figure 2), the Medi@ system satisfies the task dependency Place Order (mapped to Place Order Use Case) through the internal task Shopping Cart and its sub-tasks Select Item, Add Item, Check Out and its sub-goal Get Identification Detail (see Figure 2). These sub-components can be mapped to steps of the Place Order Use Case primary scenario, as follow:

Use Case Goal: **Place Order**
Level: **User Goal** (see guideline 7)
Actor: **Customer**
Primary Scenario:
1. The Use Case begins with the Customer selecting an available item;
2. The Customer adds item to the Shopping Cart;
3. The Customer proceeds to check out and provides the system with the necessary identification information.
4. The system gets customers identification detail through Internet and closes the order.

Extends:
1.a the customer pre-orders an unavailable item;
4.a the customer chooses to provide identification details by classic communication.

Thus, following the guideline 8, we must observe which elements are involved in the SR model to perform the Place Order task by Medi@ actor. This actor has the responsibility to perform Place Order, which generated the **Place Order** Use Case (according to guideline 6). Thereby, steps 1, 2, 3 and 4 are extracted from the decompositions of the Shopping Cart task that satisfies the Place Order task dependency (see Figure 2). Step 1 in this Use Case is extracted from the Select Item task, establishing that Customer needs to select a product. We can observe that Select Item can be achieved (mean-end link) through two alternative sub-tasks, one dedicated to select catalogued items and other to preorder unavailable products. Thus, the Customer can pick up an available catalogued item (chosen as the normal action in the scenario and described in step 1) or preorder an unavailable product (defined as extend (1.a) for the step 1 considering that this action is usually an exception). Step 2 derives from

the observation of the Add Item sub-task establishing that Customer needs to add item to shopping cart (see Figure 2). Step 3 originates from Check Out sub-task. In this step the customer proceeds to the order check out.

Step 4, is extracted from observation of the Get Identification Detail goal, establishing that Customer needs to provide the system with identification detail to successfully perform the order. This activity is achieved (mean-end link) either through sub-goal Internet Handled managing secure or standard form orderings (chosen as the normal action in the scenario and described in step 4) or Classic Communication Handled (defined as extend (4.a) for this step) dealing with phone and fax order (see Figure 2).

We also can describe the primary scenario for the **Processed Internet Orders** Use Case discovered for the Media Shop actor. This Use Case should contain all the necessary steps related the Process Internet Orders goal. This goal is satisfied by Medi@ system through the task Internet Shop Managed and its sub-goals Item Searching Handled and Internet Orders Handled, its sub-task Produce Statistics and its sofgoal Attract New Customers (see Figure 2). Thus, for the Processed Internet Orders Use Case we could have the primary scenario with the following steps:

Use Case Goal: **Processed Internet Orders**
Level: **Summary Goal** (see guideline 7)
Actor: **Media Shop**
Primary Scenario:

1. The Use Case begins when Media Shop requires item searching to be handled by database querying; (The Use Case *Keyword Search* is included << include >> in this step);
2. The system handles internet orders; (The Use Case *Place Order* is included << include >> in this step);
3. The system produces statistics for Media Shop;

Extends
1.a the Item Search is handled by catalogue consulting: the *Browse Catalogue* Use Case is called.
Special Requirements: Attract New Customer

The steps 1, 2 and 3 are extracted from the decompositions of the Internet Shop Managed task that fulfils the goal dependency Processed Internet Orders (see Figure 2). Step 1 in this Use Case is extracted from the Item Searching Handled goal, establishing that Media Shop requires that item searching be handled. We can observe that Item Searching Handled might alternatively be fulfilled (mean-end link) through tasks Database Querying or Catalogue Consulting with respect to customer's navigating desiderata, i.e., searching with particular items in mind by using search functions or simply browsing the catalogued products. Thus, the Media Shop requires that item searching be handled by Database Querying (chosen as the normal action in the scenario and described in step 1) or alternatively be handled by Catalogue Consulting (defined as extend (1.a) for this step). Note that Database Querying fulfils the task dependency Keyword Search that originated the previously defined Keyword Search Use Case (see table 4). Thus, the ***Keyword Search*** Use Case is included <<include>> in this step because it represents the necessary steps (actions) to get item searching handled by database querying. Alternatively, as Catalogue Consulting fulfils the task dependency Browse Catalogue (which generated the Browse Catalogue Use Case) we extended <<extend>> **Browse Catalogue** Use Case in the step 1.a because it represents the necessary steps (actions) to get item searching handled by catalogue consulting.

