A SERVICE-ORIENTED FRAMEWORK FOR MAS MODELING

Wautelet Yves, Achbany Youssef and Kolp Manuel
Info. Syst. Research Unit (ISYS), University of Louvain, Belgium

Keywords: Requirements Engineering, service oriented modeling, i*, multi-Agent system.

Abstract: This paper introduces an analysis framework made of different models proposing complementary views. First view - the strategic services diagram (SSD) - is the most aggregate knowledge level; it uses services for representing the system to be in a global manner. Second view offers a more detailed perspective of the agents involved into the services using traditional i* Strategic Dependency (SDD) and Strategic Rationale Diagrams (SRD). Finally, the third view - the Dynamic Services Hypergraph (DSH) - offers a dynamic approach of services realization paths at various Qualities of Service (QoS). Using those models at analysis level is interesting in a model driven software development approach. Indeed, the project stakeholders can share a common vision through a lookup at the services the system has to offer, at the depender and dependee agents for the services constituents and at the different realization paths and their QoS. The framework also offers the fundamental elements to develop a complete model driven software project management framework. It can be considered as a startup for a global broader software engineering method.

1 INTRODUCTION

Multi-Agent Systems (MAS) have become an important and promising research area in computer science in general and in software engineering in particular. Applying agent technology to huge enterprise information systems engineering is nevertheless seldom done. One may wonder where this report comes from. MAS development is maybe not mature enough to be applied in large industrial software development. Indeed, most of the research is focused on developing particular aspects of modeling, specifying, designing or implementing collaborating agents. Global frameworks for MAS model driven development and software project management are, until now, seldom approached in literature. Researchers need consequently to understand what dimensions are required to bring a technology to be used on a large scale and to fill the gap between the research field and its practical application.

This paper is part of this effort: on the basis of the weaknesses identified in the i*/Tropos MAS development methodology, a generic framework for organizational modeling and requirements engineering using higher abstraction level elements: the services. The importance of distinguishing the system behavior at such level will be characterized and a set of models allowing services representation through multiple complementary views will be introduced. This framework is composed of a static high level services view, of a static and a dynamic decomposed services views.

The paper is structured as follows, Section 2 points the limitations of the traditional Strategic Dependency and Strategic Rationale Diagrams for model driven software development. Section 3 points to the use of a framework based on services that can be extended for software project management. Section 4 formalizes the new diagrams included into the framework through the use of a meta-model. Section 5 concludes the paper.

2 PROBLEM STATEMENT

Some modern software development methodologies are said to be *Model Driven*. In such methods, the entire development process is deduced from the features modelled in high level diagrams. Object-oriented development methodologies such as the Unified Process (IBM, 2003; Jacobson et al., 1999; Jacobson and Bylund, 2000; Kruchten, 2003) are for example said to be *use case driven*. This means that the entire devel-

Yves W., Youssef A. and Manuel K. (2008).
A SERVICE-ORIENTED FRAMEWORK FOR MAS MODELING.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - ISAS*, pages 120-128
DOI: 10.5220/0001708101200128
Copyright © SciTePress

opment process is driven by the use cases identified in the analysis phases. Moreover effort estimation techniques as Use-Case Points (Schneider and Winters, 1998) directly use high level diagram elements - the use cases - as fundamentals for effort estimation. Such model elements called *scope elements* are consequently usefull not only to share a common high level vision with stakeholders, but also to estimate the project effort on a non redundant basis, to evaluate related risks, to fix an acceptable level of quality, etc. Classical i*/Tropos developments using traditional Strategic Dependency (SD) and Strategic Rationale (SR) diagrams (Yu, 1995) for organisational modelling and requierements engineering do unfortunately not possess ideal elements to drive the development process, this will be shown in this section.

