Defining Strategic Dependency Situations in Requirements Elicitation

Antonio de Padua A. Oliveira [1, 2, 3]
Luiz Marcio Cysneiros [3]

[1] *UERJ – Universidade do Estado do Rio de Janeiro*
[2] *PUC-Rio – Pontificia Universidade Catolica do Rio de Janeiro*
[3] *York University, Toronto*

padua@ime.uerj.br
padua@inf.puc-rio.br
{apadua, cysneiro}@yorku.ca

Abstract

Recently, there has been a growing interest in the Agent-oriented paradigm to cope with the needs imposed by nowadays complex and networked systems. Developing Multi-Agent Systems (MAS) calls for addressing aspects such as interaction, autonomy, collaboration and pro-activeness. One way to cope with these needs is to have agency properties as well as intentionality in the center of the software development process. In this work a proposal is presented to bring intentionality and agency properties to the early stages of software development. The proposal is based on Strategic Dependency Situations (SDsituations) as a simple technique for helping requirements elicitation. Strategic Dependency Situations applies the Agent-Oriented approach based on intentionality to face the complexity of MAS developing.

1. Introduction

In the past, a large number of works have been written highlighting how important it is to have good requirements for system development. They mention errors, omissions, and also give emphasis to the high cost of late repairing of requirements errors [5], [8]. Facts about projects failures have also been stated pointing out to problems related to the credibility of software engineering process [5]. Nowadays, practitioners are giving more importance to requirements elicitation and are adopting mechanisms that were proposed by researchers in the recent past. We have also observed that recently there has been a growing interest in the Agent-oriented paradigm, a new way to cope with the needs imposed by nowadays complex and networked systems.

Developing Multi-Agent Systems (MAS) is currently the focus of various researches [1], [2], [4], [6], [14], [15]. MAS' modeling requires the usage of concepts and metaphors that reflect the way we understand the world; it also requires techniques to deal with a higher level of abstraction for understanding how software should behave and which characteristics they should present [17]. In order to achieve a good MAS development, software engineers need to understand, think and act, using Agent-Orientation concepts from the definition to the implementation of the system. The Agent-Orientation process starts with the usage of the concepts of intentionality that involves a large number of actors with opportunities and vulnerabilities [3], moreover, each actor has their own goals, beliefs, abilities, commitments, which are intentional in nature [17]. Therefore, intentionality should play a major role in the software development process.

Aside from intentionality, we need to consider agency properties, like autonomy, pro-activeness, sociability, adaptation, and interaction as well as collaboration, learning, and mobility [6], [15]. Consequently, there are various kinds and a lot of information that the requirements engineer must keep during the process because one simple requirement being lost may result in a very expensive repair in the future.

In this context, we propose Strategic Dependency Situations (SDsituations) as a simple, technique for helping requirements elicitation. Strategic Dependency Situations applies the Agent-Orientated approach based on intentionality [18] supported by the Lexicon Extended Language (LEL) a representation schema to help the elicitation of an application [9].

SDsituations uses intentionality as the backbone of a requirements process to deal with MAS development. Furthermore, SDsituations has been created to document requirements of strategic dependencies and show the chain of strategic interdependencies situations that exists in the organizational environment, SDsituations has features to be used in Requirements Management. The requirements engineer can take notes of traceability attributes [8] in a natural way ("on the fly"). Aside from that, the software collects the requirements evolution in a baseline [10].

We illustrate SDsituations Technique in one exemplar multi-agent system: "The Expert Committee System" (EC) [4], which is a Web-based system. An Expert Committee is a group of members convened by one Director-General (Coordinator) for the purpose of reviewing and making technical recommendations on a subject of interest to one organization or to one conference. A member of an expert committee is an expert appointed by the Coordinator to serve at a particular committee. Software agents should be introduced to the EC System in order to assist researchers (members) with time-consuming activities in the paper submission and reviewing processes. EC agents are software assistants, who represent actors in different roles of the conference such as: paper authors, reviewers, committee members, chairs, and coordinators.

This paper is based on initial work developed with other co-authors, and some of the examples are reused from that previous work [20].

The paper is organized as follows: In Section 2 we briefly describe the LEL approach, Scenarios Technique and i* Modeling Framework. In Section 3 we present our proposal: Strategic Dependency Situations. In Section 4 we present one example of our proposal, using the Expert Committee System, and finally in Section 5 we conclude and point out some future works and research issues.

2. The LEL Approach, Scenarios Technique and i* Framework

In this section we briefly introduce the Language Extend Lexicon (LEL) as an important approach for requirements elicitation and modeling. We also introduce a scenario technique, a well-known technique for requirements elicitation. Finally, we introduce i* modeling framework which is an agent-oriented modeling framework.

2.1 The LEL approach

Language Extend Lexicon (LEL) is a representation model of terms of the application language. LEL [9] is centered on a very simple idea: understanding the language of the problem without worrying about understanding the problem. The goal of the LEL is to represent words or sentences, called symbols, peculiar to the application context, which Leite [9] refers as the Universe of Discourse (UofD[1])

In the Figure 2.1, we show the elements of the LEL using a class diagram. LEL is composed by SYMBOLS. Each SYMBOL, or entry, is identified by a name or names (case of synonyms) and is represented by two descriptions. The first one, called NOTIONS, is the denotation of the symbol, equivalent to a description found in a dictionary. The second one, called BEHAVIORAL RESPONSES, is the connotation of the symbol, which describes the contextualization of that symbol in the UofD. The diagram shows that both notion and behavioral response refer to other symbols. The symbols in LEL are classified into four different types: STATE, VERB, OBJECT, and SUBJECT.

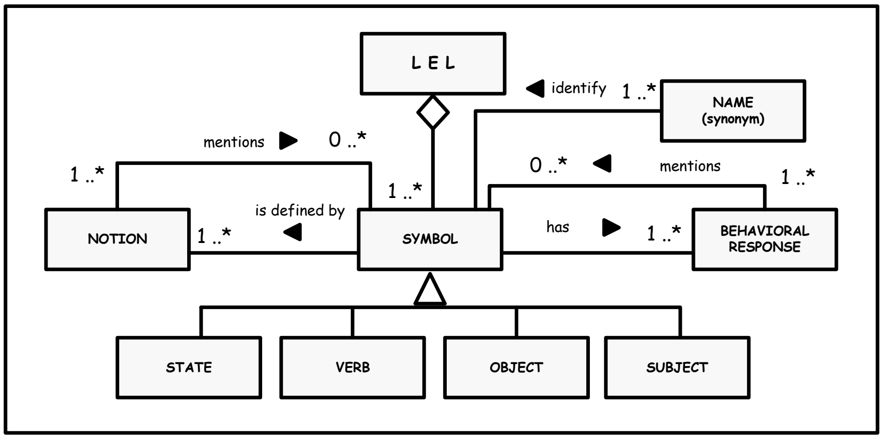


Figure 2.1 – LEL – class diagram

LEL is based on two principles in order to help the requirements engineer to elicit the symbols of the UofD. For describing LEL symbols the requirements engineer must follow (1) the circularity principle (also called "closure principle") and the (2) minimal vocabulary principle. The circularity principle states that we have to maximize the use of symbols when describing a symbol while the minimal vocabulary principle states that we have to minimize the use of words that are external to the Lexicon. These principles are essential to obtain a vocabulary of the application that is self-contained and highly connected [9].

