A Survey of Good Practices and Misuses for Modelling with *i** Framework

Ilca Webster 1, Juliana Amaral 2, Luiz Marcio Cysneiros1
1 Department of Mathematic and Statistics - Information Technology Program
York University – Toronto – Canada
(lwebster, cysneiro@yorku.ca)
2 Pontificia Universidade Catolica de Minas Gerais – Departamento de Ciencias da Computacao
Belo Horizonte - Minas Gerais – Brasil
(ju_amaral@yahoo.com)

Abstract: The ***i**** framework has been increasingly used by the requirements engineering community. However, good practices are not always followed. Many papers have presented the use of ***i**** constructs in such ways that results are not coherent with their original specifications. Some cases attempt to adapt the framework to specific needs by misusing its elements. In other situations, misuses are due to wrong interpretations of ***i**** syntax and semantic. Unfortunately, many of these misuses may lead to wrong interpretations of the models by other people using ***i****. This work carried out a survey on several published papers to collect both good practices and misuses of the framework. It aims to help requirements engineers to use the ***i**** framework in its full capacity.

1. Introduction

In more recent years requirements elicitation has been considered a critical part of the software development process. Its importance directly relates to the degrees to which software systems meet the purposes for which they were intended. A software system built on poor requirements is prone to failure. Since requirements elicitation involves dealing with organizational, managerial, economic and social issues besides technical concerns, richer frameworks to support this activity have been increasingly explored such as KAOS [1], GBRAM [2] and *i** [3].

In more complex application domains, humans, hardware, and software interact in much more intricate ways than in conventional systems which automate routine tasks. Critical factors in the successful development of complex systems are the understanding of users' needs and intentions as well as how technologies might alter their relationships and how these technologies can facilitate negotiations and communicate these needs to requirements engineers.

The emerging agent/goal-oriented paradigm conceptualizes software as being proactive and exhibiting autonomy and sociality. Agent oriented methodologies can offer the highest level of abstraction needed for this new conception of software. Among these methodologies, the *i** modelling framework has been used by a growing community to support early and late requirements modelling.

i* focuses on strategic relationships among intentional agents in organizational settings and provides features to assist in describing processes (modelling) and directing changes (reengineering) [3]. It is composed by two models, named Strategic Dependence (SD) and Strategic Rationale (SR). The SD model describes a particular configuration of dependency relationships among organizational actors while the SR model describes actors' rationales about embracing specific configurations.

Although based on the use of few constructs, ***i**** carries complex semantics. Besides a strong comprehension of its main models and components, in-depth attention is needed to different levels of details. Minor misuses such as modelling goals as tasks, or not using the proper graphical notation of an element, prevents a diagram from showing accurate details of a system and or processes. This, in turn, renders the diagram insufficient of being a base for more refinements, extensions or evaluations of processes. It may also present problems for other readers trying to understand the model, because they may be referencing the original framework proposed by Yu [3].

The objective of this paper is to present good practices and misuses when applying the ***i**** framework to support requirements modelling. It is based on surveys of different studies using the framework [4], [5], [6], [7], [8], [9], [10], [11], [12], [13] and presents a summary of the most frequent and important cases of good practices and misuses. It aims to help requirements engineers to achieve better outcomes when using ***i**** framework as well as to facilitate requirements engineers who are starting to use ***i****.

This paper is organized in 5 sections. This introduction is followed by a briefly introduction to the ***i**** framework. Section 3 presents ***i**** framework good practices with their respective discussions. Section 4 presents misuses of the ***i**** framework together with the good practices that should have been followed to avoid them. Section 5 summarizes our work.

2. The i* framework

The i* framework [3] models relationships among intentional agents in organizational settings and provides features to assist in describing processes (modelling) and directing changes (reengineering). It is composed by two models, named Strategic Dependence (SD) and Strategic Rationale (SR).

The Strategic Dependency (SD) model depicts a process as a network of dependency relationships among actors. Actors can be roles, positions and agents. In the context of the i* framework, actors refer to generic entities that have intentionality. To reflect different degrees of concreteness of agency, the concepts of roles, positions and agents are defined as specializations of actors. Actors may be abstract (roles defining responsibilities), concrete (agents – human and non-human individuals or classes with specific capabilities), or other organizational constructs (e.g., positions which package a number of roles together to be assigned to a single concrete agent).

In i*, a dependency is a relationship in which one actor (the depender) depends on another actor (the dependee) for something to be achieved (the dependum). A dependum can be a goal, task, resource, or softgoal, reflecting the types of freedom allowed by the relationship. A goal dependency is one in which one actor depends on another to bring about a certain condition or state in the world, while the depended actor (the dependee) is free to, and is expected to, make whatever decisions are necessary to achieve the goal. Thus, it also indicates that one actor does not care how the other actor will achieve this goal.

The Strategic Rationale (SR) model describes actors' rationales about embracing specific configurations. It models the intentional relationships that are "internal" to actors, in terms of process elements and the rationale behind them. The generic notion of actor may be also differentiated into agents, roles and positions. Rationales are modelled through means-ends relationships, softgoal contributions and task decompositions.

A means-end relationship 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 generally expressed in the form of a task which represents a particular way of doing something.

A softgoal is a particular type of goal used in i* to model quality attributes for which there are no pre-established criteria for satisfaction. Instead, actors may judge that these attributes are sufficiently met ("satisficed") on a case-by-case basis.

A task decomposition link models a task in terms of its decomposition into its subcomponents. A task can be decomposed in four types of subcomponents: subgoals, subtasks, resources or softgoals. The components in a task decomposition are only those that are strategically significant for the the actor.

