Understanding the Strategic Actor Diagram: an Exercise of Meta Modeling

Julio Cesar Sampaio do Prado Leite
PUC-Rio (Brasil)
www.inf.puc-rio.br/~julio

Vera Maria B. Werneck
UERJ (Brasil)
www.ime.uerj.br/~vera

Antonio de Padua Albuquerque Oliveira
PUC-Rio (Brasil); UERJ (Brasil)
padua@inf.puc-rio.br ; padua@ime.uerj.br

Claudia Cappelli
PUC-Rio (Brasil)
ccappelli@inf.puc-rio.br

Ana Luiza A. Cerqueira
PUC-Rio (Brasil)
analuiza@inf.puc-rio.br

Herbet de Souza Cunha
PUC-Rio (Brasil)
herbet@inf.puc-rio.br

Bruno Gonzalez-Baixauli
Universidad de Valladolid (España)
bbaixauli@infor.uva.es

Abstract

i-star (i) modeling uses the actor concept to ground the intentions of a given Universe of Discourse. Our work contributes to the understanding of the actor concept as used in i*. We have used a collaborative approach to better understand the actor concept. The authors met 9 times to discuss the topic. The goal was to discuss i* meta-models, which was later specialized to discuss actor modeling. After the meetings and after one week of collaborative work using a collaboration based editor, "Writely", we have agreed on presenting our model from two different perspectives, but both using UML as the meta language. We understand that these models, designed by consensus, represent what we have labeled the SA Diagram or the Strategic Actor Diagram. The article presents the models we have arrived as well as the process we have used. We believe that making this process transparent will help to shed light not only on the concept of actor, but on the process of meta-modeling as well.*

1. Introduction

i* modeling has attracted the attention of the requirements engineering community since its proposal by Eric Yu´s dissertation thesis [25]. Several researchers [8][21][6][17] did see it as a way of modeling early requirements. In the context of information systems [17][6][21][7], i* is viewed as a way of modeling the organization and theirs actors intentionality. In the context of multi-agent systems, it is viewed as the initial step towards agent specification [10][4][20][18][9][5]. Yu´s keynote address at WER 2006 [2] was entitled *"Social Modeling in Requirements Engineering – Why, Where, and How"*, stressing the fact that i*, despite being first named after the idea of distributed intentionality, has been more and more perceived as a social modeling framework.

Embarking on this direction, of a social modeling framework, it is master to underpin a central concept: the concept of actor. The idea of actor modeling is not new, and Carl Hewit [14] was one of the first to use it as a modeling concept. We understand that in order to proper model organizations, actor modeling is fundamental. In this work we concentrated our focus on i* itself [25] and on the work performed around i*.

As stated in the abstract, we started our work aiming at a meta-model for i*, but later on, decided to focus on the actor part. In order to pursue our goal of better understanding this concept, we used a process based on meetings. We conducted 9 meetings, most of which were based on analyzing published i* meta-models. Most of these models were written in UML, but two of them were written in Telos [19]. Yu´s thesis was the primary source of information as to ground our review of the published models as well as on the proposal of our own model.

We believe that making this process transparent will help to shed light not only on the concept of actor, but on the process of meta-modeling as well. Since our long term project is one of improving software transparency [16], we understand that actors should be modeled in a transparent way. The description of our group's work is an exemplar of the ways to explore transparency, as well as the roadblocks that have to be faced in trying to do so.

We start describing the overall process (Section 2), and then we focus on the comparison of previous published models (Section 3). Section 4 discusses two models we have worked with. Section 5 presents our proposed model. We conclude with a section on lessons learnt and with some open questions we believe do remain (Section 6).

2. A Collaborative Process

Working as a group is challenging [11]. Several work have been reported on the topic of computer support collaboration, from work on identifying mechanisms to help collaboration [14] to work that discuss collaboration models [12]. It is not our goal to contribute to this line of research, but to report on the strategy we have used.

The basic elements of our process were meetings, literature review and model building. Eight people, overall, were involved. We have used different sorts of tools, from graphic editors to collaborative editing [13]. Central to our work, outside the meeting room, was the use of group support software provided by Yahoo [23], where we shared messages and files. We shared around 70 messages and 20 files. The SADT below describes the overall process.

PLAN was performed at each meeting, deciding on what to do for the next meeting. Due to the evolving characteristic of our group work, we needed to foresee what was to be done next as to adjust our work to the conclusions of each meeting.

PREPARE was needed to enhance the work. It was done outside the meetings either individually or in group. Based on the agenda set forward by the end of each meeting, people would perform their homework, which was basically literature review and model building.

DISCUSS was performed at the meetings trying to achieve consensus within the group. We mostly worked on trying to understand the model being discussed. We started with complete i* models, but, later on, we decided to focus just on the actor part. Besides the models prepared by others, we have reviewed several models built by our group.

We started discussing i* concepts in general with the goal of sharing the knowledge of i* among the group. After reviewing a meta-model produced by one member of the group, we decided to look for other such models in the literature. In subsequent meeting we discussed several of those models presented in the literature, but focused on the ones using UML as a meta-language. In these discussions we needed to go back to the UML definition [22] as to better understand its capabilities and limitations as a meta-modeling language, which is not an easy task given the volume of information on the definition document. After spending a whole meeting discussing the meta-language as well as the target of our modeling activity we have decided to build a table showing the similitude and differences among the models

At the fourth meeting, as we discussed the contents of the comparison table, it was clear to us that in order to make progress, we needed more focus. We decided to focus on modeling the concept of actor. So, at the end of this meeting, we planed to reduce the scope, and work only with the actor part. This part of the model contains relationships among actors, roles, positions and agents. Figure 2 provides an example of an actor diagram as seen