2.2 The Scenarios Technique

A scenario is a structured description of one situation [11], which uses the language of the UofD. The situations occur in the real world. Eliciting scenarios that reflect

[1] *"The overall context in which the software will be developed and operated, the UofD includes all the sources of information and all the people related to the software. It is the reality reviewed by the set of objectives established."*

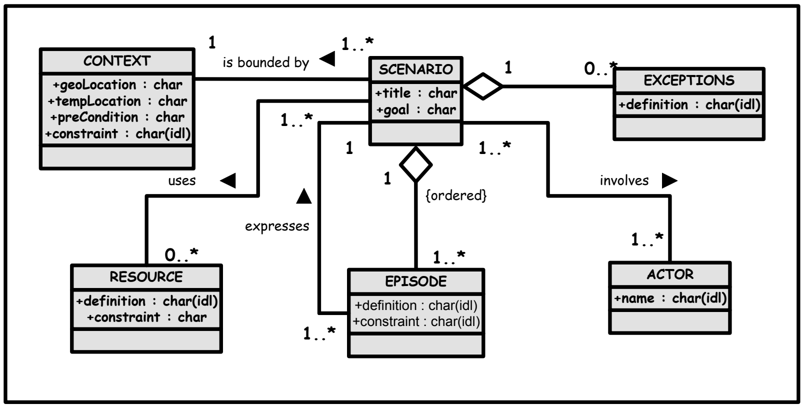


Figure 2.2 – Scenarios – class diagram

these situations needs the peculiar and most used words or phrases used in the UofD.

Situations have characteristics: they are concrete and have goals; they involve actors and need resources; they happen in a defined time and place, they may have restrictions which may either qualify the scenario or give some impositions. They are individually independent, interrelated and they may have alternative course represented by exceptions.

Figure 2.2 shows the relationship of components in the Scenario technique. Each scenario is identified by one name (title) and could have exceptions the scenario must consider. A scenario satisfies one goal and one context specifies its boundaries. They need resources and involve one or more actors. Scenarios are compound by episodes that define functional requirements.

2.3 The i* Modeling Framework

The i* modeling framework [16] models organizational contexts based on the dependency relationships among actors. The dependency relationship among actors helps the requirements engineer to elicit goals at early stages (Why do actors have these dependencies?).

I* Framework uses two models: the Strategic Dependency (SD) model and the Strategic Rationale (SR) model. The central idea of i* involves actors who depend on each other for goals to be achieved, for resources to be provided, for tasks to be performed, and for softgoals to be satisficed[2]. The SD model depicts the organizational context of the system as a network of dependency relationships among actors. This network consists of a set of nodes and links where each node represents an actor and each link map out one dependency between two actors. Thus, a dependency is a relationship in which one **actor** (the depender) depends on another **actor** (the dependee) to achieve a **goal**, to perform a **task**, to provide a **resource** or to achieve a **softgoal**; reflecting distinct types of freedom allowed by the relationship. The modeling process creates more details with the representation of the intentional relationships that are "internal" to actors. The SR models make the representation, in terms of process elements and the "internal rationale" (how? how else?), of the relationships behind the actors [16].

Figure 2.3 illustrates the components of the SD Model. Each dependency (strategic dependency), represented in the SD Model, is linked to two kinds of actors (dependers and dependees). A dependency has a degree (critical, committed or open) that reflects the importance of the dependency. Each dependency is classified as goal, softgoal, task, or as resource.

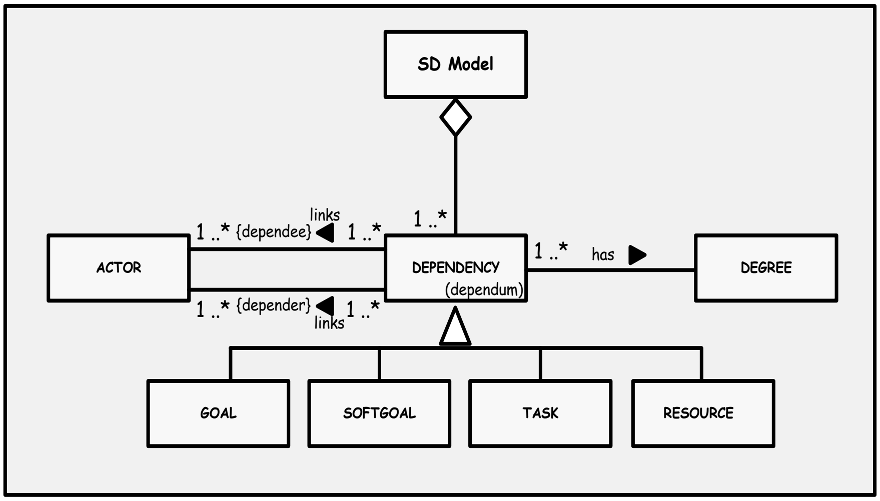


Figure 2.3 – The class diagram of SD model

3. Intentional Elicitation Process

[2] We use here the same notion used in [1] that an NFR can rarely be said to be satisfied. Goal satisficing suggest that the solution used is expected to satisfy within acceptable limits. The term satisfice was coined by Hebert Simon to express "good enough" alternatives.

In this section we propose an elicitation process based on activities as is shown in the SADT diagram [13] in Figure 3.1. The first step is the elicitation of UofD symbols. Based on this LEL the requirements engineering must define strategic dependency situations SDsituations. These SDsituations will be used to elicit and define scenarios. Finally the SDsituations and the scenarios will be used as front-end to develop i* models.

Diagram 3.1 shows "feedbacks" between (1) Elaborate LEL and (3) Define Scenarios and also between (2) Define SDsituations, (3) Define Scenarios and (4) Model Intentionality. Developing SD & SR models may demand changes in SDsituations as well as in scenarios definition. In its turn, developing scenarios may also demand LEL to be updated which may re-start the process.