2.1 Related Work

As pointed out earlier, considerable work has been done in agent-oriented software development methodologies. The traditional i* modeling framework has however been poorly criticized and weaknesses as the lack of higher abstraction level elements has been only recently pointed out in literature (Pastor et al., 2007). Similarly, considerations on software project management using Tropos or other MAS methodologies are not easy to find in MAS literature. In the same way, even if some attempts such as AUML exist, no dynamic dimension implying i* concepts has been proposed and formalized. We will reference here three approaches, each one addressing one of the previously mentioned aspects. We will also briefly comment our view in regard to the aspects described in those papers.

- The process followed in this paper has been strongly inspired by the concepts introduced in (Estrada et al., 2006; Pastor et al., 2007). Those papers define a series of features for evaluating the i* framework on the basis of case studies. The empirical study addresses the weaknesses of the approach and points to the use of higher level abstractions called business services into i* developments. Our conclusions are rather the same and we take their approach as a starting point for multiple view modeling framework.
- (Dubois et al., 2007) proposes to model threats and vulnerabilities directly into SD and SR models and relates them directly with actors and goals. We also propose a framework where threats (and even opportunities) are incorporated into high level models but we relate then to services not in SD and SR diagrams. Furthermore operational solutions are incorporated to keep/improve quality of service.
- (Mouratidis and Giorgini, 2007) adresses i* security limitations and claims for the adoption of secure Tropos, an extention to the Tropos process incorporating security issues. Secure Tropos offers extentions for modeling requirements with security constraints. Moreover, in that process, security is an issue only considered at late requirement stage, the methodology partially identifies conflicts between security and other requirements. Our approach does not model security as constraint but identifies the potential threats to determine the most risky issues at the earliest stages of the project.

2.2 The Lack of Scope Elements

One of the main problems in using the Tropos software development methodology is to find an adequate scope element to drive the software process.

Actors from the SD and SR models are involved in dependencies of different intentional elements. The goal is the only candidate as scope element since it represents the most abstract functional issue. A complete description of these elements can be found in (Yu, 1995). However, goals are not considered as the ideal scope element candidate since:

- The atomicity of the goal is not clearly defined so that a goal decomposition can involve atomic goals part of higher level goals consequently (i) a great amount of goals – too many for keeping an easily understandable vision of the software project – can be present for a huge software projet; (ii) the goals are often at different levels of atomicity so that they can be redundant which is a major drawback to keep the simplest vision possible;
- The distinction between a task and a goal is not always trivial; different persons can model similar behavior by interchanging those different concepts.

Consequently, we need an element located at sufficiently high level of abstraction to represent a service the agents should provide in a non redundant way. This element should provide a clear vision of the project, allow to identify environmental threats for risk management, time management, etc. This element is called a *service* and is depicted in the next section.

2.3 Contributions

Compared to the approach followed into (Estrada et al., 2006; Pastor et al., 2007), we provide:

- A complete multiple view analysis framework for service-oriented modeling using i*.The framework also offers the fundamental elements to develop a complete model driven software project management framework. It can be considered as a startup for a global broader software engineering method;
- A dynamic model for representing services realization at defined quality of service level;
- Enhancements into the Strategic Services Model to model system threats;
- A case study.

3 A UNIFIED-FRAMEWORK FOR AGENT-ORIENTED REQUIREMENTS ENGINEERING

This section introduces the services approach and depicts the framework in an informal manner.

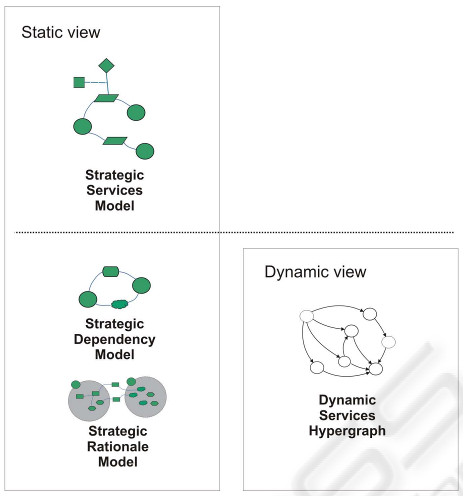
3.1 The Service Approach

In (Estrada et al., 2006; Pastor et al., 2007), the authors describe business services as "*an abstract set of functionalities that are provided by a specific actor*", they also specify that "*an actor can be an organizational entity . . . that uses or offers services*".