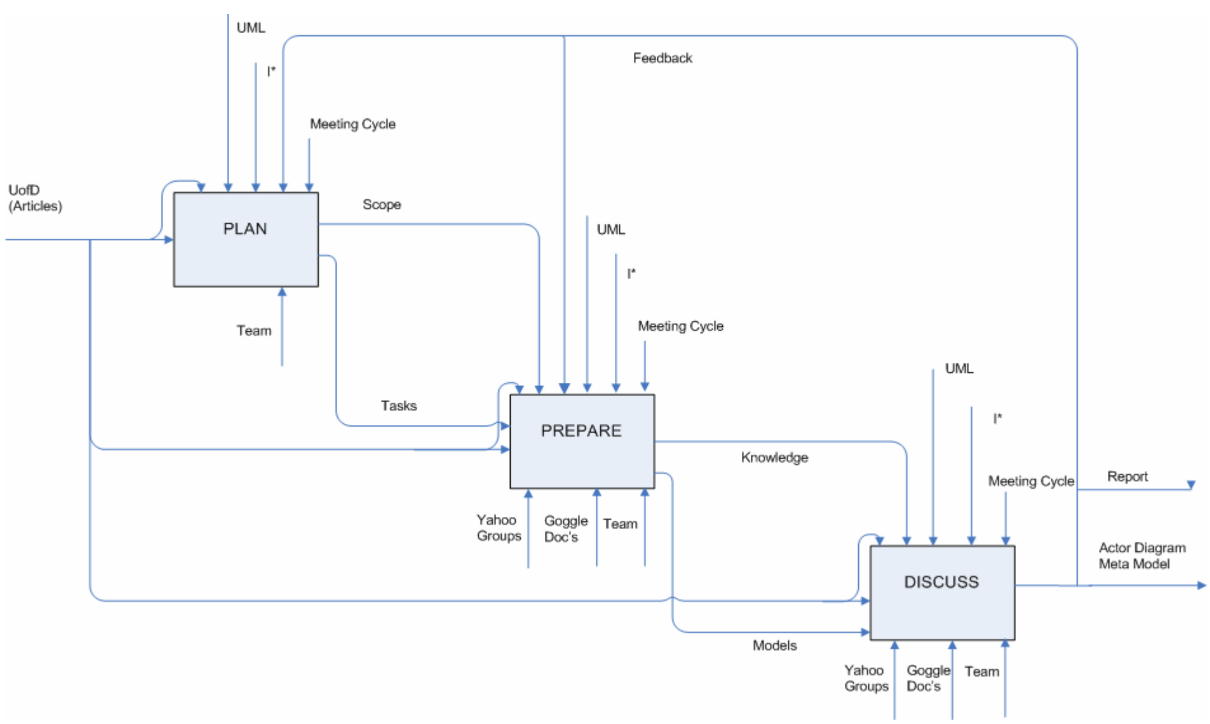


Figure 1: SADT describing the overall process of used strategy

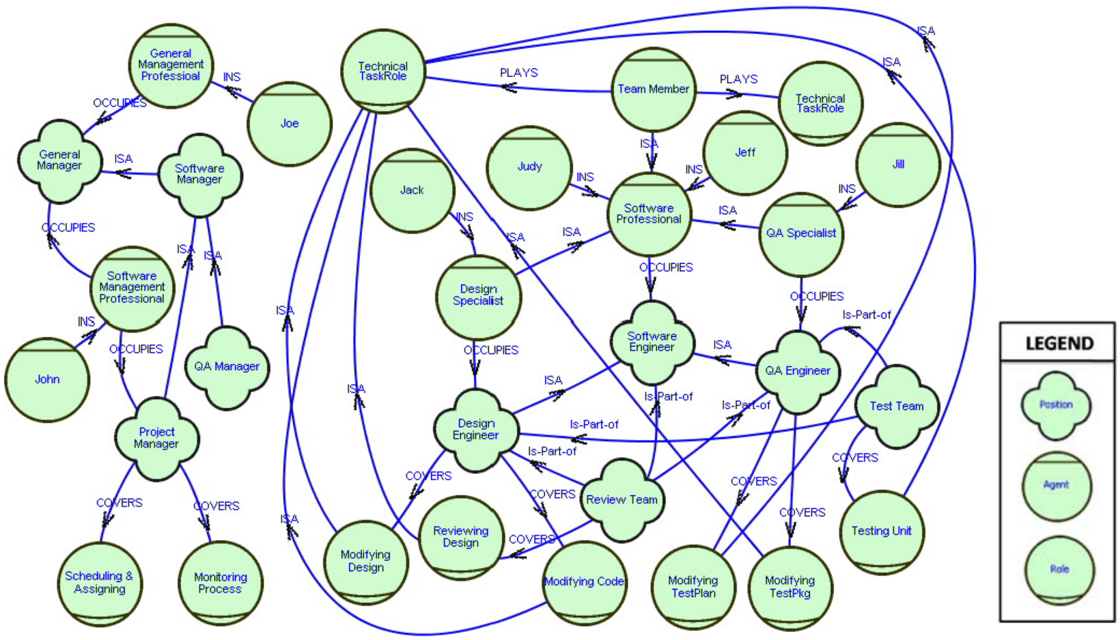


Figure 2: Actor Diagram for a Software Engineering Organization - adapted from [25] pp 99

in Yu´s thesis.

At the fifth meeting we had started with a new comparison table, focused on four meta-models; three from the literature and one from our own. All the other meetings were aimed at producing the actor diagram meta-model. We elected the thesis [25] as the central information source to base our elicitation strategy. No other articles from Yu were used. This part of the work created a lot of discussion in the group because of the different participants visions and especially because of the parts-of relationship among roles, positions and agents, since there was a proposal to use the composite design pattern as to base the parts-of meta-modeling. After revisiting the discussion on composition and aggregations we decided not to use the design pattern version. Around eight different UML meta-models were discussed until we reached a consensus. The result of each discussion is represented by the "Feedback" arrow in the SADT (Figure 1).

3. Comparison of Previous Models

We have looked at different literature [3][20][4][24][1], but focused our comparison on the first three models. One was discarded because the model was the same of a previous publication, and other was discarded since it did not use UML as a meta-model [24]. We compared these models with the one produced by one of us as we started our work (Figure 6). See Figure 3, 4, 5 and 6 below.

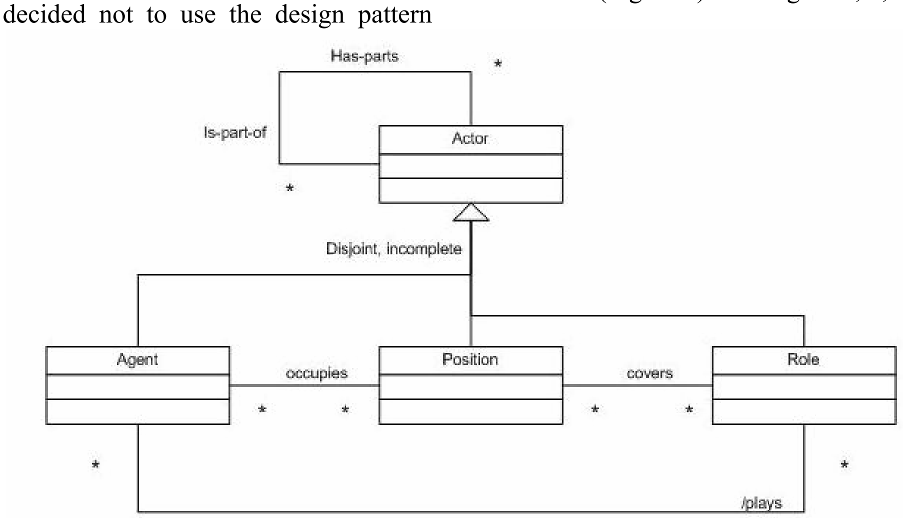


Figure 3: Model from Ayala et al. [3]