The ***i**** framework is based on these few constructs but its semantics is very complex. Modelling with it requires a strong comprehension of its main models and components as well as an in-depth attention to the different levels of details necessary to provide effective comprehension of the overall subject being modelled. The following sections discuss good practices and most common misuses of this framework.

3. *i** Framework Good Practices

This section presents examples that illustrate good practices related to the use of the ***i**** framework. Comments on how to successfully use the framework to achieve the specific goals associated with the examples are also included.

3.1 Modelling the contributions from a softgoal or task

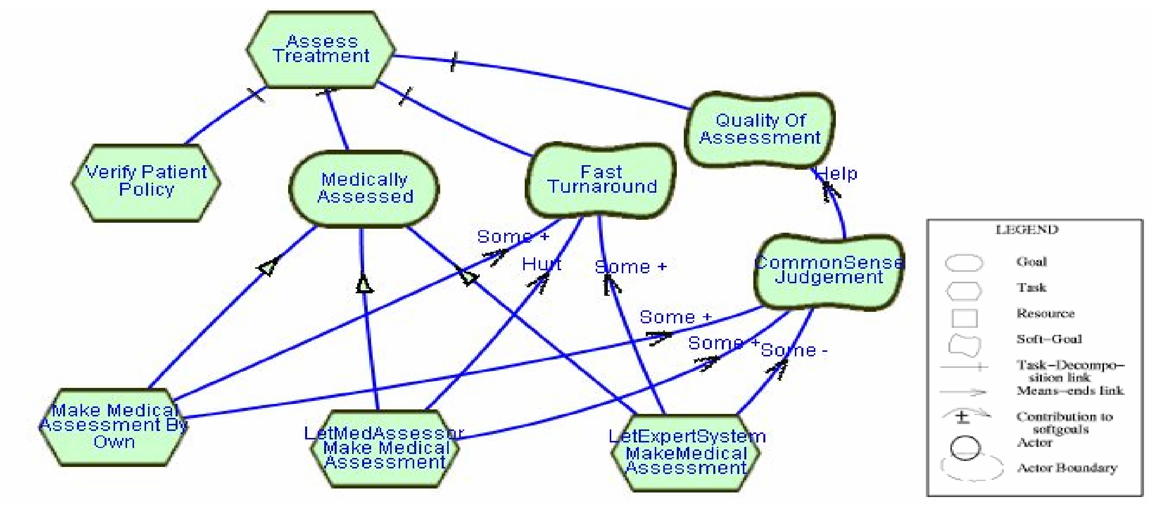


Figure 1

Softgoals have to be evaluated through a qualitative reasoning. This can be carried out using contribution links among the softgoals. The semantics of the links are based on the concept of satisficing [14], which means a softgoal can be satisfied within acceptable limits. Contribution links enable the decomposition of softgoals up to the point where the operationalizations to this softgoal are reached (i.e., the goals are no longer "soft") [15].

Therefore, designers should model the contributions from a softgoal or a task to a more abstract softgoal with the use of a contribution link. Figure 1 depicts a situation where the softgoal "Common Sense Judgement" contributes to the more abstract softgoal "Quality of Assesment". The tasks "Make Medical Assesment by Own", "Let Med Assessor Make Medical Assesment" and "Let Expert System Make Medical Assesment" contribute to the more abstract softgoals "Fast Turnaround" and "Common Sense Judgement".

3.2 Representing whole-part relationships between actors

Represent whole-part relationships between actors. This should be done using "is-partof" links. Each actor has parts or is part of a larger whole. Figure 2 depicts whole-parts relationships. Info Broker, Educational Broker, Reservation Broker, Virtual Visit Broker and System Manager are part of the eCulture System. By doing so, it helps to represent most of the complex social relationships one can find in any environment.

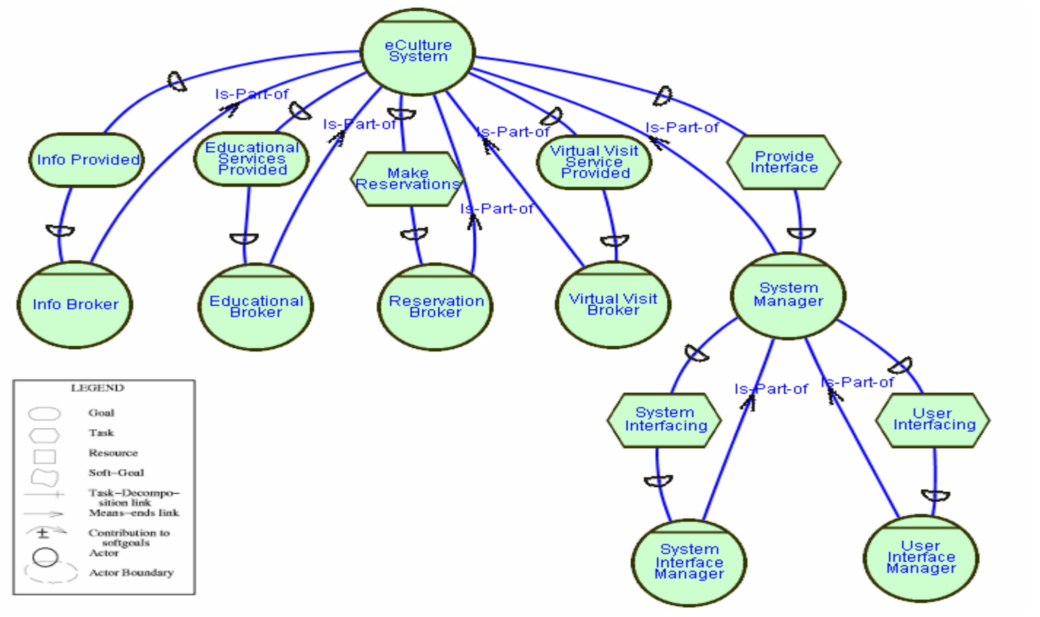


Figure 2

3.3 Representing dependency links

Agents depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. Agents are considered intentional because they have desires and wants. Moreover, agents are strategic as they are concerned about opportunities and vulnerabilities [3]. A good practice is to use various dependency links to describe strategic relationships between actors. There are four types of intentional dependencies: resource dependency, task dependency, goal dependency and softgoal dependency, which is based on a notion of non-functional requirements.