Services will be formalized in Section 4.2. In our approach, they present the following properties:

- they represent the highest abstraction level entities of the project, a service can encapsulate a large amount of atomic agregate goals or tasks;
- they are independent but complementary to each other in the sense that they do not overlap, but represent the whole of the system functionalities;
- they involve at highest level two super actors that can be refined in multiple sub actors at lower levels;
- their realization induces of different quality of service (QoS) areas.

Moreover, we also propose to add concepts to model environmental elements addressing a threat – potentially lowering QoS – or an opportunity potentially raising QoS. Once identified, we can also represent in a generic and in a more detailed manner, the measures that should be taken to fight against the threats (to maintain QoS) or to put the opportunities into practice (to raise QoS).


Static view
Strategic Services Model
Strategic Dependency Model
Strategic Rationale Model
Dynamic view
Dynamic Services Hypergraph
Figure 1: A Multiple View of the Framework.

3.2 The Framework

To fix the weaknesses pointed out previously, a service-oriented framework for organizational modeling will be introduced. This framework is made of two complementary views represented by four different diagrams. Figure 1 depicts the framework diagrams hierarchy.

The first view called the *static view* allows to represent the services at two complementary abstraction levels:

- the highest level is materialized through the *Strategic Services Diagram* (*SSD*) described and formalized in section 4.1. This level provides a synthetic description of the services provided at defined QoS by the organization and that will be operationalized into the application. Environmental factors influencing positively or negatively the provided services as well as means to operationalize them are represented here. This diagram can be used as a reference vision document for stakeholders as well as a basis for iteration planning, risk management, quality management, effort planning, etc.(see Section 3.3);
- the lowest level is materialized through the traditional Strategic Dependency and Strategic Rationale Diagrams as described in (Yu, 1995). This level provides a more detailed description of the

tasks, goals and resources involved in the achievement of all the services.

The second view called the *dynamic view* allows to represent the service achievement through the use of the *Dynamic Service Hypergraph* (*DSH*) described and formalized in Section 4.2. The goals and tasks of the multiple agents involved are sequentialized in the graph to represent possible paths for service realization at given QoS.

Taken as a whole, these views allow to represent the multiple aspects of the services the system should offer.

3.3 The FaMOs Framework - A Project Management Perspective

As described previously, one of the main goals of the framework is to allow model driven software development. The main advantage of this type of approach is to furnish structured developments methodologies to software professionals. The framework was conceived to provide this kind of achievement but also furnish a broader range of possibilities.
Those services include:

- **Risk Management.** By allowing modeling services threats and the operational solutions designed to maintain QoS when those occur, the framework includes some essential features for risk management. The framework includes different levels of detail (abstraction) for modeling those features.
- **Quality Management.** Quality indicators are present through the use of a QoS-level so that quality benchmarks for each service can be fixed early into the project. Furthermore, through the "services opportunities", improvement factors that, once operationalized, allow to higher QoS can be modelled at different levels. Finally, the framework constitutes an extension of traditional i*/Tropos modelling diagrams so that softgoal modeling into the Strategic Dependency and Strategic Rationale Diagrams remains possible.
- **Time Management.** Using the services as high level abstractions and the DSH for the service complexity evaluation, an approximate service effort can be computed (as for example in the Use Case Points methodology (Anda et al., 2002; Schneider and Winters, 2001)). On the basis of this evaluation, a project planning using waterfall or better an iterative development life cycle can be created. By including the number of human resources, their role, their wages, etc. a project cost evaluation can also be computed. The precision of those evaluations will successively increase from a project to another on the basis of the empirical statistics collected during the previous ones.

Due to a lack of space, this paper will only focus on the models formalization. (Wautelet et al., 2007) introduces a case study to illustrate the use of the framework. it only uses elements of risk and quality management but do not involve time and software process management. A complete presentation of the project management capabilities of the FaMOs-framework is currently under development.