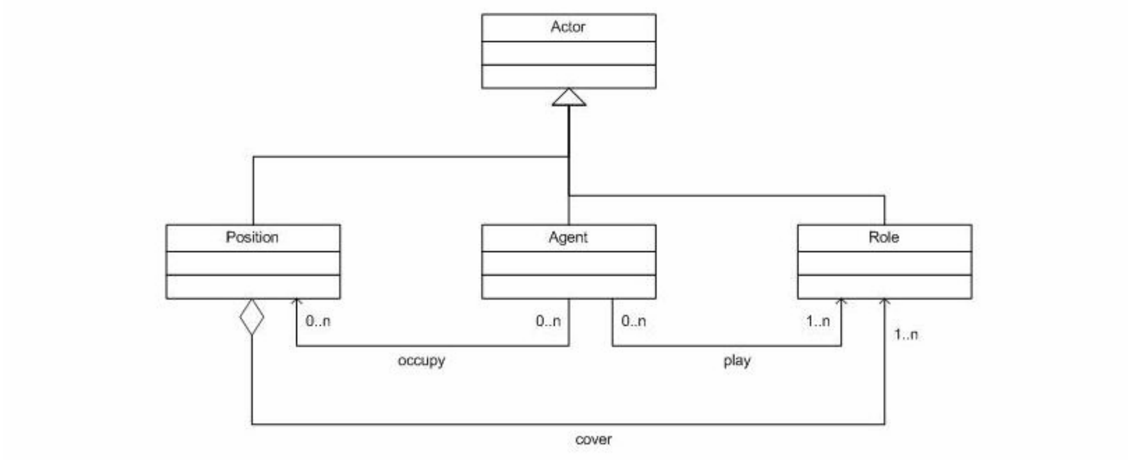


Figure 4: Model from Susi et al. [20]

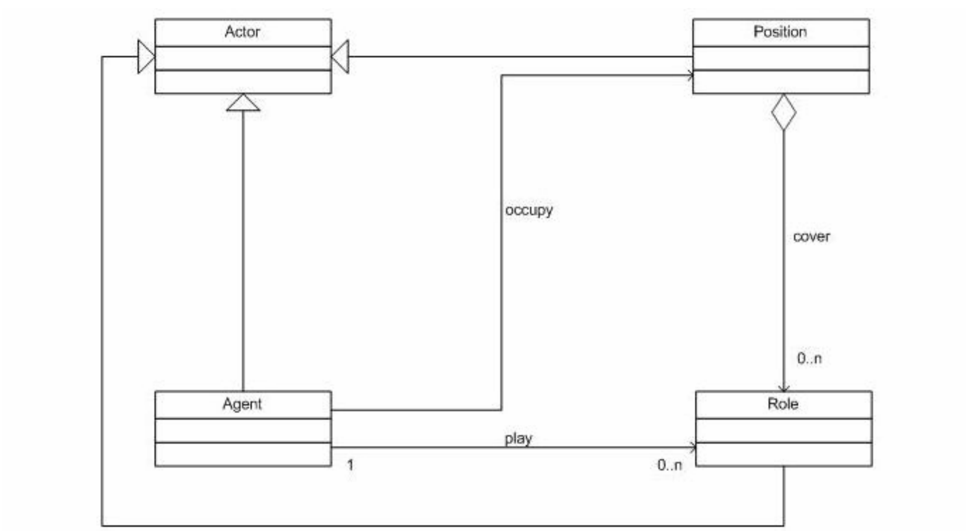


Figure 5: Model from Bertolini et al. [4]

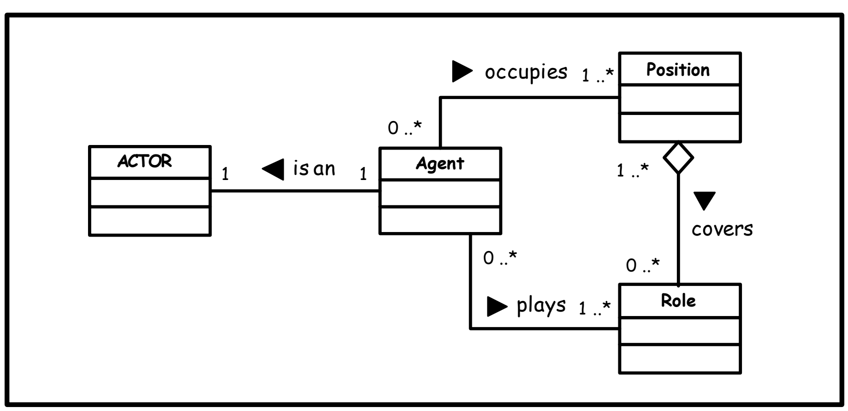


Figure 6: Our initial model

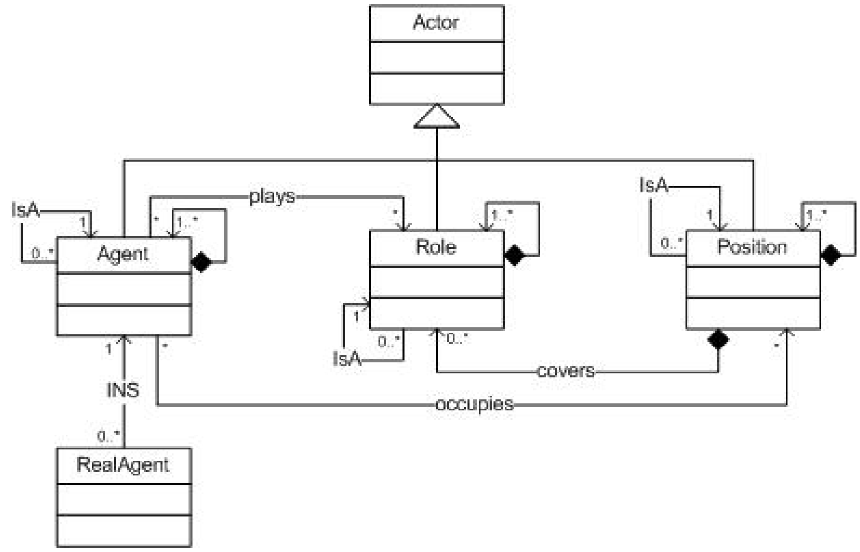


Figure 7: Intermediary Model Using UML Specialization and Aggregation

All the first three models have the specialization relationship from Actor; our initial model had only the specialization related to Agent. Ayala et al. [3] added the constraint {Disjoint, incomplete} over the Actor specialization. We targeted our comparison on 4 relationships:

- Relation Cover (Position → Role)
- Relation Plays (Agent → Role)
- Relation Occupies (Agent → Position)
- Relation Part of (Actor → Actor)

As Table 1 shows, there is no agreement over multiplicity as seen by these authors. It also shows a relationship, parts-of, among actors, that was present at just one model. In the next Section we comment on the intermediary models we have built in the process of better understanding the actor concept.