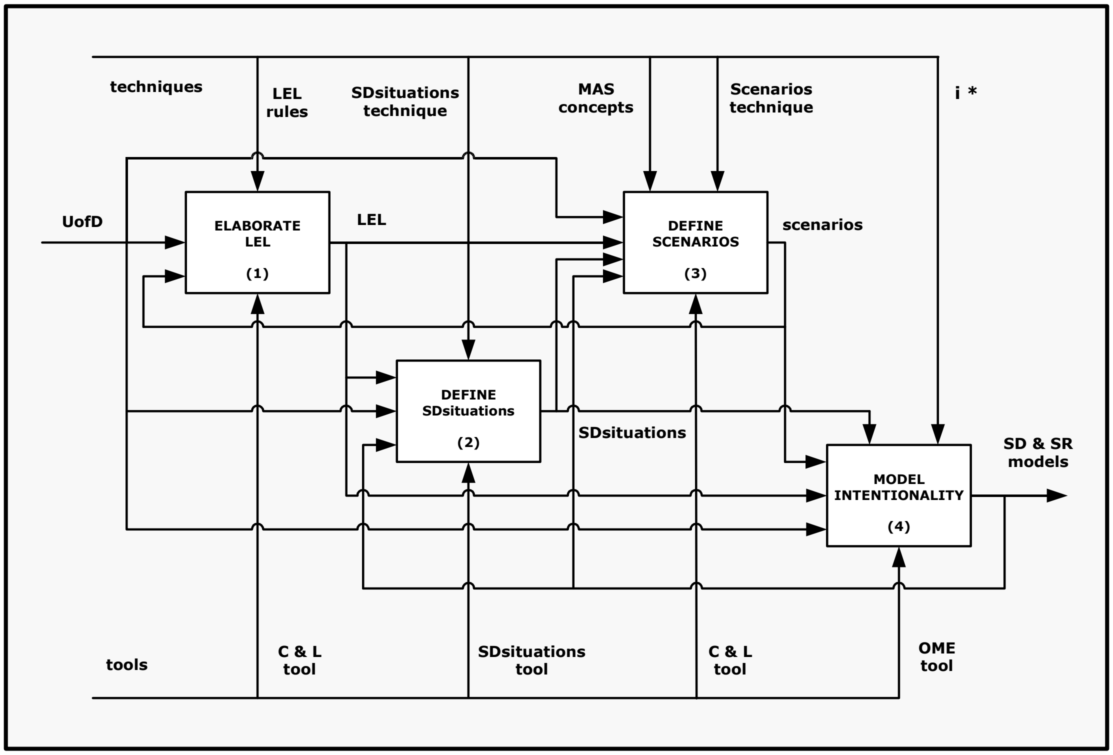


Figure 3.1 – SADT of the Intentional Elicitation Process

(1) ELICIT SYMBOLS

The requirements engineer needs to identify and understand what are the most peculiar and most used words or phrases in the UofD. By doing so the requirements engineer may familiarize himself with the relevant terms in the UofD and some of the semantics arising from these terms. Each symbol (word or phrase) must be completely defined into LEL as well as the attributes of traceability. The attributes indicated by Kotonya and Sommerville are: source, rationale, modeling information, implementation information and user's documents. The symbols' attributes required by LEL are: symbol's name and synonyms, notion, and behavioral response.

(2) DEFINE SDSITUATIONS

"One SDsituation is a structured representation of one Strategic Dependency situation." The situations of dependency occur in the organizational environment. The central idea of SDsituations is: each dependency link (goal, softgoal, task or resource) that involves actors is not isolated; it is part of one well defined situation of collaboration called one "strategic dependency situation" or one SDsituation. One SDsituation, is composed by one or more dependency elements, and each SDsituation can be identified separately from other SDsituations forming a chain of interdependencies. Interdependencies among SDsituations may be physical, logical or temporal. Identifying SDsituations and the chain of interdependencies among SDsituations has several advantages among them:

(a) Eliciting SDsituations before modeling is a better way of handling the complexity instead of dealing with all of the dependencies at the same time [12]. This is true because each SDsituation should be identified separately, although the engineer must elicit strategic interdependencies among the SDsituations.

(b) Validating requirements using one readable representation is useful because stakeholders feel more comfortable with models centered on natural language. The validation can be customized through the SDsituations applying more than one viewpoint (dependers or dependees viewpoints).

Table 1 –Describing elements of SDsituations

	GOAL	SOFTGOAL	TASK	RESOURCE
NAME <syntaxes>	Subject+BeVerb	Softgoal [topic]	Verb + object	Noun
DEFINITION	simple sentences	simple sentences	simple sentences describing subtasks	simple sentences describing components
CONSEQUENCES	simple sentences	simple sentences	simple sentences	simple sentences

(c) SDsituations can help the Requirements' Management keeping the traceability (backward-from & forward-to) during the elicitation process and keeping one baseline [10] in order to register the requirements evolution.

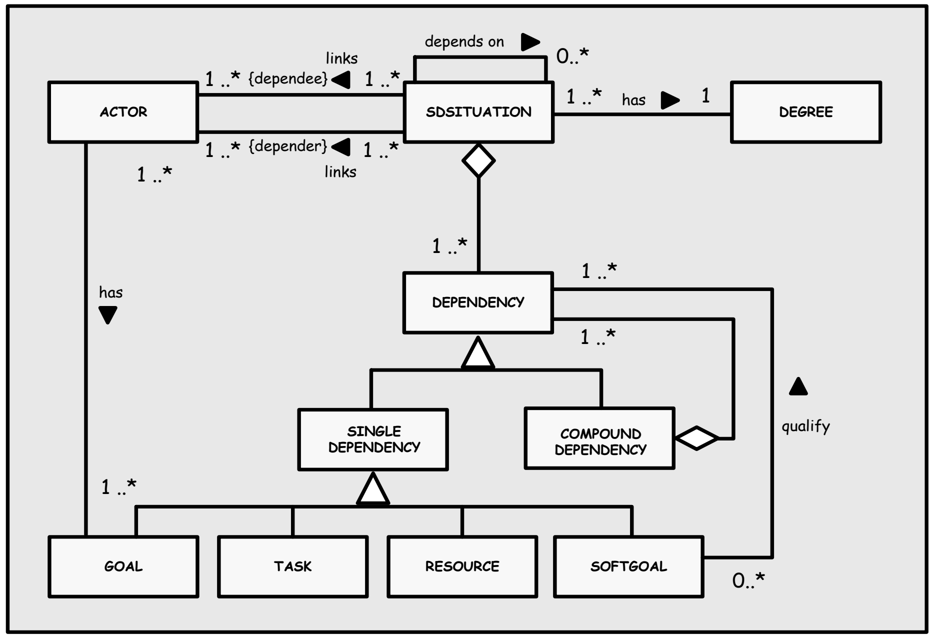


Figure 3.2 – Class Diagram of SDsituations

After registering most of the symbols, the requirements engineer must look into SDsituations and also elicit situations that have some dependency relationship with another situation. For example: An electoral process situation can only be performed if the candidates were defined previously. It means that we should identify two separate situations but one depends on the other critically: The **election process** depends on **candidates' definition**.

In Figure 3.2, one SDsituation reflects only one strategic dependency situation that may be formed by either one element (single dependency) or by a compound dependency. The compound dependency gives a recursive idea for the elements of SDsituations. Elements of dependency are classified as: GOAL, TASK, RESOURCE or SOFTGOAL. Moreover a softgoal may qualify an element of dependency; meaning that the element of dependency can not be excluded if there is a qualification. The elements of dependency are described like symbols of LEL with Name, Definition and Consequences. Table 1 shows the syntaxes, meanings and functions of the attributes. SDsituations should be written using the syntax <object + noun of situation>, e.g.: "Reviewers indication". We use i* elements' syntax used in [19].

The diagram portrait in Figure 3.2 also shows that actors (dependers and dependees) participate on SDsituations and each SDsituation has a degree of dependency [16]. The degrees may be: critical, committed or open.

SDsituations may depend on each other. There are three types of SDsituations interdependencies. It may be a **physical** dependency if one resource is prepared by one SDsituation and is needed by another SDsituation. It may be a **logical** dependency either when one or more SDsituations need the conclusion of other SDsituations to perform their initiation or when one or more SDsituations need the conclusion of other SDsituations to perform their conclusion. Last but not least, a dependency may also be **temporal**, either when one or more SDsituations need to wait some time after the beginning of another SDsituation or when one or more SDsituations need to wait some time after another SDsituation conclusion. Moreover, we may have more than one kind of interdependency at same time. Figure 3.3 shows temporal interdependencies.

Once again, the attributes of traceability: source, rationale, modeling, implementation, and user's documents should be informed.

(3) DEFINE SCENARIOS

The third activity is to describe scenarios in detail:

- Maximize the use of LEL symbols when describing each scenario. Describe: actors, goals, resources, episodes, and constraints.
- Give emphasis to elements which have agency properties, like autonomy, pro-activeness, sociability, adaptation, and interaction as well as collaboration, learning, and mobility.