4 SERVICE DRIVEN AGENT MODELING: A FORMALIZATION

To drive the business and user services acquisition, we propose a meta-model which provides modeling elements relevant for specifying both strategic and operational aspects of the organizational context in which the future information system will be deployed. This meta-model is made of *meta-concepts* (e.g., "Actor", "Dependencies", "Services", etc.), *meta-relationships* relating meta-concepts (e.g., "Performs", "Operationalize", "Act", etc.), *meta-attributes* of meta-concepts or meta-relationships (e.g., "Probability_of_happening", "Improvement_rate", etc.), and *meta-constraints* on meta-concepts and meta-relationships (e.g., "An actor occupies a position if and only if that actor possesses all the capabilities required to occupy it").

4.1 Strategic Services Model

The Strategic Services Model (or SSM) supports the acquisition, representation and reasoning about the services that should be provided by the agents of the application.

Figure 1 depicts the SSM. *Actors* are intentional entities used to model people, physical devices or software systems that are processor for some actions. The inheritances from the *Actor* into *Position*, *Agent* and *Role* as well as the linking with the *Dependency* class have been inspired by the *Tropos* metamodel presented in (Susi et al., 2005). A Position can cover 1 to n Roles, an Agent can play 0 to n Roles and can occupy 0 to n Positions. Actors achieve some *Services* which are functionalities they offer to others in order to fulfil a portion of their goals and tasks (for further formalization see section 4.2). The Service is always achieved through an Actor Dependency. The

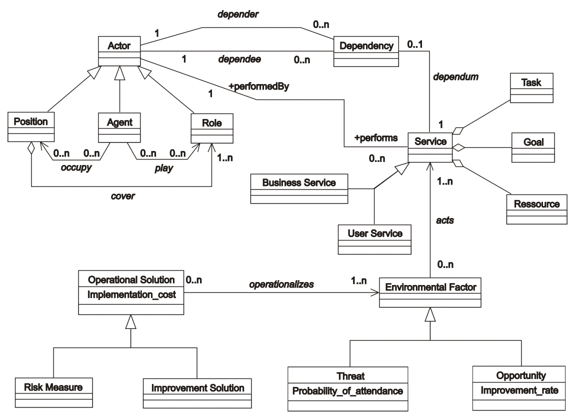


Figure 2: Strategic Services Diagram: A Meta-Model.

later relates two Actors: a *depender* and a *dependee* as well as a *dependum*, the Service.

Since the aim of the diagram is to offer a high level view of an under development enterprise information system, there are basically two types of services that need to be modelled: Business Services – business processes achieved into the company business domain – as well as User Services – services provided to the end user.

An *Environmental Factor* can be an external *Threat* for the adequate fulfilment of the service (both in terms of achievement and quality of service degradation) or an *opportunity* that can potentially raise the QoS. A service and its QoS are operationalized into a digraph as depicted in section 4.2. Moreover a *Probability_of_happening* is associated to each Threat as well as an *Improvement_rate* to each Opportunity. Those features will be useful for the Threats and Opportunities prioritization into project planning.

An *Operational Solution* operationalizes an *Environmental Factor*. In other words Threats and Opportunities are external factors that can lower or raise QoS. To fight against a Threat or to benefit from an opportunity measures need to be taken. Those can be represented through the *Operational Solution*. The later is specialized into the *Risk Measure* and the *Improvement Solution* respectively aimed to model the operational solution putted into practice to fight against a threat and to take benefit of a potential opportunity. Note that once again an element (the operational solution) distinguished at SSD level induces a higher level of abstraction that will be refined further into the framework through SD/SR Models and DSH.

The elements of the SSM are instantiated to provide a *Strategic Service Diagram* (or *SSD*). An SSD presents on the highest aggregation level of the framework the services provided by each agent as well as the potential threats and opportunities they are facing. It also supports the identification of operational solutions that can be taken to fight against the Threats potentially lowering QoS and to take benefits of Opportunities potentially raising QoS. It shows the potential problems that will be encountered when attempting to achieve Services and, with the guidance of the methodology, helps operationalizing them. By essence, the SDD helps to understand the purpose of the software project in terms of Services, the problems (i.e. Threats) they can face, the potential improvements (i.e. Opportunities) and the social structure (i.e. the Dependencies) which govern Actor interactions.