4. Working (Draft) Models

Most of our discussions (see Figure 1, DISCUSS) were regarding the multiplicity of the mapped relationships.[1] The discussions were based in the examples shown in Yu's thesis [25] and on the Telos definition of the i* classes as in the thesis. Figures 7 and 8 depict two of the intermediary models we had scribbled. Figure 8 is basically a different modeling perspective, where the relationships are modeled explicitly.

[1] As seen on Table 1, our initial model (Figure 6) did not treat the "Part of" relationship, so one of our first attempts was to look into this issue. Our discussions led to Figure 7, whereas this issue is treated.

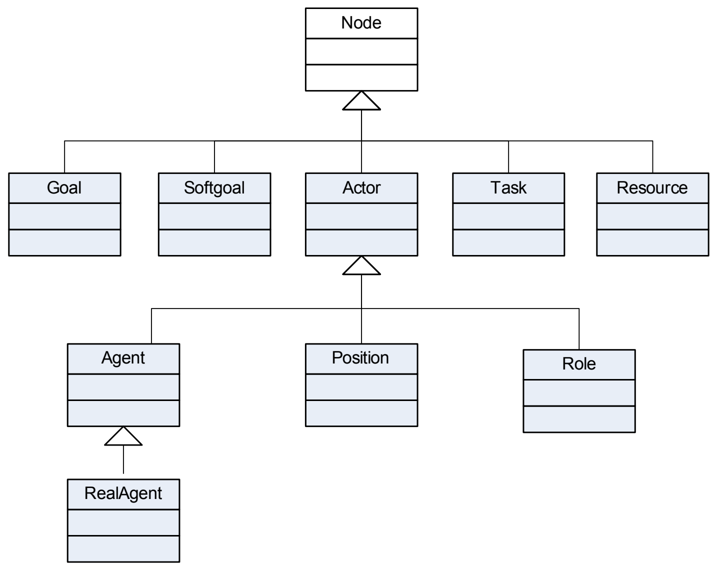
Figure 7 is a more standard UML model that shows the parts-of relationships as composition. The specialization relationship is represented in two forms, with the special UML symbol and with the annotated relationship IsA. This reflects the difficulty to find out a good enough representation of "is-a-kind-of" (is a) relationship concerning actors, agents, positions, and roles. The specialization semantic in UML is not the same that is used in i* framework. Agents, positions, and roles are represented in i* framework as an actor refinement which is not exactly a specialization in UML sense.

Table 1: Comparison Table

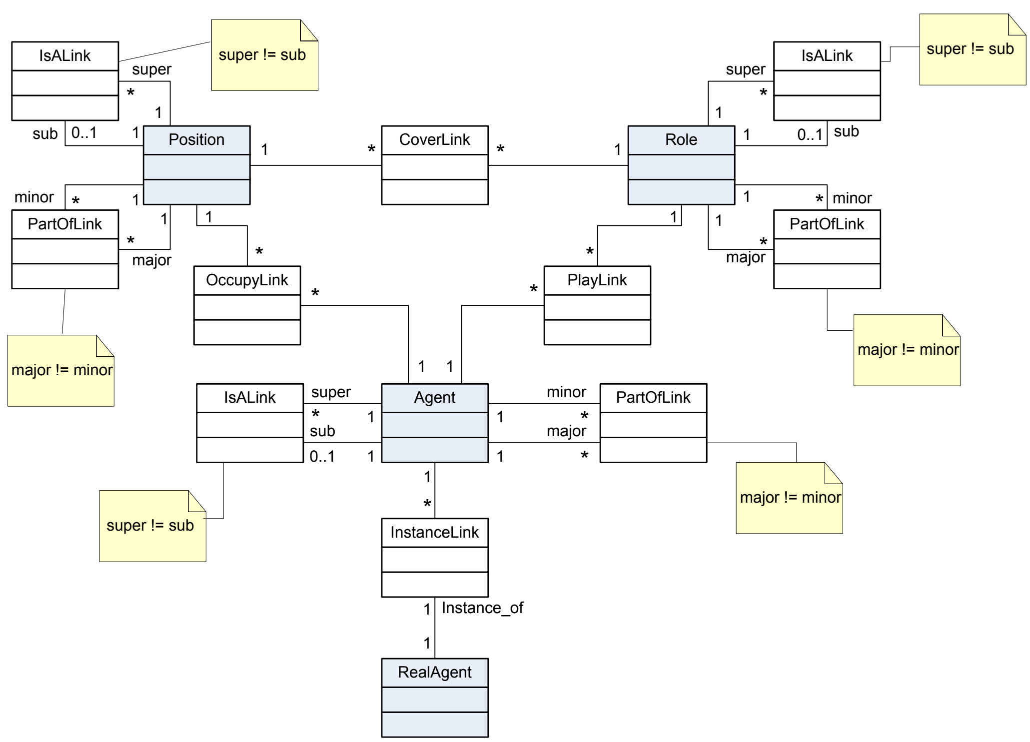
Relationship	Ayala [3]	Susi [20]	Bertolini [4]	Our initial model
Position covers Role	*...*	1 ... 1,n	1 ... 0,n	1,* ... 0,*
Agent plays Role	* ... *	0,n ... 1,n	1 ... 0,n	0,* ... 1,*
Agent occupies Position	* ... *	0,n ... 0,n	no multiplicity	0,* ... 1,*
Part of (Actor→Actor)	*...*	does not have	does not have	does not have

Figure 8.b shows the relationships as classes. We used the same scheme as used in the UML meta definition [22] to represent the relationships. This model stresses the multiplicity and factors out the relationships.