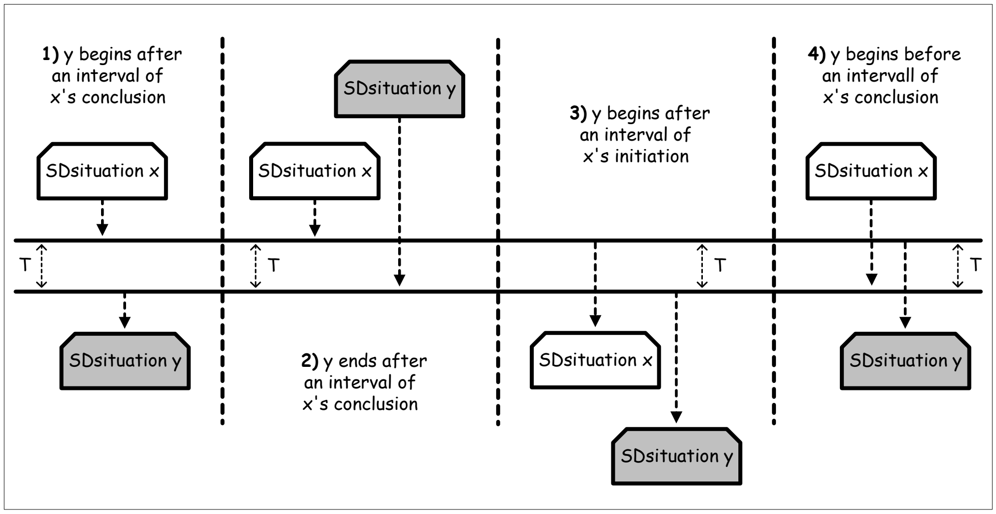

1) y begins after an interval of x's conclusion
SDsituation x
SDsituation y
2) y ends after an interval of x's conclusion
SDsituation y
SDsituation x
3) y begins after an interval of x's initiation
SDsituation x
SDsituation y
4) y begins before an intervall of x's conclusion
SDsituation x
SDsituation y
T
T
T
T

Figure 3.3 – Illustrations of SDsituations interdependencies

(4) MODEL INTENTIONALITY

The fourth activity is to create i* models. The i* framework [16] models organizational contexts using the relationships among actors, that are the same components of the scenario approach. In the models created with i*, each actor can be autonomous [18]. The dependency relationship with actors guides the requirements engineer to identify individual and shared goals (Why do actors have these dependencies?). In our case study for example, while modeling the Expert Committee Application, we had to answer "why" questions such as: Why do authors submit articles? Why do reviewers review articles? Why do committee's members vote on conflicts of reviews? Why do coordinators indicate reviewers? Why do researchers accept to be reviewers?

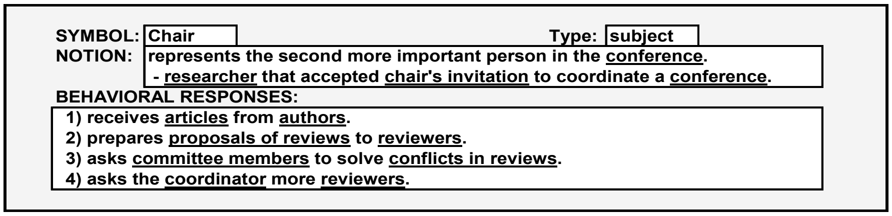
External relationships among actors are expressed in the Strategic Dependency (SD) model. Internal relationships among the intentional elements within an actor's reasoning are expressed in the Strategic Rationale (SR) model. Rationales are modeled through means-ends relationships, task decompositions, and softgoal contributions.

4. Example of using SDsituations

(1) Elaborate LEL:

a) Elicit symbols

- Identify the SYMBOLS (words or sentences) that are peculiar to the social environment.
- Use one or more technique for fact gathering (e.g.: interviews, observation, document reading) [7].
- Classify symbols in LEL as: OBJECT, SUBJECT, VERB, and STATE. The most reliable information sources in UofD are documents and people. First we have to identify the right actors and they will be defined as subjects in the LEL. In our example we elicited as subjects some actors. The ACTORS elicited were: Author, Chair, Reviewer, Coordinator, and Committee Member. Some symbols elicited were: abstract, acceptance due time, article, author information, author institution, conference, co-author, institution.



SYMBOL: Chair **Type:** subject
NOTION: represents the second more important person in the conference.
- researcher that accepted chair's invitation to coordinate a conference.
BEHAVIORAL RESPONSES:
1) receives articles from authors.
2) prepares proposals of reviews to reviewers.
3) asks committee members to solve conflicts in reviews.
4) asks the coordinator more reviewers.

Figure 4.1 – LEL subject symbol definition – from [20]

b) Describe notions and behavioral responses

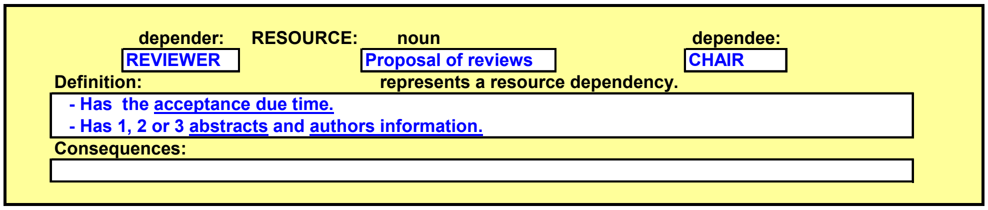
- Maximize the use of LEL symbols when describing.
- Follow both the circularity principle and the minimal vocabulary principle.

Figure 4.1 shows an example of LEL symbol. Note that: words or sentences are underlined because they are LEL symbols.

(2) Define SDsituations:

a) Identify elements of SDsituations

- Actors of SDsituations are LEL's symbols that are classified as subject. Figure 4.1 shows an example: "chair".
- Dependency elements are: goal dependency, task dependency, resource dependency or softgoal dependency.



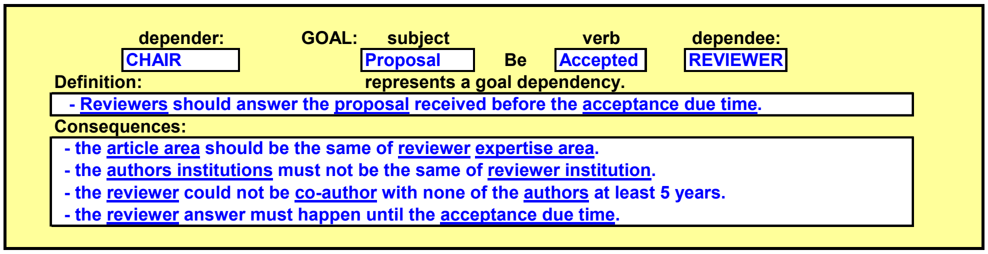
depender: REVIEWER — RESOURCE: noun: Proposal of reviews — dependee: CHAIR

Definition: represents a resource dependency.

- Has the acceptance due time.
- Has 1, 2 or 3 abstracts and authors information.