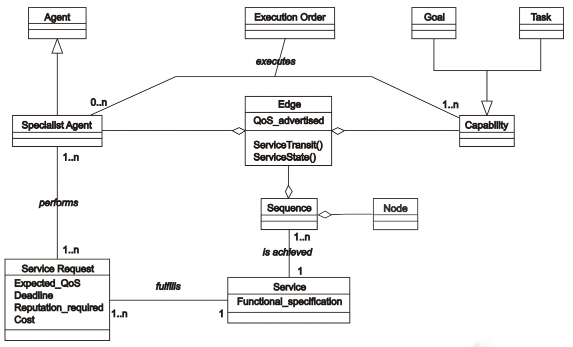


Figure 3: Meta-Model of the Dynamic View.

4.2 Dynamic Service Model

In this section, we will bring the *Dynamic Service Model* (*DSM*) to further formalization Figure 3 depicts the DSM. As the only properties of the agent relevant for this dynamic view are the capabilities in terms of tasks and goals it can execute along with its advertised quality values, an agent will be defined as follows[1].

Definition 1. *A tuple* $\langle\{(cp_i, q^a_{cp_i}), \ldots, (cp_{i+m}, q^a_{cp_{i+m}})\}, Ag^a\rangle$ *is called an agent a, where* cp_i *is a capability. The agent advertises its capability to execute a task or goal to the QoS level and cost* $q^a_{cp_i}$*. The advertised level is a vector of QoS- and cost-property and value pairs following a QoS ontology.* Ag^a *is assumed to contain all additional properties of the agent irrelevant for the present discussion, yet necessary when building a MAS.*

Format and content of Ag^a will depend on the agent programming language being used. A *Capability* is the generalization of *Goal* and *Task*. As it is unimportant to know that an agent can perform more

[1]Note that since, in this view, the service realization is documented we refer to agent rather than to actors as in the SSD.

than one capability, a specialist agent is defined over a single capability.

Definition 2. *$\langle a, cp_i, q^a_{cp_i} \rangle$ associating a capability cp_i to a quality level $q^a_{cp_i}$ advertised by the agent a is a specialist agent a^{SA}_i. The agent must be capable of perfoming the capability: $\forall a^{SA}_i = \langle a, cp_i, q^a_{cp_i} \rangle, (cp_i, q^a_{cp_i}) \in a$.*

Any agent a that can accomplish $m > 1$ capabilities can also be seen as a set of specialist agents: $\{a^{SA}_i, \ldots, a^{SA}_{i+m}\}$. It is necessary to have a more precise idea of how capabilities and services are conceptualized.

Definition 3. *A capability cp_i is $\langle cp^{pre}_i, \tau_i, cp^{post}_i \rangle$, where cp^{pre}_i describes the capability precondition, τ_i is a specification (in some language) of how the agent is to execute the capability, and cp^{post}_i describes the conditions true after the capability is executed. Capabilities belong to the set $\mathbb{CP}$.*

Definition 4. *$\langle s^{\iota}_j, s^N_j, s^E_j, servTransit_j, servState_j \rangle$ is a service s_j, where s^{ι}_j provides the details of the functional specification of the service, (s^N_j, s^E_j) defines a directed acyclic graph. Nodes represent states, and edges transitions between states. The two functions label nodes and edges with capability information: $servTransit_j : s^E_j \longmapsto \mathbb{CP}$ is a partial function returning the capability for a given edge in the graph, while $servState_j : s^N_j \longmapsto \{cp^{pre}_i\}_{cp_i \in \mathbb{CP}} \cup \{cp^{post}_i\}_{cp_i \in \mathbb{CP}}$ maps each edge to a condition from the set of all capability preconditions (i.e., $\{cp^{pre}_i\}_{cp_i \in \mathbb{CP}}$) and postconditions (i.e., $\{cp^{post}_i\}_{cp_i \in \mathbb{CP}}$). The capability specified on an edge must have the precondition and postcondition corresponding to conditions given, respectively, on its origin and its destination node.*