In order to represent the "instance" relationship among agents we preferred to add a class called "real agent". In this way we explicit state that only real agents can instantiate agents. At this point we decided to work with two meta-models, the first one more concise giving



(a) Nodes classes



(b) Relationships (between nodes) represented by classes

Figure 8: Intermediary Model Using Classes to Represent Relationships

emphasis on actors, agents, positions, roles elements and the other one giving importance to the relationships representing them as UML classes, in the same way as elements.

5. Proposed Models

Figures 9 and 10 depict our understanding of the actor concept as a consequence of our discussions, which departed from the evaluation presented at Figure 1. We choose to present both as different visions, of the same understanding. They are richer than our previous models (Figures 6, 7 and 8), as well as from the models we found in the literature (Figures 3, 4 and 5).

In this Section we summarize our findings for each concept we model, present the models and provide the rationale for the concepts, relationships, and multiplicity we have modeled. Theses parts follow as different Sub-Sections.

5.1. Concepts

The concepts were captured from Yu's thesis [25]. The relationships are commented with respect to their multiplicity as used in our proposed models.

Actor : "An actor is an active entity that carries out actions to achieve goals by exercising its Know-how" [25], pp 12. "We use the term actor to refer generically to any unit to which intentional dependencies can be ascribed" [25], pp 17. We consider actor as a super class for agent, position and role.

Position : "A position is an intermediate level in abstraction between a role and an agent. It is a set of roles typically played by one agent (e.g., assigned jointly to that one agent). We say that an agent occupies a position. A position is said to cover a role." [25], pp 17.

Role : "A role is an abstract characterization of the behavior of a social actor within some specialized context or domain of endeavor. Its characteristics are easily transferable to other social actors Dependencies are associated with a role when these dependencies apply regardless of who plays the role." [25], pp 17.

Agent : "An agent is an actor with concrete, physical manifestations, such as a human individual. We use the term agent instead of person in order to generalize, so it can refer to human as well as artificial (hardware/software) agents. An agent has dependencies that apply regardless of what roles he/she/it is playing. These characteristics are not typically transferable to other individuals, e.g. its skills and experiences, and its physical limitations." [25], pp 17.

Real Agent : As said before, we differentiated real agent from agent to clarify that an agent (more generic) must be instantiated by real agent (more specific). So, a real agent is a specific agent that can be uniquely identified, e.g., a specific a person or a specific hardware or software.

Instance : It relates real agent with agent. We consider that a real agent can instantiate exactly one agent and one agent can be instantiated by zero or more real agents.

Cover : It relates position with role. We consider that a position can cover zero or more roles as well a role can be covered by zero or more positions. Although it seems odd to have a position without covering any role (or a role that is not covered by any position) in some cases is desirable to work just with positions and do not care about the roles and vice-versa.

Play : It relates agent with role. We consider that an agent can play zero or more roles as well a role can be played by zero or more agents. Although it seems odd to have an agent without playing any role (or a role that is not played by any agent) in some cases is desirable to work just with agents and do not care about the roles and vice-versa.

Occupy : It relates agent with position. We consider that an agent can occupy zero or more position as well a position can be occupied by zero or more agents. Although it seems odd to have an agent without occupying any position (or a position that is not occupied by any agent) in some cases could be desirable to work just with agents and do not care about the positions and vice-versa.

Part of : Roles, positions and agents can each have subparts. "Aggregate actors are not compositional with respect to intentionality. Each actor, regardless of whether it has parts, or is part of a larger, whole is taken to be intentional. Each actor has inherent freedom and is therefore ultimately unpredictable. There can be intentional dependencies between the whole and its parts, e.g., a dependency by the whole on its parts to maintain unity." [25], pp 17. We consider that roles, positions and agents, can have zero or more subparts as well can be taken as a subpart of zero or more roles, positions and agents respectively. In the case of positions there is a restriction: a position P1 is (sub) part of another position P2 if, only if, all the roles covered by the position P1 are also covered by the position P2. We do not consider the relation part-of as applied to Actor, as did the model at Figure 3.

Is a : Actors, roles, positions and agents can be specialized. We consider that actors, roles, positions and agents, can be specialized by zero or more actors, roles, positions and agents respectively as well an actor, a role, a position and an agent can specialize zero or more actors, roles, positions, and agents respectively.

5.2. Models

As explained above, the model of Figure 9 used the UML special symbols for the generalization and aggregation associations. Since we need to use an ISA association with a particular semantics, we named it as an association, and where the case, we refrained to use the UML special symbol.

The model of Figure 10.c is formed by two groups of meta-classes: nodes meta-classes and links meta-classes. The nodes meta-classes (Figure 10.a) represent the key concepts in a SA model: actor, agent, position, role and real-agent. The links meta-classes (Figure 10.b) represent the different types of relationship between key concepts. We use a different link meta-class for each possible relationship in a SA: instance link, cover link, play link, occupy link, is-a link and part-of link. Like all the links are directed in a SA model, and connects exactly two different nodes, we named the two correspondents