Consequences:

Figure 4.2 - Example: definition of a resource dependency



depender: CHAIR — GOAL: subject: Proposal — Be — verb: Accepted — dependee: REVIEWER

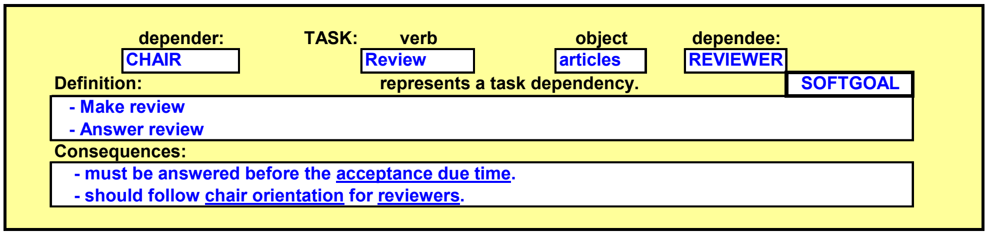
Definition: represents a goal dependency.

- Reviewers should answer the proposal received before the acceptance due time.

Consequences:

- the article area should be the same of reviewer expertise area.
- the authors institutions must not be the same of reviewer institution.
- the reviewer could not be co-author with none of the authors at least 5 years.
- the reviewer answer must happen until the acceptance due time.

Figure 4.3 - Example: definition of a goal dependency



depender: CHAIR — TASK: verb: Review — object: articles — dependee: REVIEWER — SOFTGOAL

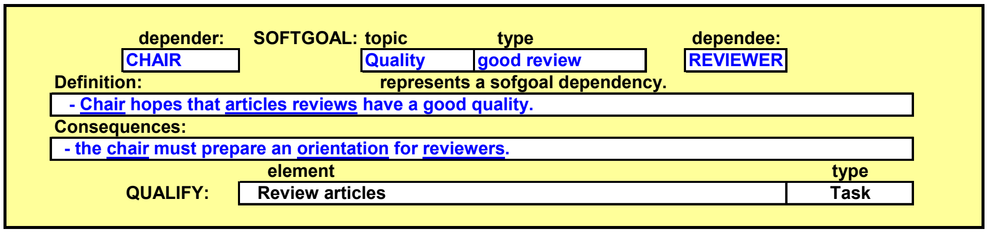
Definition: represents a task dependency.

- Make review
- Answer review

Consequences:

- must be answered before the acceptance due time.
- should follow chair orientation for reviewers.

Figure 4.4 - Example: definition of a task dependency



depender: CHAIR — SOFTGOAL: topic: Quality — type: good review — dependee: REVIEWER

Definition: represents a sofgoal dependency.

- Chair hopes that articles reviews have a good quality.

Consequences:

- the chair must prepare an orientation for reviewers.

	element	type
QUALIFY:	Review articles	Task

Figure 4.5 - Example: definition of a softgoal dependency

- For each actor in the LEL:

  - Find out which other actor this actor depends on; use the view-point of depender. By using the view-point of dependee, one can find out the same dependencies. Other symbols classified as subject appearing either on the notions or in the behavioral responses may indicate such dependencies.

  - Prepare the definition of the elements. Figure 4.2, 4.3, 4.4 and 4.5 are examples of dependency elements.

Figure 4.1 shows the definition of the symbol CHAIR. There are other actors being mentioned in the figure like: RESEARCHER, AUTHOR, REVIEWER, COMMITTEE MEMBER, and COORDINATOR that are linked with CHAIR in some dependency. Examining the symbol CHAIR we can observe that there are symbols which are classified as objects in LEL, e.g.: PROPOSAL and CHAIR INVITATION that may be a resource dependency. For example, focusing on CHAIR (dependee) and REVIEWER (depender) we defined "Proposal of Reviews" (synonym of Proposal) as a resource dependency. Note again that: words or sentences, defined as elements of SDsituations are also underlined because they are LEL symbols.

Examining CHAIR (dependee) and REVIEWER (depender) symbols, we defined "Proposal Be Accepted" (Figure 4.3) as a goal dependency because “A goal is a condition or state of affairs in the world that the actor would like to achieve” [16]. In the above interaction CHAIR hopes that the REVIEWERS accept the proposals.

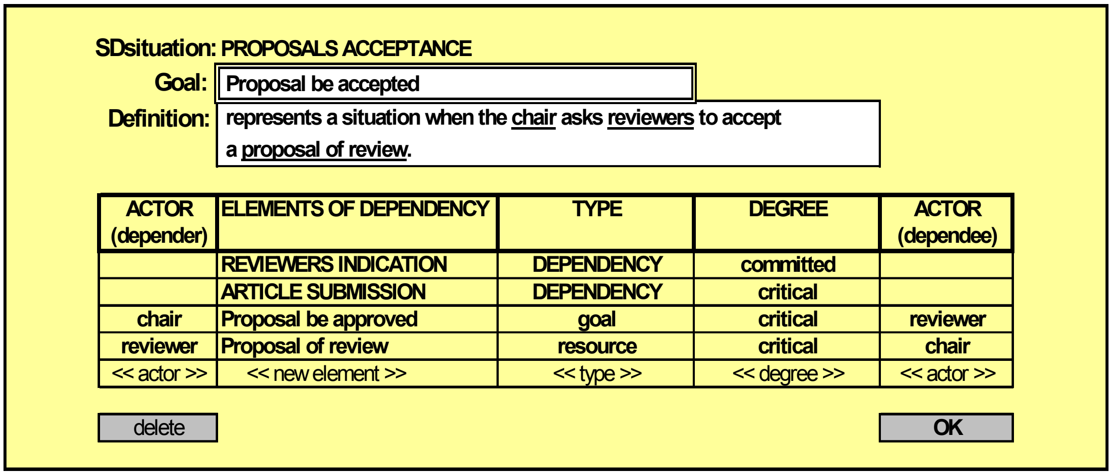
- We defined “Review Articles” (Figure 4.4) as a task dependency because: chair depends on reviewers to perform reviews and in Figure 4.5 is shown that there is a softgoal because the review should be done following some concerns of quality.

b) Compound SDsituations

One SDsituation can be compounded of one or more dependencies. The requirements engineer should realize theses dependencies are together in the same situation. For example: the COORDINATOR depends on the RESEARCHER to achieve the goal “Invitation Be Accepted” but, on the other hand, almost at the same time the RESEARCHER depends on COORDINATOR for getting the “invitation”, which is a resource dependency.

c) Find out interrelationships among SDsituations

- Identify dependencies (temporal, physical or logical) among SDsituations.
- Although one SDsituation can be compounded of one or more dependencies, one dependency may be another SDsituation. For example: when the COORDINATOR is going to indicate REVIEWERS to CHAIR, in the SDsituation REVIEWERS INDICATION, the SDsituation RESEARCHER INVITATION must be concluded.



SDsituation: PROPOSALS ACCEPTANCE

Goal: Proposal be accepted

Definition: represents a situation when the chair asks reviewers to accept a proposal of review.

ACTOR (depender)	ELEMENTS OF DEPENDENCY	TYPE	DEGREE	ACTOR (dependee)
	REVIEWERS INDICATION	DEPENDENCY	committed	
	ARTICLE SUBMISSION	DEPENDENCY	critical	
chair	Proposal be approved	goal	critical	reviewer
reviewer	Proposal of review	resource	critical	chair
<< actor >>	<< new element >>	<< type >>	<< degree >>	<< actor >>

Figure 4.6 – SDsituation with two single dependencies that depends on two SDsituations

- Another example is illustrated in Figure 4.6. The SDsituation PROPOSAL ACCEPTANCE is compounded of two SDsituations: ARTICLE SUBMISSION and REVIEWERS INDICATION and two single dependency elements: “Proposal of Review” and “Proposal be Accepted”. The SDsituation PROPOSAL ACCEPTANCE demands the conclusion of ARTICLE SUBMISSION and REVIEWERS INDICATION.