Figure 3 represents the use of dependency links to describe strategic relationships between actors Card Holder and Card Issuer as well as between Card Issuer and Card Manufacturer.

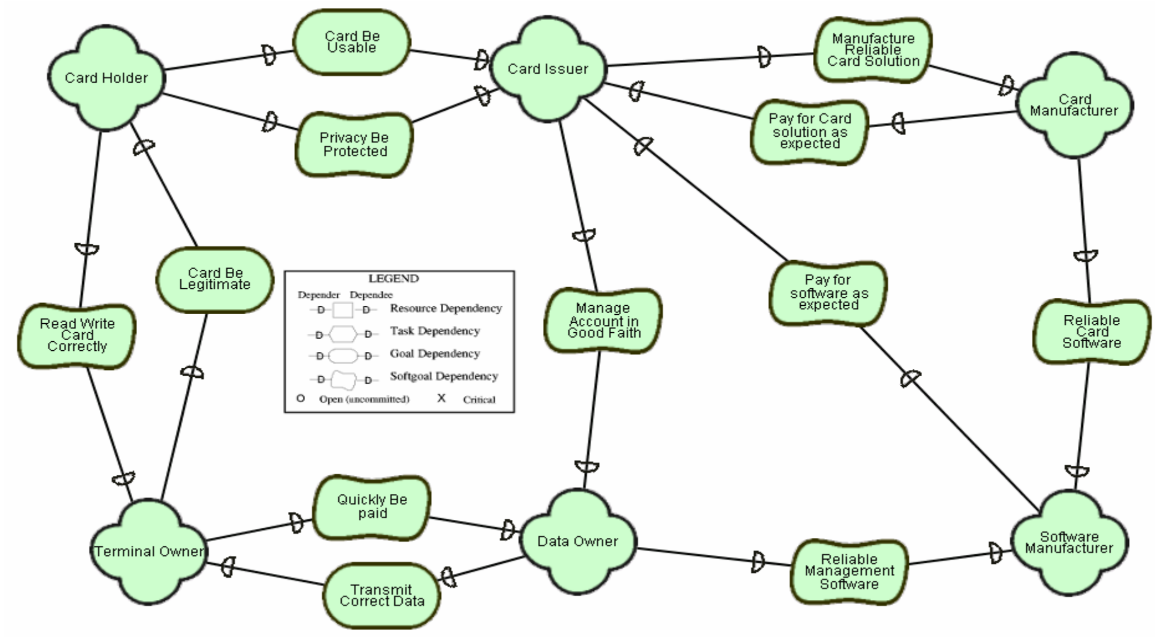


Figure 3

3.4 Hiding goal refinements inside an actor

Actor abstraction is a scoping mechanism used to set boundaries for the propagation of issues and their evaluation. At an actor boundary, an incoming dependency link is considered an implicit means-ends link where the dependum represents the end. On the other hand, an outgoing dependency link is usually also a task-decomposition link, where the dependum is one of the task components [3].

The use of the actor abstraction mechanism is an important good practice. It enables requirements engineers to hide goal refinements inside an actor. This is done with the actor boundary and Expand/Contract mechanism. All components inside the circle on figure 4 are goal refinements of Card Manufacturer. The Expand/Contract mechanism can be used to show or to hide them. Note that at the beginning the requirements engineer may use the simple Actor notion and only later qualify it as role, agent or position. This allows the requirements engineer to start modelling the requirements without having a complete understanding of the environment and later change the models to reflect the knowledge acquired from the requirements process.