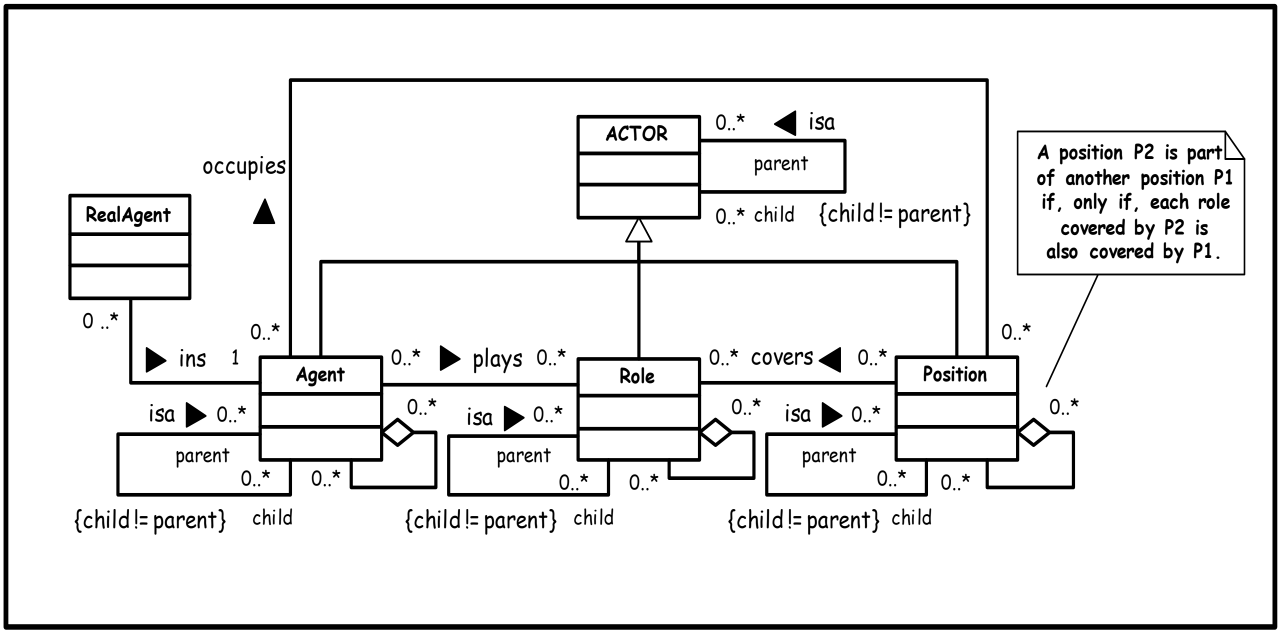


Figure 9: Proposed Model Using UML Specialization and Aggregation

associations in the link meta-classes as "from" and "to"[2]. The restriction that a link can not connect an element to itself applies for all types of links. In fact, by the nature of the relationships between actors represented in SA model it does not make sense that any actor (in any level of abstraction) could have a relation to itself. For instance, stating that actor A plays a role of actor A.

5.3. Rationale

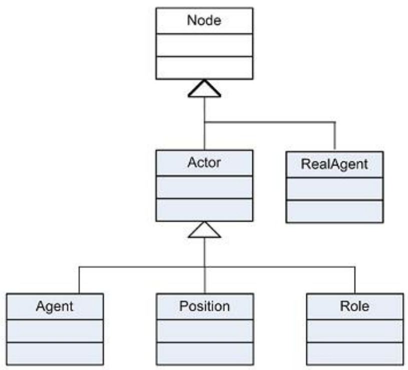
All relationships we proposed were defined in [25], pp 17, and for the multiplicities we based our choice in examples found in the thesis [25].

In our model we defined that a Position can cover zero or more Roles and the Roles can be covered by zero or more Positions. The example of Strategic Dependency model of a software engineering process (Figure 2) presents the justification for the zero or more Positions and the more Roles. A Technical Task Role is defined covering any Positions. As we defined as a restriction rule, if a position P1 is part of other position P2 by definition all roles of P1 covers also P2. So the Test Team Position and the QA Engineer Position covers the Testing Unit Role, because the Test Team Position is a part of the QA Engineer Position. The QA Engineer Position is covered by the Roles Modifying Test Plan and Modifying Test Pkg. An example of Position defined without any Role can be found in the Figure 2.5 [25], pp 103, where the Project Manager Position is defined showing only the dependency on the Designer Agent without any Roles to be covered by this Position.

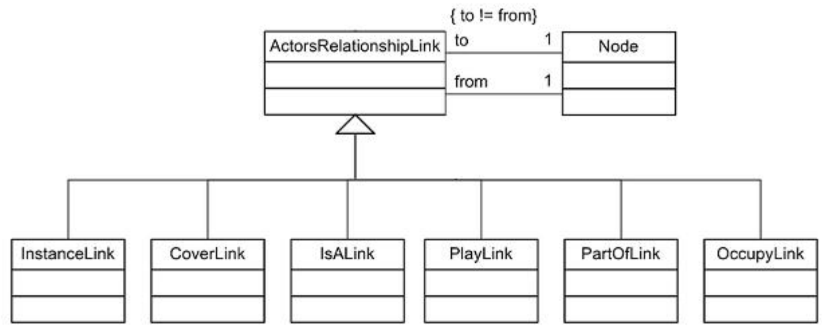
An Agent can play zero or more Roles and Roles can be played by zero or more Agents. In the example of Strategic Dependency model of a software engineering process (Figure 2) we can find these situations. The Team Member Agent plays the Technical Task Role, the Software Management Professional Agent does not play any Role and the Scheduling Assign Role has no Agent associations. In the Figure 2.5 in [25], pp 103, the Designer Agent plays the Designing Role and the Tester Agent plays the Testing Role. The Designer and the Tester Agents are part of the Tech Team so we can infer that the Tech Team Agent can also play the Designing and the Testing Roles, so those two roles can be played by two different agents.

An Agent can occupy one or more Positions and we also defined that a Position can be occupied by zero or more Agents. The example of Agents, Roles and Positions in Figure 2.8 in [25], pp 24, shows the Physician Person Agent that occupies two Positions: Professor Position and Physician Position. We can also define a domain that has two Agents (Physician Person and Nutritionist Person) with a same Position (Professor Position). The example of Strategic Dependency model of a software engineering process (Figure 2) shows the Team Member that that has no positions associations and the QA Manger Position that has no agents associations.