(3) Define Scenarios:

a) Create scenarios

- Define one scenario for each SDsituation.
  - Although it is possible that one scenario have more than one SDsituation, the usual is to have one scenario for one SDsituation.

b) Write the scenarios

- Maximize the use of LEL symbols when describing each scenario. Describe: actors, goal, resources, episodes, and constraints. Give priority in the representation of agency properties elements.

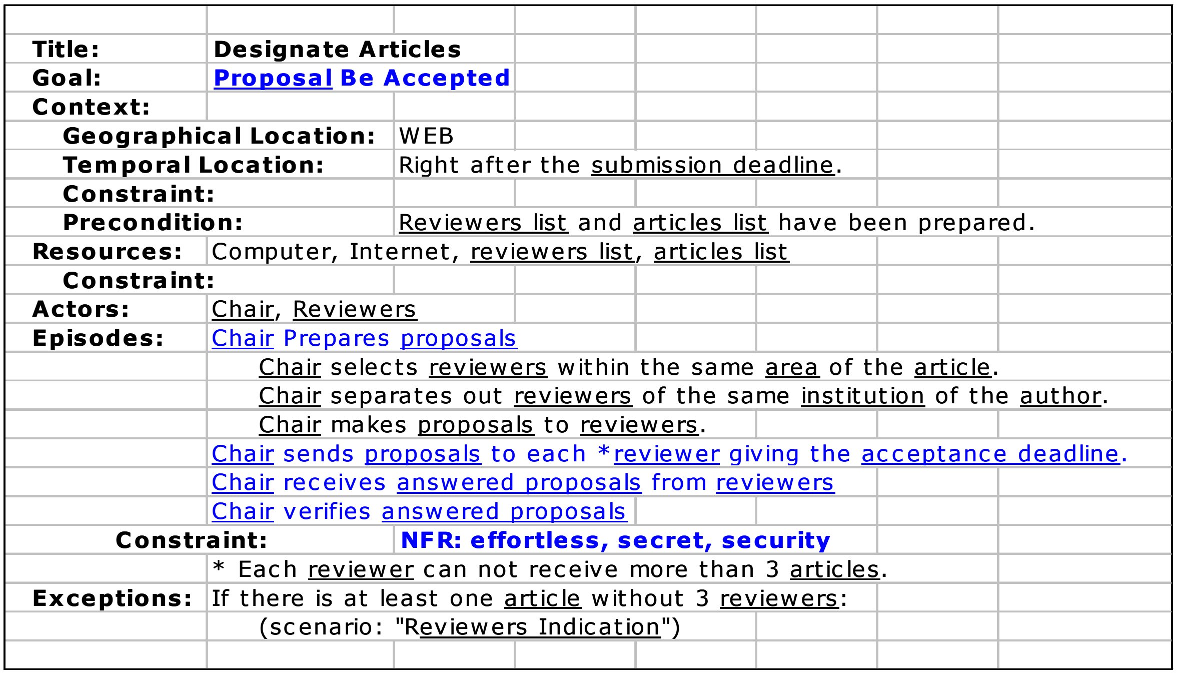
- Choose the goal’s name considering the best dependency element. The first choice is a goal dependency, the second is a task, and the last is a resource dependency. Softgoal dependency can not give the goal of scenario because a softgoal is always together another element. Table 2 shows our suggestion for choosing the name of the scenario’s goal.

Table 2 – Suggestion of scenario goal’s name

DEPENDENCY ELEMENT	SCENARIO: GOAL’S NAME
1) Goal dependency	Same name of the dependency
2) Task dependency	“task’s name” be performed
3) Resource dependency	“resource’s name” be received

- Pick up actors from SDsituation. Actors in SDsituations will be the same actors in scenario definition.

- Context does not exist in SDsituation, so you should include this information in scenario.



Title:	**Designate Articles**
Goal:	**Proposal Be Accepted**
Context:	
Geographical Location:	WEB
Temporal Location:	Right after the submission deadline.
Constraint:	
Precondition:	Reviewers list and articles list have been prepared.
Resources:	Computer, Internet, reviewers list, articles list
Constraint:	
Actors:	Chair, Reviewers
Episodes:	Chair Prepares proposals
	Chair selects reviewers within the same area of the article.
	Chair separates out reviewers of the same institution of the author.
	Chair makes proposals to reviewers.
	Chair sends proposals to each *reviewer giving the acceptance deadline.
	Chair receives answered proposals from reviewers
	Chair verifies answered proposals
Constraint:	**NFR: effortless, secret, security**
	* Each reviewer can not receive more than 3 articles.
Exceptions:	If there is at least one article without 3 reviewers:
	(scenario: "Reviewers Indication")

Figure 4.7 – Scenario definition: Designate Articles – from [20]

- Preconditions you should list all resources prepared in previous SDsituations needed by the current SDsituation.
  - "Reviewer List" was prepared in REVIEWERS INDICATION and "Article List" was prepared in ARTICLE SUBMISSION.
  - Include also all material resources that are necessary in scenario (e.g. internet, computer).

- Episodes describe, in functional terms, the details of scenario.
  - Write sentences showing the tasks performed by actors in scenario. Some tasks may appear inside of a task dependency in SDsituation, some tasks may appear in symbols (behavioral response) and others you must define considering agency properties as interaction.

- Constraints mention softgoals (NFR) that must be considered and describe impositions of the process. In this scenario the stakeholders want effortless, secret and security, as NFR. And has the imposition: "Each reviewer can not receive more than 3 articles for review".

- Exceptions mention alternatives in scenario definition. For example, if there are not enough reviewers another scenario must be executed.

Figure 4.7 portrays the scenario Designate Articles. This scenario depicts what is involved in designating article to be reviewed by reviewers. Note that words or sentences are underlined because they are LEL symbols.

(4) Model Intentionality

a) Create SD model

- Use elements from SDsituations.
  - Start using SDsituation that do not have interdependencies. The mapping is direct; all elements of SDsituation have only one representation in SD model. In our example, using the SDsituation: Researcher Invitation in Figure 4.5, we can create two actors: coordinator and researcher in the SD model and put two dependencies. Invitation Be Accepted is a goal dependency and invitation is a resource dependency. You should choose the right direction of the dependency. The result appears in SD model, Figure 4.8, upper in the left side of the diagram.
  - Continue modeling; use SDsituations which have 1 (one) interdependency after, model the SDsituations with 2 (two) interdependencies, and so on. In our example, using the SDsituation: Proposals Acceptance in Figure 4.6, we can create two actors: chair and reviewer in the SD model and put two dependencies. Proposals Be Accepted is a goal dependency and Proposal of Review is a resource dependency. You should again choose the

right direction of the dependency. The result appears in SD model, Figure 4.8, down in the left side of the diagram.