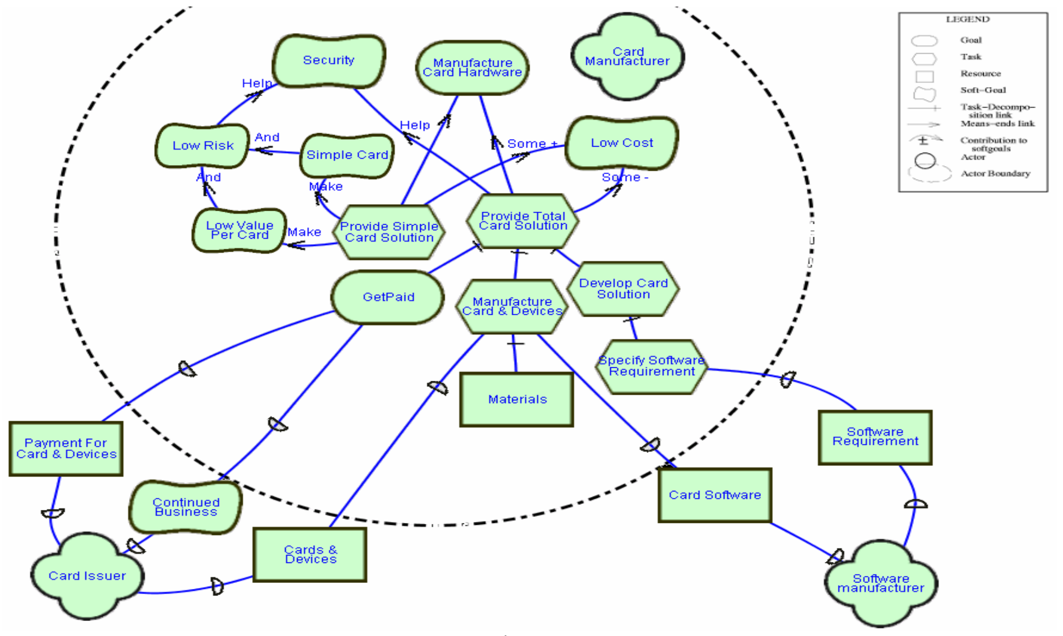


Figure 4

3.5 Evaluating the viability of a softgoal or the satisfaction of a dependency

The notion of viability provides a qualitative assessment on how well the softgoals in a routine are met [15]. The viability of a softgoal or the satisfaction of a dependency can be evaluated with the use of labeling nodes. These are among the strongest mechanisms provided by the i* framework.

Figure 5 shows how well the softgoals "Read Write Card Correctly", "Trust-Worthness Teminal Owner" and "Trust-Worthness" are met. On the paths from the tasks to these softgoals there are labels indicating whether a task contributes to satisfy the softgoal (V) or not (X).

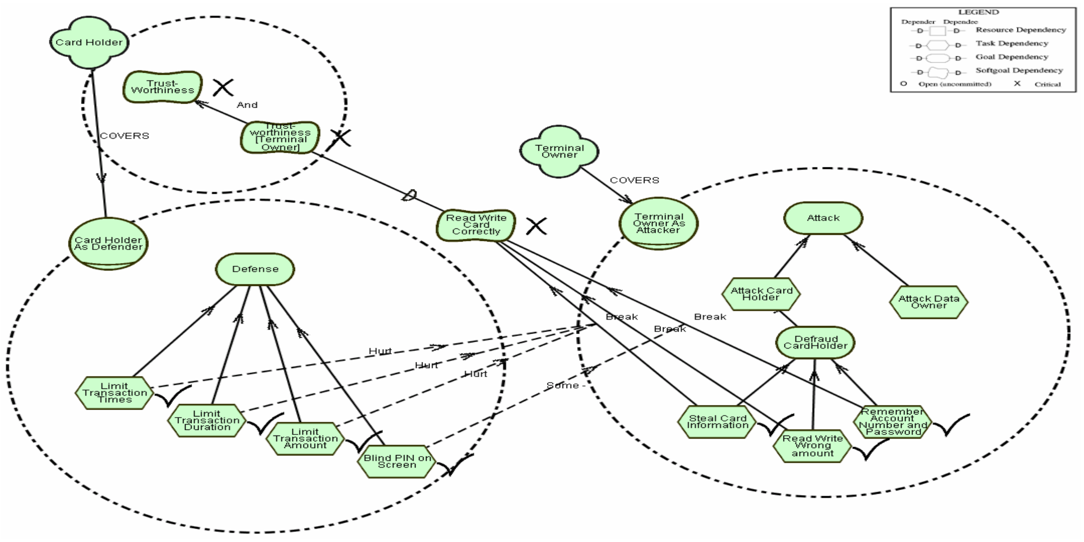


Figure 5

3.6 Evaluating agent social situations according to their roles and positions

Structural relationships among actors are means of representing a social structure. An agent plays some roles and occupies some positions which cover some roles. Furthermore, an actor can be “part-of” another actor or “is-a” specialization of another actor [15]. Requirements engineers should evaluate agents’ social situations according to their roles and positions with the use of inter-agent dependency, role-playing and positionoccupancy relations. Figure 6 shows some role-playing relations. These relations and the agents involved with them are marked with circles. They contribute to evaluate the social situations of the agents.