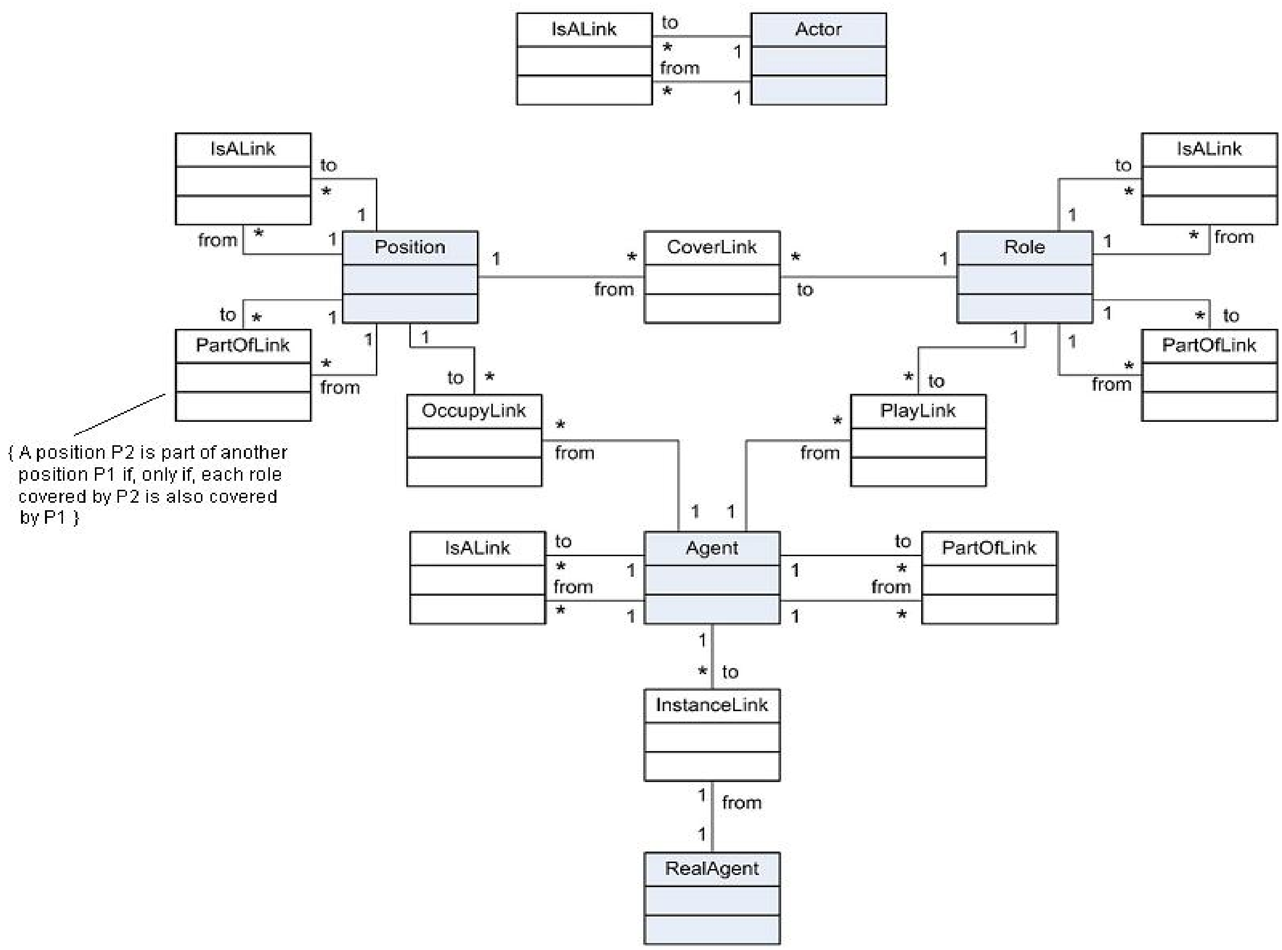
[2] The labels "from" and "to" are indicative of the relationship direction and were used as to improve the labels used in Figure 8.



(a) Nodes meta-classes



(b) Relationships meta-classes



(c) Model using relationships meta-classes

Figure 10: Proposed Model Using Classes to Represent Relationships

A Real Agent can instance exactly one Agent but an Agent can have zero or more Real Agent that instances this Agent. The example of Strategic Dependency model of a software engineering process (Figure 2) presents the Judy and Jeff two Real Agents that instance the Software Professional Agent. In the same example we find the Team Member Agent that has no Real Agent associations.

An Agent is a part of zero or more Agents. The example in the Figure 7.3 in [25], pp 103, presents that the Designer Agent and the Tester Agent are part of the Tech Team Agent. In the Figure 2, the Design Specialist Agent is not a part of any Agent.

The Generalization/Specialization relationship between two Agents (ISA) can also be expressed by zero or more Agents. Figure 2, presents two Agents, Design Specialist and QA Specialist, ISA Software Professional Agent. In the same example the General Management Professional Agent has no relationship between other agents.

A Position is a part of zero or more Positions. The example on Figure 2 presents that the two Positions, Review Team and the Tester Team are part of the QA Engineer Position. In the same example the QA Manager Position is not a part of any Position and Review Team Position is part of three Positions (Design Engineer, Software Engineer and QA Engineer).

The Generalization/Specialization relationship between two Positions (ISA) can also be expressed by zero or more Positions. Figure 2 presents the Positions Design Engineer and QA Engineer as a specialization of Software Engineer Position. In the same example the QA Manager Position is neither a generic nor a specialization of any Position.

A Role is a part of zero or more Roles. The example on Figure 2.8 in [25], pp 24, presents that the Billing Role as part of Managing Clinic Role and the Treating Patient Role is not a part of any other Role.

The Generalization/Specialization relationship between two Roles (ISA) can also be expressed by zero or more Roles. The Figure 2 presents the Roles Modifying Design, Reviewing Design and Modifying Code as a specialization of Technical Task Role. In the same example the Monitoring Progress Role is neither a generic nor a specialization of any other Role.

Although, Yu defined in the i* Meta Model a CompositeActorClass (Figure 2.4 in [25], pp 18) we understand that this relationship hurts the principle of "inherent freedom", as such actor is understand as an independent entity and not decomposable in other actors, as seen in the model presented at Figure 3.

The Generalization/Specialization relationship between two Actors (ISA) can also be expressed by zero or more Actors. In the example Strategic Dependency Model of Meeting Schedule, Yu (Figure 4.1 and 4.2 pp 60 and 61 in [25]) defined Important Participant as a specialization of a Participant. In this Domain those two actors have relevant dependencies to be modeled.

6. Conclusions

This paper aims to report, in a transparent way, the work we have developed to obtain a better understanding of the actor concept in the context of i* modeling. Central to this initiative was how the UML models have evolved as the group discussed the models and the modeling process. One of our conclusions is that the actor concept should be treated as a first class graph as did the Strategic Rationale and the Strategic Dependency models. From the thesis we are lead to believe that the actor concept and its relationships would be accessory to the Strategic Dependency model. It is our understanding that we should have a Strategic Actor model as well. We believe that You [24] in a certain way also reached that conclusion. As such, our proposed model is a starting point to structure and discuss a Strategic Actor model.

Our proposed model is different from the previous ones, as it is more detailed and mostly because it explicitly states the rationale of the decisions we took for building the model. Providing this kind of traceability will help others analyze our model. It is not our goal to have the last word on the i* actor concept model, but we understand that we built a solid base to continue our research on actor modeling. Our model is less restrictive than others, as can be seen by the use of the * multiplicity. As such, our model is more flexible, and we firmly believe this was the intention of the author [25].

Mapping actors is of fundamental importance to large organizations, either people organizations or software organizations. One of the problems we found on exploring the i* concept of actor is the lack of support to model hierarchical models. At first, the parts-of would be the natural candidate, but as we look further we understood that it could not be the case, as we explained in the previous Section.

Regarding the process, although we did not aim to research on collaboration, we realize the difficulties of working with a large group. We spent a lot of time discussing supposedly previously known concepts, in the case of UML, and we faulted as we did not keep the minutes of each meeting. We rely too much on the collaborative memory, as we had plans for each next meeting. This also caused that some discussions occurred over and over again.