A service can therefore be understood as a process, composed of a set of Tasks and Goals (i.e., these are specializations of capability) ordered over the graph representing the service. The functional specification of the service, i.e., s^{ι}_j is not of interest here, but involves in practice, e.g., a specification of interfaces, and other implementation considerations. Requesting a service requires the specification of expected QoS, in addition to a deadline for providing the service, minimal level of reputation for agents that are to participate in service execution, the maximal monetary cost, and explicit user preferences on agents to select (e.g., users may prefer globally the services of some providers over others, regardless of actual performance—this may occur with preferential threatment resulting from environment constraints such as, e.g., legal constracts on cooperation between organizations and/or individuals).

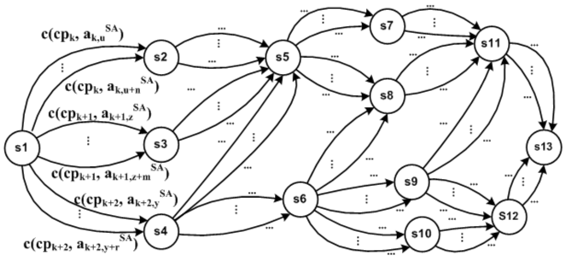

c(cp$_k$, a$_{k,u}^{SA}$)
c(cp$_k$, a$_{k,u+n}^{SA}$)
c(cp$_{k+1}$, a$_{k+1,z}^{SA}$)
c(cp$_{k+1}$, a$_{k+1,z+m}^{SA}$)
c(cp$_{k+2}$, a$_{k+2,y}^{SA}$)
c(cp$_{k+2}$, a$_{k+2,y+r}^{SA}$)
s1
s2
s3
s4
s5
s6
s7
s8
s9
s10
s11
S12
s13

Figure 4: The service depicted as a labeled hypergraph.

Definition 5. *A service request $\hat{s}_j$ is $\langle s_j, s^{QoS}_j, s^D_j, s^R_j, s^{cost}_j, s^{pref}_j \rangle$, where:*

- *s_j is the service to provide.*
- *s^{QoS}_j specifies expected qualities and their required level. Its definition follows a QoS ontology, such as, e.g., the FIPA QoS ontology specification (for Intelligent Physical Agents, 2002). Whatever the specific QoS ontology, expected qualities are likely to be specified as (at least) $s^{QoS}_j = \langle (p_1, d_1, v_1, u_1), \ldots, (p_r, d_r, v_r, u_r) \rangle$, where:*
  - *p_k is the name of the QoS parameter (e.g., connection delay, standards compliance, and so on).*
  - *d_k gives the type of the parameter (e.g., nominal, ordinal, interval, ratio).*
  - *v_k is the set of desired values of the parameter, or the constraint $<, \leq, =, \geq, >$ on the value of the parameter.*
  - *u_k is the unit of the property value.*
- *s^D_j is a deadline, specified as a natural.*
- *s^R_j specifies minimal levels of reputation over task quality parameters that any agent participating in the provision of the given service must satisfy. It is not necessary to specify reputation for all qualities over all tasks, selective reputation expectations are admitted.*
- *s^{cost}_j is the maximal monetary cost the user requesting the service is ready to pay to obtain the service.*
- *s^{pref}_j is a set of expressions that constrain the pool of potential agents to which the service mediator can allocate tasks.*

By conceptualizing the service as suggested in Def.4, the service is thus mapped onto a directed hypergraph G where each node is a step in service provision and an edge in G corresponds to the execution of a capability cp_k by a specialist agent $a^{SA}_{k,u}$, where u ranges over specialists that can execute cp_k according to the criteria set in the service request. Each individual edge is labelled by a function of the criteria set,

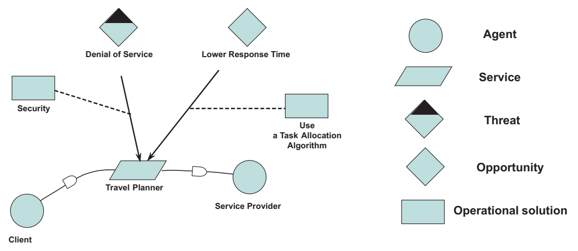


Figure 5: Strategic Services Diagram - Travel Planner.