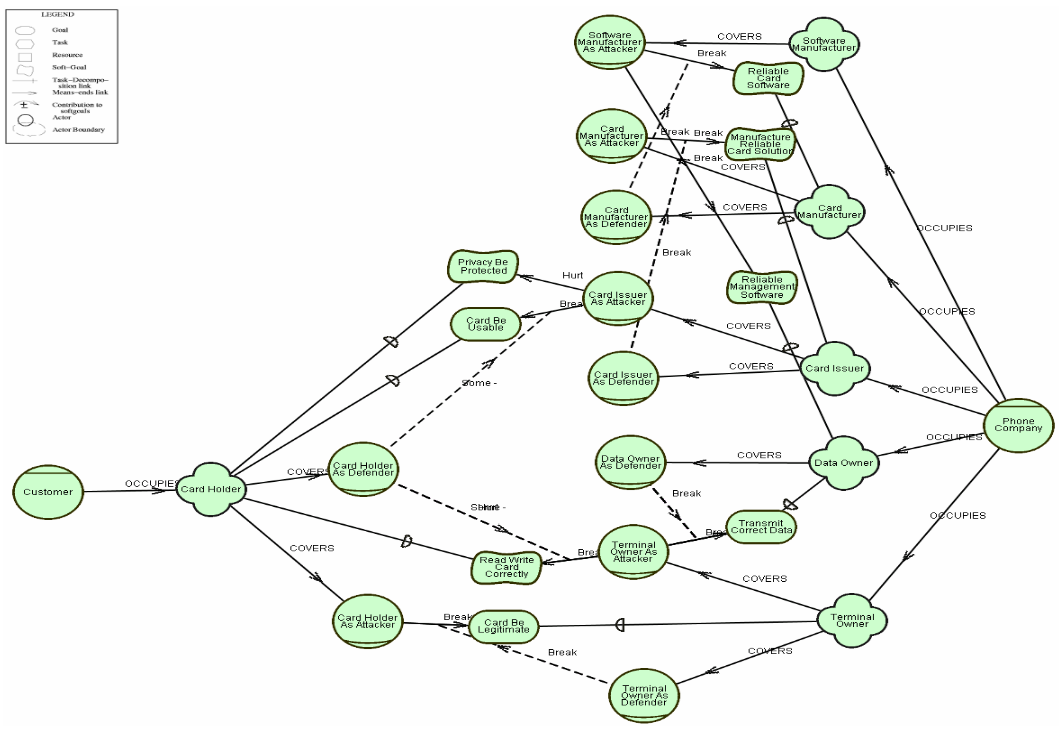


Figure 6

4. i* Framework Misuses

This section presents examples of misuses of the i* framework. Good practices are associated to misuses with the objective of showing the most conforming way of modelling this type of situation.

4.1 Decomposing goals directly into subgoals

i* goals should not be decomposed directly into their subgoals. Figure 7 depicts this misuse [9] on the left side. The goal “Provide e-Cultural Services” should have been decomposed in the solutions it helps to achieve. These solutions, which are on the right side of Figure 7, are the tasks “Allow virtual visit”, “Make Reservation”, “Offer Educational Services” and “Provide Info”.

The goal “Provide info”, on the left side of Figure 7, is decomposed in two subgoals: “Cultural info” and “Logistic info”. These subgoals should have been represented as resources. The correct representation of these subgoals as the resources “Logistc Info” and “Cultural Info” is shown on the right side of Figure7.

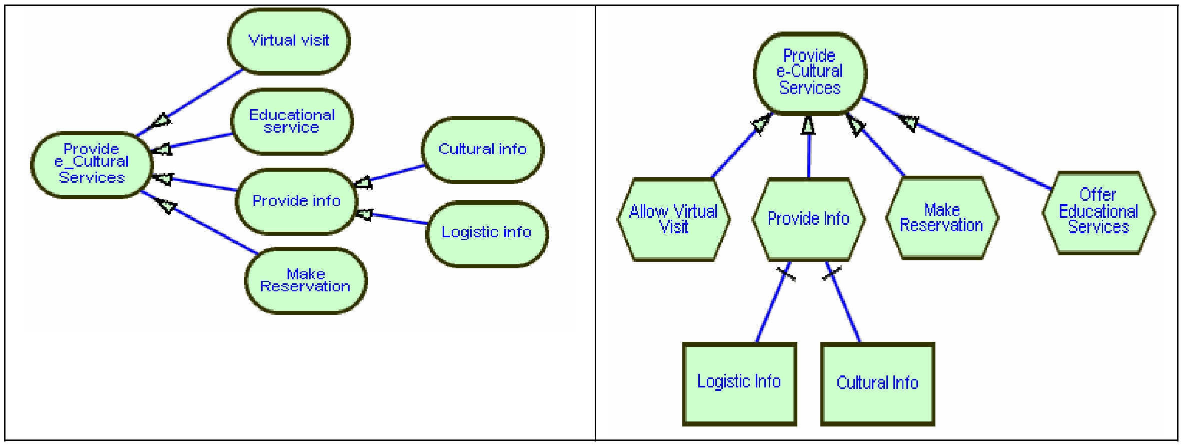


Figure 7 - Misuse (left) and Good Practice (right)

More specifically, this misuse of the i* framework could have been avoided with the connection of alternative solutions to the goal they help to achieve. A good example is shown on figure 8, where three alternative solutions or “means” (search by geographical area, search by keyword and search by time period) are connected to the “goal” (search information). This practice contributes to the design of more detailed models where requirements engineers can better explore alternative systems and organizational arrangements to achieve their goals more effectively.

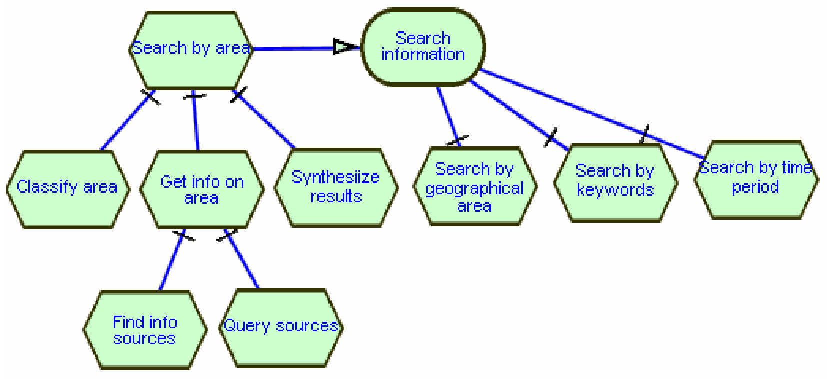


Figure 8

Besides supporting systematic search for alternatives, i* also supports hierarchical decompositions of tasks in their intentional elements [3]. Differently from a goal decomposition where a means-end link is used, a hierarchical decomposition involves all elements that decompose a task.

A good practice is to decompose a task into the subcomponents related to current concerns. This should be done with the use of a decomposition link. A task is modelled in terms of its decomposition into its sub components [3]. The example bellow shows the decomposition of the task “Asses Treatment” in the subcomponents “Policy Manual”, “Verify Patient Policy”, “Medically Assessed” and “Fast TurnAround” which correspond to a resource, task, goal and softgoal respectively.