UML modeling is deceiving; it is more difficult than at first sight. The first author had the opportunity of reading a draft of Hugo Estrada´s doctoral thesis where he provides a detailed analysis of i* concepts. There he uses a more refined ontology (multiplicity, transitivity, reflexivity, symmetry, homogeneity, work assumption, shareability, existence dependency …) to categorize and describe the concepts. As we had chosen UML we did not addressed all these issues, and as such we had more difficulty in expressing some of our findings.

We foresee a continuation of our work. We will use the actor concept in modeling organizations. We will also thrive to enhance the model with other characteristics. One such opportunity is dealing with hierarchies. Other opportunity is exploring what characteristics would be necessary to model, for instance, competences and responsibilities of employees. Providing better models for organization actors will definitely help in having more transparent organization models.

Acknowledgements: Bruno González-Baixauli has been supported by the Spanish government (MEC-FPU grant and by MEC-FEDER project TIN2004-03145).Julio Cesar Sampaio do Prado Leite has been supported by CNPq.

7. References

[1] F. Alencar, A. Moreira, J. Araújo, J. Castro, C.T.L.L. Silva, and J. Mylopoulos, "Using Aspects to Simplify iModels", *14th IEEE International Conference on Requirements Engineering (RE 2006)*, Sept. 2006, Minneapolis, Minnesota, USA. IEEE Computer Society, pp. 328-329 (extended version).

[2] F. Alencar, J. Sánchez, and V. Werneck (Ed.), "Anais do WER06 - Workshop em Engenharia de Requisitos", Rio de Janeiro, RJ, Brasil, Julho 13-14, 2006. (http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER06/introducao-indice.pdf)

[3] C. Ayala, C. Cares, J.P. Carvallo, G. Grau, M. Haya, G. Salazar, X. Franch, E. Mayol, and C. Quer, "A Comparative Analysis of i*-Based Goal-Oriented Modeling Languages". *17th International Conference on Software Engineering and Knowledge Engineering (SEKE'05),* Howard International House, Taipei, Taiwan. July, 2005

[4] D. Bertolini, A. Perini, A. Susi, and H. Mouratidis, "The Tropos visual modeling language. A MOF 1.4 compliant meta-model". *AOSE TFG meeting, collocated with the 2nd Agentlink III Technical Forum (AL3-TF2)*, Feb. 2005, Ljubljana, Slovenia

[5] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini, "TROPOS: An Agent-Oriented Software Development Methodology". *Journal of Autonomous Agents and Multi-Agent Systems*, May 2004, Kluwer Academic Publishers.

[6] D. Bolchini and P. Paolini, "Goal-driven requirements analysis for hypermedia-intensive Web applications". *Requirements Engineering Journal. 9(2)* May 2004, Springer, pp. 85-103.

[7] D. Bolchini, P. Paolini, and G. Randazzo, "Adding Hypermedia Requirements to Goal-Driven Analysis," *11th IEEE International Requirements Engineering Conference (RE'03)*, 2003, pp. 127.

[8] J.F. Castro, F.M.R. Alencar, G.A.C. Filhol, and J. Mylopoulos, "Integrating organizational requirements and object oriented modeling," *Requirements Engineering, RE 2001, Fifth IEEE International Symposium on*, 2001, pp.146-153

[9] J.F. Castro, P. Giorgini, S. Kethers, and J. Mylopoulos. "A Requirements-Driven Methodology for Agent-Oriented Software." *Agent-Oriented Methodologies*, B. Henderson-Sellers and P. Giorgini (Eds), Idea Group. 2005.

[10] J.F. Castro, M. Kolp, and J. Mylopoulos, "Towards Requirements-Driven Software Development Methodology: The Tropos Project," *Information Systems*, June 2002

[11] C.A. Ellis, S.J. Gibbs, and G.L. Rein, "Groupware: Some Issues and Experiences." *Commun. ACM 34(1)*, pp. 39-58, 1991

[12] H. Fuks, A. Raposo, and M.A. Gerosa, "Do Modelo de Colaboração 3C à Engenharia de Groupware". *WEBMIDIA 2003 - Simpósio Brasileiro de Sistemas Multimídia e Web, Trilha especial de Trabalho Cooperativo Assistido por Computador*, November 2003, Salvador-BA, pp. 445-452.

[13] Google Docs: http://docs.google.com

[14] C. Hewitt, "Offices are open systems," *ACM Trans. Inf. Syst.* 4(3) (Jul. 1986), pp.271-287.

[15] R.E. Kraut, S.R. Fussell, S.E. Brennan, and J. Siegel, "Understanding effects of proximity on collaboration: Implications for technologies to support remote collaborative work", *Distributed work*, P. Hinds & S. Kiesler (Eds.), Cambridge, MA, US: MIT Press, 2002, pp. 137-162

[16] J.C.S.P. Leite, "Transparência: Desafios para a Engenharia de Software", Sexta-Feira, May 19th, 2006 http://jcspl.wordpress.com/2006/05/19/transparencia-desafios-para-a-engenharia-de-software/

[17] N.A.M. Maiden, S.V. Jones, S. Manning, J. Greenwood, and L. Renou, "Model-Driven Requirements Engineering: Synchronising Models in an Air Traffic, Management Case Study", *Advanced Information Systems Engineering*, LNCS 3084, Springer, 2004, pp. 368-383

[18] H. Mouratidis, P. Giorgini, G. Manson, and I. Philp. "Using Tropos methodology to Model an Integrated Health Assessment System", *Proceedings of the 4th International Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS-2002), Toronto-Ontario, May 2002

[19] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis, "Telos: Representing Knowledge About Information Systems" *ACM Transactions on Information Systems 8(4)*, October 1990

[20] A. Susi, A. Perini, P. Giorgini, and J. Mylopoulos. "The Tropos Metamodel and its Use". Journal Informatica 29(4), pp. 401-408, 2005

[21] A. Sutcliffe, and A. Gregoriades, "Validating functional system requirements with scenarios", *IEEE Joint International Conference on Requirements Engineering 2002 (RE 2002)*, Essen, Germany, pp. 181-188.

[22] UML 2.0. http://www.uml.org/#UML2.0

[23] Yahoo Groups. http://br.groups.yahoo.com/

[24] Z. You, "Using meta-model-driven views to address scalability in i* models", *Master of Science thesis*, Graduate Department of Computer Science, University of Toronto, Toronto, Canada, 2004

[25] E. Yu, "Modelling Strategic Relationships for Process Reengineering", *PhD Thesis*, Graduate Department of Computer Science, University of Toronto, Toronto, Canada, 1995

[26] E. Yu, "Agent-Oriented Modelling: Software versus the World Agent-Oriented Software Engineering", *AOSE-2001 Workshop Proceedings*. LNCS 2222. Springer Verlag. pp. 206-225