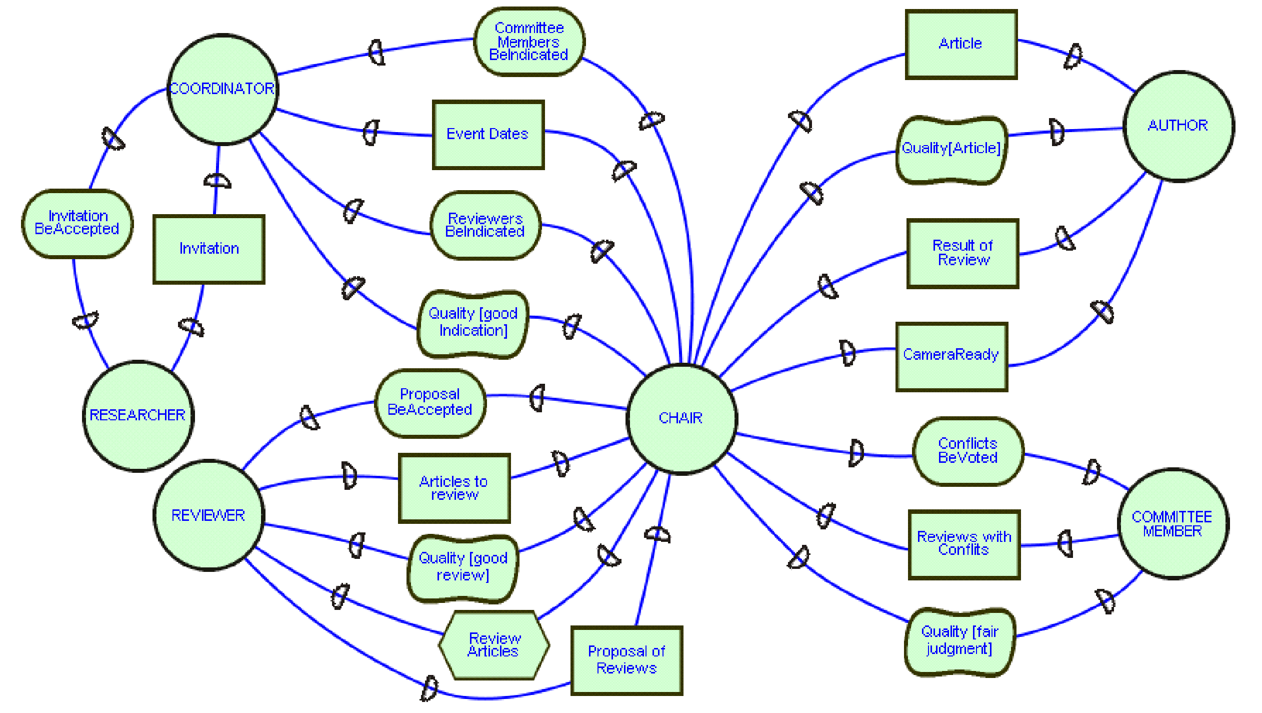


Figure 4.8 – SD Model of Expert Committee - adapted from [20]

Figure 4.8 shows the SD model used to represent the example (EC - Expert Committee). The SD model illustrates the relationships among several actors, although, in the SR model in this paper, we only focus on the relationship between chair and reviewer. The SD model shows that the CHAIR depends on the REVIEWER to achieve the goal "Proposal Be Accepted", to achieve the softgoal "Quality [GoodReview]" and to perform the task "Review Articles" furthermore; the reviewer depends on the chair to get the resource "Articles to Review". These dependencies were created using another SDsituation named: Articles Review that is not shown because lack of space in the paper.

b) Create SR model

Figure 4.9 portrays the SR model involving the actors CHAIR and REVIEWER.

- Show goals, tasks, resources and softgoals like intentional elements inside of the boundary of the actor (dot-dashed circle). Pick up goals, tasks, resources and softgoals from scenarios specifications and from SDsituations. Connect the elements using intentional links (means-ends, decomposition, and contribution).

- The same dependencies between the actors that appeared in SD model will appear here again connecting the actors; these elements appear outside of the boundaries of the actors, but now, in SR model, the connections will appear connecting the elements inside the actor's boundary.

- Identify what are the main goals for each actor. The SDsituations and scenarios provide the goals but the engineer should represent the hierarchy into the correct way.

- When you are going to draw the SR model defining two actors, you should find out all goal dependencies between these two actors and choose the hierarchy of them. In the example: Although Proposals Be Approved, Reviewed Articles Be Received and Conflicts Be Solved came from SDsituations, we had to create a new goal Articles Be Reviewed and a new task Manage Review.

- Using scenarios:

- High level episodes will map tasks and they will be mapped as "means-end" connections to the goal.
- Alternative episodes will be mapped as alternatives in the SR model, if there is more than one way to achieve a goal (means-end link to a goal).
- Low level episodes will be sub-tasks connected using a decomposition link.

- Resources used in one episode will be mapped either as a resource needed by the task (should be connected using a decomposition link) or as a resource dependency when this is an exchange between another actor and an agent, see the SDsituation that appear the resource dependency.

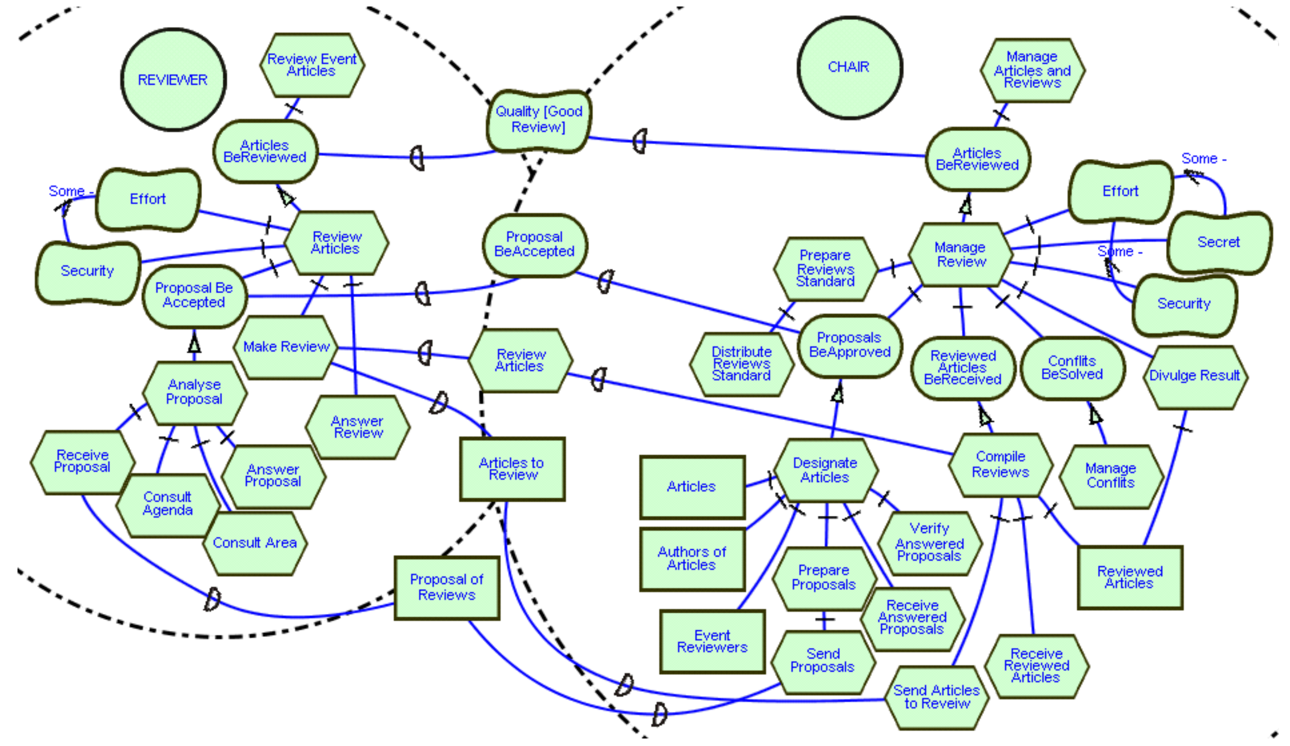


Figure 4.9 – SR model: Reviewer and Chair - adapted from [20]

- Quality attributes such as performance and security (NFRs), will be softgoals and they should be connected to tasks using a contribution link. However, some time, softgoals will have a decomposition link to a task. This happens whenever the softgoal is essential do perform the task adequately. We brought the softgoals: Effort, Secret and Security from scenarios Designate Articles, Divulge Reviews, and Manage Conflicts.