4.2 Representing actors inside another actor

An actor should not be represented inside another actor. Figure 9 shows the misrepresentation of actors “Area Classifier”, “Info Searcher” and “Results Synthesizer” inside actor “Info Broker” on the left side. The right side represents the correct practice, where “Info Broker” is an i* position that covers the roles of “Info Searcher”, “Area Classifier” and “Results Synthesizer”.

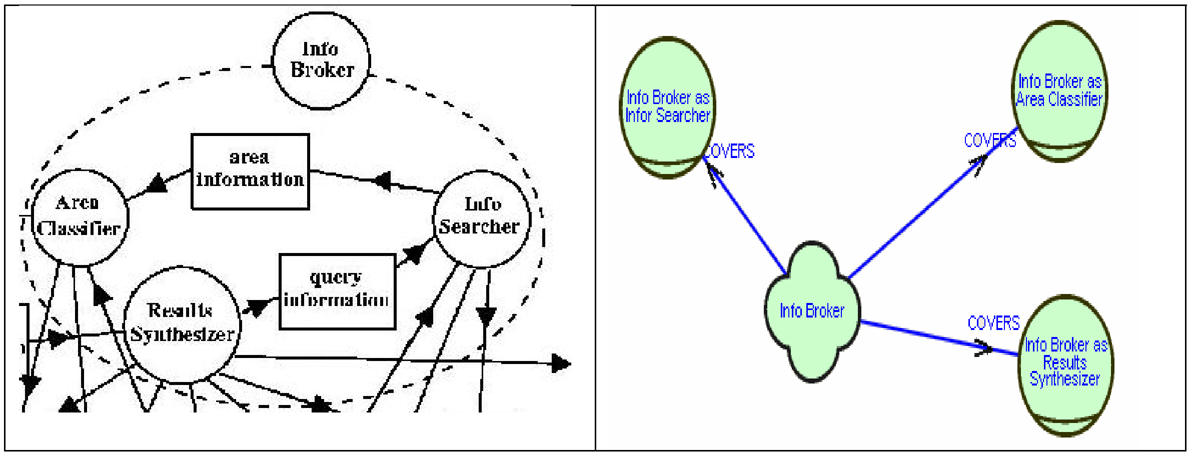


Figure 9 - Misuse (left) and Good Practice (right)

In the context of the i* framework, actors refer to generic entities that have intentionality. To reflect different degrees of concreteness of agency, the concepts of roles, positions and agents are defined as specializations of actors [3]. Actors may be abstract (roles defining responsibilities), concrete (agents – human and non-human individuals or classes with specific capabilities), or other organizational constructs (e.g., positions which package a number of roles together to be assigned to a single concrete agent) [15]. The relationships among the different specializations of actors are defined as follows: an agent occupies a position, and a position is said to cover a role.

A good practice when modelling requirements with the i* framework is to support the use of different specializations of actors. Figure 10 is an example of this practice. It shows the use of agents (Customer and Jim), roles (Card Holder as Attacker and Card Holder as Defender) and positions (Card Holder and Terminal Owner).

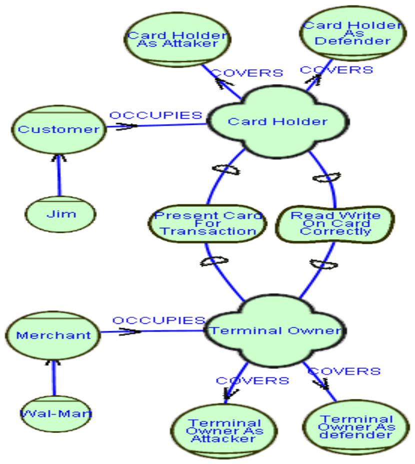


Figure 10

4.3 Modelling softgoals as goals

Softgoals should not be modelled as goals. A softgoal represents a condition for which the criteria of achievement are subject to interpretation because they are not precisely defined. A softgoal functions as a quality goal for a task, goal or resource; directing or restricting selections of alternatives for its decomposition. On the left side of figure 11, “Satisfy customer” should have been modelled as a softgoal and not as a goal. Differently from the goal “Store controller” for which the act of storing determines the condition for its achievement, the attainment of the “Satisfy customer” is subjected to many interpretations. For example, the task “Diagnose fault and perform repairs” could have been one of the criterion to achieve this softgoal. The right side of Figure 11 contains the good practice for this example.