Step 2 derives from the observation of the Internet Orders Handled goal. This goal establishes the need to handle internet orders. However, this activity is achieved (mean-end link) through task Shopping Cart. Note that Shopping Cart fulfils the task dependency Place Order that originated the previously defined Place Order Use Case (see table 4). Thus, the **Place Order** Use Case is included <<include>> in this step because it represents the necessary steps (actions) to handle internet orders using Shopping Cart. Step 3 originates from Produce Statistics task. This step represents the actions of the system and media shop to produce statistics that can be used by media shop to improve the e-commerce business. Finally, the last sub-component of the Internet Shop Managed task, the sofgoal Attract New Customer has been defined as Special Requirements (non functional requirements) for the Processed Internet Orders Use Case.

Note, that we can describe the others Use Cases for Medi@ system in a similar way. Additionally, we also can observe the sub-tasks and sub-goals mapped to steps of the discovered use cases looking for new use cases (applying guidelines 5.1 and 5.2). Due to space limitation, these elements have not been evaluated.

After we have applied the proposed guidelines to this case study, we can define, as described in the Figure 4, a version of the Use Cases diagram in UML for the Medi@ system. The descriptions of the discovered Use Cases could still be modified or complemented, as new relationships are found. Another important aspect is that the development of other Use Cases depends on the requirement engineers' experience. However, modeling of this nature can vary, it aims to facilitate the understanding as well as to establish an agreement between customers and developers in the system requirements definition.

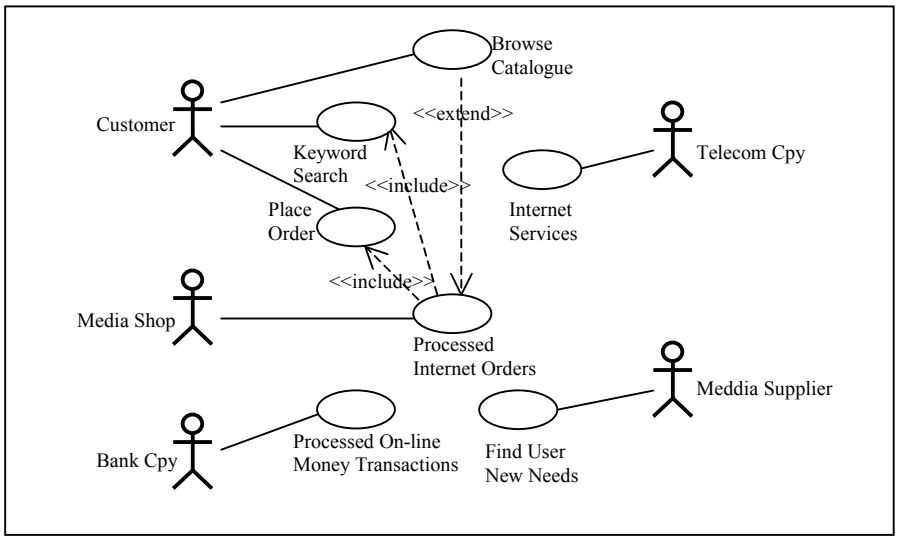

Figure 4. Use Case Diagram for the Medi@ system.

5. Conclusions

In this paper we argued that the Use Cases development can be improved by using the i* organizational models. We presented some heuristics seeking to show the viability and benefits of integrating organizational models developed using the i* framework with Use Cases in UML. Both techniques were described and the proposed guidelines were applied partially to a Medi@ system case study. Starting from the case study it was possible to observe that the existent information in the strategic dependency model as well as in the strategic rationale model can be used as base for the Use Case development. Besides, it enables the requirement engineers, starting from a more detailed observation of the organizational models, to choose the best alternative for the software development as well as to concentrate in the Use Cases that really fulfils the organizational goals. In the traditional

development, Use Cases and scenarios do not consider in an effective way, motivations, intentions and alternatives for the systems development. In this paper we show how the use of organizational models can assist these activities.

Using our proposal that integrates Organizational Models and Use Cases, some important issues such as how the system fulfils the organization goals, why it is necessary, what are the possible alternatives, what are the implications to the involved parts, can be better treated and proposed solutions incorporated into the software system development.

Some related works include the proposal of requirements-driven development presented in the ***Tropos*** framework [5] [14] and the proposal of the integration of i* and pUML diagrams [6]. These works argue that organizational models are fundamental for the development of quality software, which can satisfy the real needs of users and organizations. Other related works on requirements engineering combines goals and scenarios. For instance, the ScenIC method [13] uses goal refinement and scenario analysis as its primary methodological strategies. This method includes systematic strategies to identify actors, goals, tasks, and obstacles into evolving systems. In Anton et al. [2], the GBRAM method [1] is used to derive goals from a use-case based requirements specification. The challenges and associated risks building quality system during goal and scenario analysis are described. In the CREWS project [15] [16], the CREWS-L`Ecritoire approach [16] aims at discovering/eliciting requirements through a bi-directional coupling of goals and scenarios allowing movement from goals to scenarios and vice-versa.

However, these approaches do not consider organizational models for deriving goals and scenarios for intended systems.