Take for example chair when the CHAIR is dealing with REVIEWERS, the main task is to "Manage Articles and Reviews", which has only 1 (one) decomposition, the goal "ArticlesBeReviewed". This association, by decomposition, means that the goal is one part of the task and only if the CHAIR achieves the goal, the task should be concluded. Alternatively, the model shows a previous softgoal "Quality[GoodReview]" meaning that the CHAIR depends on the REVIEWER. In the model it is also represented that the CHAIR has the task "PrepareReviewsStandard", to give the quality specifications and directions, and that the chair needs the softgoals: "Secret, Security and Effortless" for the task, these softgoals were elicited in one interview of scenario Designate Articles validation. In the model it is shown that the softgoal "Security" contributes (some -) "negatively" to the softgoal "Effort" because probably the user will have to enter with a password for this process. Showing the softgoals in the SR model the software engineer has the information about these concerns must be operationalized.

5. Conclusions and Future Works

In this paper we show an approach to bring intentionality and agency properties to the early stages of software development. The proposal is a lexicon oriented elicitation strategy that leads to i* models giving focus on Strategic Dependency Situations (SDsituations) which applies the Agent-Oriented approach based on intentionality to face the complexity of MAS developing.

Our proposal helps the identification of the i* representation elements as well as provide support to gather traceability information. Moreover, we have detailed a proof of concept using the Expert Committee examplar. Leite and Franco [9] showed that it was possible to discover several Kaos [http://www-di.inf.puc-rio.br/~julio/Slct-pub/lel.pdf] representation elements through an analysis of LEL representations, but noted that they failed to discover goals, since the LEL is a nonintentional representation. In this work, we have tackled the problem from a different angle; by focusing on dependency analysis we found out a way of helping the elicitation of these dependencies and as an important side effect we have general heuristics to discover goals.

Our results are initial, but we believe they are positive ones. Anchoring the construction of i* models on well defined processes for elicitation is a necessity. We believe that our approach, geared towards analysis of dependency based on the lexicon is a possible solution to the problem of i* models elicitation. Future works point to investigate more detailed traceability features to be implemented and how we could help requirements management using from SDsituations.

Acknowledgements: Thanks to Julio Cesar S. P. Leite for his valuable review and comments of this paper.

6. References

[1] Castro, J.; Kolp, M.; Mylopoulos, J. (2002) "Towards Requirements-Driven Information Systems Engineering: The Tropos Project." In: The 13th international conference on advanced information systems engineering, Oxford: Elsevier Science Ltd, v.27, n.6. p. 365-389.

[2] Choren, R and Lucena, C “The ANote Modeling Language for Agent-Oriented Specification”. SELMAS 2004: 198-212.

[3] Cysneiros, L. M. and Yu, E. Requirements Engineering for Large-Scale Multi-Agent Systems Book chapter in Software Engineering for Large-Scale Multi-Agent Systems – Research Issues and Practical Applications. A. Garcia, C. Lucena, F. Zambonelli, A. Omicini and J. Castro (eds.) LNCS 2603, Springer Verlag. 2003. (Revised and extended version of [SELMAS02]).

[4] Deloach, S. et al. Multiagent Systems Engineering. International. In: Journal of Software Engineering and Knowledge Engineering, 11(3):231--258, 2001.

[5] Dennis, Alan; Wixon, Barbara and Tegarden, David; Systems Analysis and Design – An Object-oriented Approach with UML. John Wiley &Sons, Inc. 2002.

[6] Garcia, A. F.; Lucena, C. J. P.; Cowan, D. D.; Agents in Object-Oriented Software Engineering; In: Software, Practice & Experience, Elsevier, vol. 34, Issue 5, pp. 489-521, May 2004.

[7] Goguen, J.A. and Linde, C. - Techiques for Requirements Elicitation, In Proceedings of the First IEEE International Symposium on Requirements Engineering, San Diego, Ca, IEEE Computer Society Press - 1994, pp 152-164.

[8] Kotonya, Gerald.; Sommerville, Ian. Requirements Engineering – Processes and Techniques – 1998 – John Wiley & Sons Ltd. England.

[9] Leite, Julio C. S. P.; Franco, Ana P. M. Client A Strategy for Conceptual Model Acquisition, Proceedings of the International Symposium on Requirements Engineering, IEEE Computer Society Press, San Diego (1993), pp. 243-246.

[10] Leite, Julio C. S. P.; Oliveira, A. de Padua A.; Client Oriented Requirements Baseline, Proceedings of the Second International Symposium on Requirements Engineering, RE95, IEEE Computer Society Press, 1995, pp. 108-115.

[11] Leite, Julio C. S. P., Hadad, G., Doorn, J., Kaplan, G. A Scenario Construction Process - RE Journal: Vol. 5, N. 1, Pags. 38 - 61, (2000), Springer-Verlag London Limited.

[12] Miller, George; The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information by originally published in The Psychological Review, Harvard University - 1956, vol. 63, pp. 81-97

[13] Ross, D. T.; "Structured Analysis (SA): A Language for Communicating Ideas", IEEE Transactions on Software Engineering Vol. SE-3 NO. 1, 1977.

[14] Silva, V.; Lucena, C. "From a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language", In: Sycara, K., Wooldridge, M. (Edts.), Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers, 2004.

[15] Wooldridge, Michael, An Introduction to Multi-Agent Systems, John Wiley and Sons Limited: Chichester, 2002, ISBN 047149691X

[16] Yu, E. Modelling Strategic Relationships for Process Reengineering. PhD Thesis, Graduate Department of Computer Science, University of Toronto, Toronto, Canada, 1995, pp. 124.

[17] Yu, E. Agent-Oriented Modelling: Software versus the World In Agent-Oriented Software Engineering AOSE-2001 Workshop Proceedings, Montreal, Canada -. LNCS 2222.

[18] Yu, E. Agent Orientation as a Modelling Paradigm Wirtschaftsinformatik. 43(2) 2001.(123-132.

[19] You Zheng, Leite, J C, Yu, E, “An Assessment of i* Modeling and Evaluation Using the London Ambulance Service Case Study”, Department of Computer Science, University of Toronto.

[20] Oliveira, A. P. A.; Cysneiros, L. M.; Leite, J. C. S. P.; Figueiredo, E. M. L.; Lucena, C. J. P.; “Integrating Scenarios, i*, and AspectT in the Context of Multi-Agent Systems” CASCON 2006 the 16th Annual International Conference on Computer Science and Software Engineering **- submitted**.