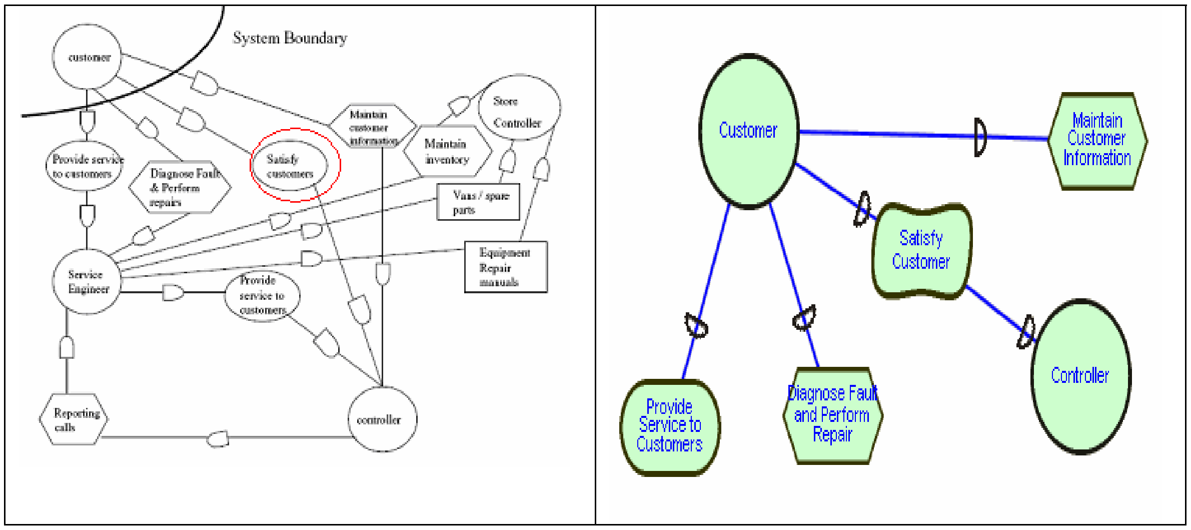

System Boundary
customer
Provide service to customers
Diagnose Fault & Perform repairs
Satisfy customers
Maintain customer information
Maintain inventory
Store Controller
Vans / spare parts
Equipment Repair manuals
Service Engineer
Provide service to customers
Reporting calls
controller
Customer
Maintain Customer Information
Satisfy Customer
Provide Service to Customers
Diagnose Fault and Perform Repair
Controller

Figure 11 - Misuse (left) and Good Practice (right)

Figure 12 is a good example of a well represented softgoal. There are several ways of attaining “Fast Processing (Claim)”, therefore this was modelled as a softgoal.

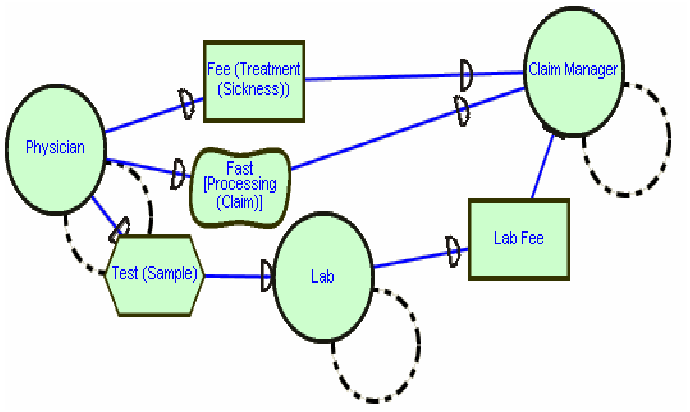
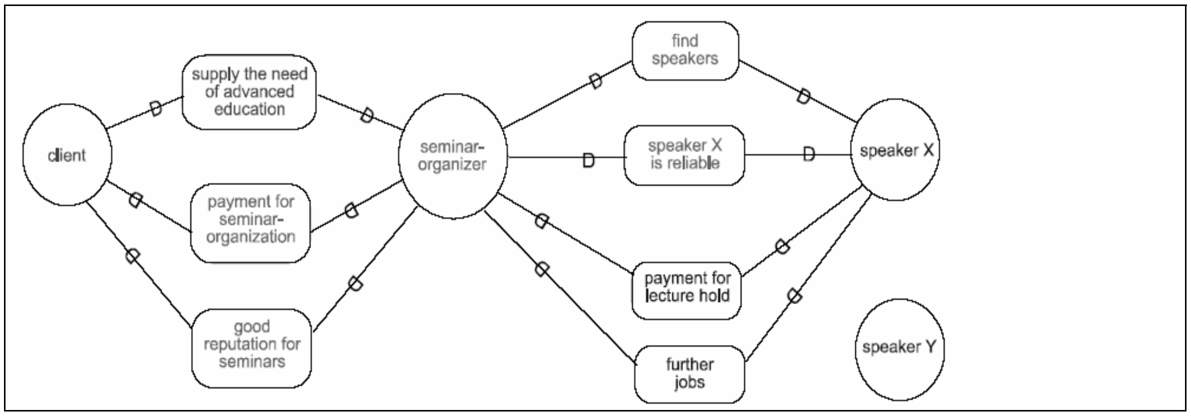


Figure 12

4.4 Modelling softgoals out of the context

On the top of Figure 13 there are two misuses of the i* framework. The first one is quite common and very severe. It represents an actor (Speaker Y) with no links to the model. Although this model depicts a big picture of the seminar organization by stating there are 3 main actors, and showing the main dependencies among them, it confuses the reader. For example, a reader might ask why the modeler chose to model the speakers with two different actors only. Another reader might infer this implies having 2 and only 2 pools of speakers. A third reader might think only Speaker X has dependencies on the seminar organizer, while Speaker Y is completely independent. Such ambiguities may lead to disqualifying the main purpose the diagram tries to achieve.

The second misuse is related to the dependencies. All the dependencies on figure 14 are goal dependencies whereas some of them should have been modelled as other types. For example, "Speaker X is reliable" is modelled as a goal dependency, whereas it should be a softgoal dependency. The goal dependency "Further jobs" should as well be a softgoal dependency. Dependency types should be established based on the type of the dependum. Therefore it is very important to have well defined dependums to avoid the use of wrong dependencies. The bottom of Figure 13 shows the correct example for this case of misuse.