Further research is still required to describe more systematic guidelines, that can aid requirement engineers to relate non-functional requirements [7] (softgoals in i*) with functional requirements of the system, described through Use Cases in UML. Work is underway to incorporate goal-oriented modeling approaches [1] [9] [13] [16] into our proposal aiming at discovering other Use Cases from exploration of already discovered goals. We also expect to develop more real case studies as well as to provide some tool support for the proposed mapping.

7. References

[1] Anton, A., *Goal identification and refinement in the specification of software-based information systems*. Phd Thesis, Georgia Institute of Technology, Atlanta, GA, June 1997.

[2] Antón, A.I. ., Carter, R.A., Dagnino, A., Dempster, J.H., Siege, D.F., *Deriving Goals from a Use Case Based Requirements Specification*, *Requirements Engineering Journal,* Springer-Verlag, Volume 6, pp. 63-73, May 2001.

[3] Booch, G., Jacobson, I., Rumbaugh, J., *"The Unified Modeling Language User Guide"*, Addison-Wesley, 1999.

[4] Bubenko, J, A., Kirikowa, M., "Worlds" in Requirements Acquisition and Modeling, Sweden, 1994.

[5] Castro, Jaelson F., Kolp, M., Mylopoulos, J., "A Requirements-Driven Development Methodology", In: CAISE'01, Proceedings of the 13th Conference on Advanced Information Systems Engineering. Heildelberg, Germany: Springer Lecture Notes in Computer Science LNCS 2068, pp. 108-123, 2001.

[6] Castro, J., Alencar, F., Cysneiros, G., Mylopoulos, J., "Integrating Organizational Requirements and Object Oriented Modeling", In Proceedings of the Fifth IEEE

International Symposium on Requirements Engineering - RE'01, pp. 146-153, August 27-31, Toronto, 2001.

[7] Chung, L., Nixon, B.A.,Yu, E., Mylopoulos, J., *Non-Functional Requirements in Software Engineering (Monograph)*, Kluwer Academic Publishers, 472 pp, 2000.

[8] Cockburn, A., *Writing Effective Use Cases*, Humans and Technology, Addison-Wesley, 2000.

[9] Dardene,A., Lamsweerde, V., Fikas, S., *Goal-Directed Requirements Acquisition.* Science of Computer Programming, 20, pp. 3-50, 1993.

[10] Jacobson, I., *Object Oriented Software Engineering: A Use Case Driven Approach,* Addison-Wesley, 1995.

[11] Leite, J.C.S.P, Rossi, G., Balaguer, F., Maiorana, V., *Enhancing a requirements baseline with scenarios.* In Proceedings of the Third IEEE International Symposium on Requirements Engineering – RE97, pages 44-53. IEEE Computer Society Press, January 1997.

[12] Lilly, S., Use Case Pitfalls: top 10 problems from Real projects using Use Cases, In: Proceedings, technology of object oriented languages and systems, 1-5 August 1999, pp 174-183.

[13] Potts, C., *ScenIC: A Strategy for Inquiry-Driven Requirements Determination*, In Proceedings of the Fourth IEEE International Symposium on Requirements Engineering – RE'99, June 7-11, Ireland, 1999.

[14] Mylopoulos, J., Castro, J., "Tropos: A Framework for Requirements-Driven Software Development", Brinkkemper, J. and Solvberh, A. (eds), *Information Systems Engineering: State of Art and Research Themes*, Lectures Notes in Computer Science, Springer-Verlag, June 2000.

[15] Ralyté, Jolita., Rolland, C., Plihon, V., *Method Enhancement With Scenario Based Techniques*, In *Proceedings* of CAISE 99, 11th Conference on Advanced Information Systems Engineering Heidelberg, Germany, June 14-18, (1999) (CREWS Report Series 99-10).

[16] Rolland, C., Souveyet, C., Achour, C. B., *Guiding Goal Modeling Using Scenarios*, IEEE Transactions on Software Engineering, Vol 24, No 12, Special Issue on Scenario Management, December 1998.

[17] Santander, V. F., Castro, J.F., Accepted for Publication in IEEE Joint International Requirements Engineering Conference, RE'02, University of Essen, Germany, September, 9-13, 2002.

[18] Schneider, G., Winters, J. P., *Applying Use Cases: a practical guide*, Addison Wesley, 1998.

[19] Yu, E., *Modelling Strategic Relationships for Process Reengineering,* Phd Thesis, University of Toronto, 1995.

[20] Yu, E., Mylopoulos, J., "Why Goal-Oriented Requirements Engineering", Proc. Fourth International Workshop Requirements Engineering: Foundations of Software Quality REFSQ'98, pp 15-22, Pisa, June, 1998.

[21] Castro, J., Kolp, M., Mylopoulos, J., Towards Requirements-Driven Information Systems Engineering: The Tropos Project, 35 pages. IAG Working Paper 31/02. In *Information Systems*, Elsevier, Amsterdam, The Netherlands, 2002.