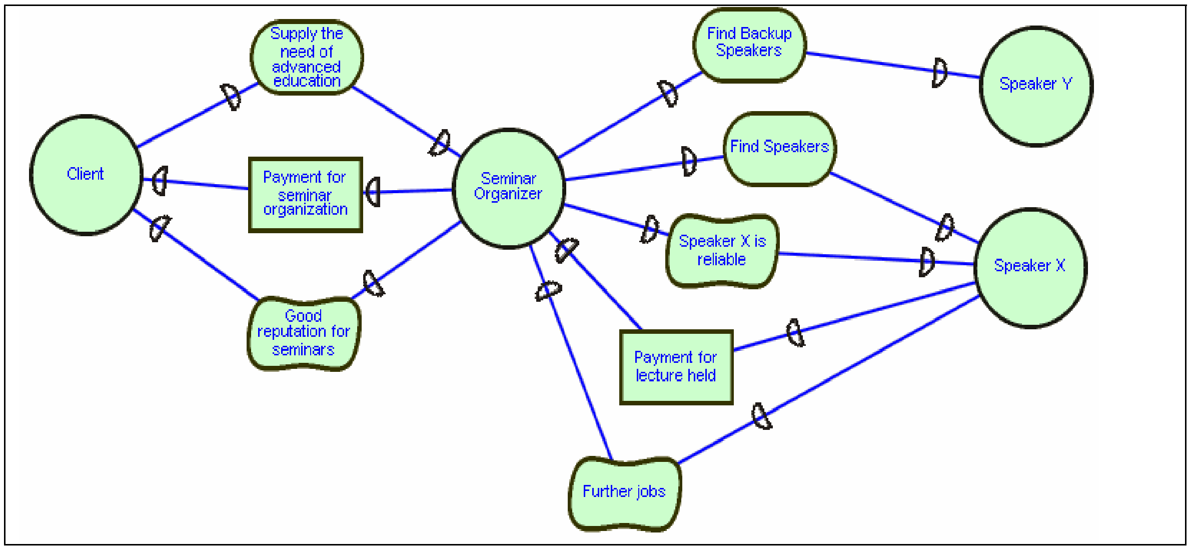


Figure 13 – Misuse (top) and Good Practice (bottom)

5 . Conclusion

This paper brings a set of good practices and common misuses of i* throughout the requirements engineering community. It is based on a survey carried out over more than ten works from many different authors. It brings clarification on what one should aim at or avoid to use when modelling requirements using the i* framework.

We believe that following these guidelines will improve resulting models in the sense that they should be able to more accurately reflect the complexities of current domains. We also believe that these guidelines, if followed, may avoid communication mismatch among the several different groups currently using i*.

References

[1] A. Dardenne, A. van Lamsweerde and S. Fickas, "Goal-Directed Requirements Acquisition", Science of Computer Programming, Vol. 20, 1993, 3-50.
[2] A.I. Antón; "Goal-Based Requirements Analysis", in International Conference in Requirements Engineering (ICRE 96), pages 136-144, Colorado Springs, Colorado, USA, April 1996.
[3] Yu, Eric; "Modelling Strategic Relationships for Processing Engineering", Ph.D. Thesis, University of Toronto, 1994.
[4] Giunchiglia, Fausto. Invited talk at ATAL 2001. 8th International Workshop on Intelligent Agents: Agent Theories, Architectures and Languages, ATAL'01, Seattle, USA, August 1-3, 2001.
[5] Gans, G; Lakemeyer, G.; Jarke, M. and Vits, T.; "SNet: A Modeling and Simulation Environment for AGent Networks Based on i* and ConGolog". In Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE02), LNCS 2348, pages 328--343, Toronto, Canada, May 2002.
[6] Franch, X. and Maiden, "N.A.M. Modeling Component Dependencies to Inform their Selection". In: 2nd International Conference on COTS-Based Software Systems. 2003: Springer.
[7] Dubois, E., Yu, E., Petit, M.; "From Early to Late Formal Requirements". In: Proc. IWSSD '98. IEEE Computer Society Press, 1998.
[8] Yu, E. and Mylopoulos, J.; "From ER to" AR"-Modelling Strategic Actor Relationships for Business Process Reengineering"; Proceedings of the13th International Conference on the Entity-Relationship Approach, pages: 548 – 565, Springer-Verlag London, UK, 1994.
[9] A. Perini, P. Bresciani, F. Giunchiglia, P. Giorgini, J. Mylopoulos. A Knowledge Level Software Engineering Methodology for Agent Oriented Programming. To appear in Proc. of the Fifth International Conference on Autonomous Agents, Montreal, Canada, 28 May - 1 June 2001.

[10] Yu, E., “Strategic Modelling for Enterprise Integration”; Proceedings of the 14th World Congress of the International Federation of Automatic Control, July 5-9, 1999, Beijing, China.
[11] Sutcliffe, A. and Gregoriades, A.; “Validating Functional System Requirements with Scenarios”, Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering; IEEE Computer Society Washington, DC, USA, 2002.
[12] Sutcliffe, A. and Minocha, S.; “Linking Business Modelling to Socio-technical System Design”; Proceedings of the 11th International Conference on Advanced Information Systems Engineering table of contents, pages: 73 – 87, Springer-Verlag London, UK, 1999.
[13] Gans, G.; Jarke, M.; Kethers, S.; Lakemeyer, G.; Ellrich, L.; Funken, C. and Martin, M.; “Requirements Modeling for Organization Networks: A (Dis-)Trust-Based Approach”; International Symposium on Requirements Engineering (RE01), page 154, IEEE Computer Society Washington, DC, USA, 2001.
[14] Chung, L., B.A. Nixon, E. Yu, J. Mylopoulos; “Non-Functional Requirements in Software Engineering”; Kluwer Academic Publishers, 2000.
[15] Cysneiros, L. M. and Yu, E.; “Requirements Engineering for Large-Scale Multi-Agent Systems”; in: A. Garcia, C. Lucena, A. Omicini, F. Zambonelli, J. Castro (Eds). "Software Engineering for Large-Scale Multi-Agent Systems". Springer-Verlag, LNCS 2603, April, 2003.