$c(cp_k, a_{k,u}^{SA})$, representing the QoS advertised by the specialist $a_{k,u}^{SA}$ for performing the capability cp_k. Each path from the starting node (e.g., node s1 in Figure 4) to the destination node (node s13 in Figure 4) thus corresponds to a sequence of capabilities ensuring the completion of the service within the prescribed QoS. The model thus assumes that there are alternative ways for completing the service. The topology of the graph—i.e., the node structure and the capabilities associated to edges between the nodes—is provided by the designer through the service definition, so that the graph is a graphical model of the different ways the service can be performed as a sequence of capabilities.

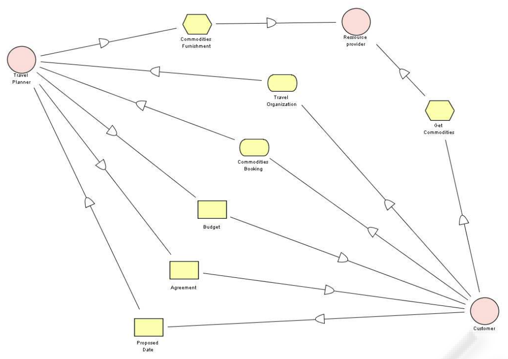


Figure 6: Strategic Dependency Diagrams: Travel Planner.

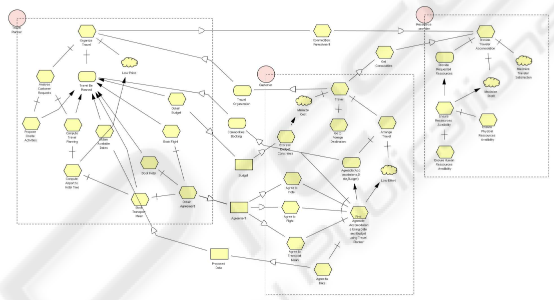


Figure 7: Strategic Rationale Diagrams: Travel Planner.

5 CASE STUDY

In this section, the framework is applied to a simplified case study: the travel planning service. Travel planning regroups all the activities provided by a service provider destined to organize a business or holiday trip. Those activities include the booking of facilities such as an hotel, a flight, a car, a taxi, etc. The different dimensions of a service description and fulfillment will be envisaged, their complementarities will be illustrated.

5.1 The Strategic Services Diagram

Figure 5 depicts the Travel Planner's Strategic Services Diagram.

In this service, the Customer addresses to a Travel Planner for its travel organization. The Travel Planner is in charge of the Commodities Booking to the Resource Provider (which can be an hotel resort, a flight company, a taxi company, etc. i.e. all the companies furnishing commodities to the Customer) for the destination of the Customer on the basis of its Budget, Proposed Date and Agreement.

5.2 Strategic Dependency Diagram

Figure 6 illustrates a strategic dependency diagram of the Travel Planner system's requirements. The system is intended to support a customer and a travel planner in travel related activities. Three actors are introduced into this diagram: Travel Planner is responsible for the organization of the travel and the necessary reservations. Customer gives these preferences concerning its travel which he wants to plan. Resource provider deals with providing the various resources helping to planning.

A strategic rationale diagram of the Travel Planner system's requirements is depicted in Figure 7. This Figure details the various actors introduced into the strategic dependency diagram.

5.3 Dynamic Services Hypergraph

Figure 8 represents fulfilment paths for the service Travel Planner. Each node is a step in service provision and each edge corresponds to the execution of a task t_k by a specialist agent $a_{k,u}^{SA}$, where u ranges over specialists that can execute t_k according to the criteria set in the service request. A function of the criteria set, $c(t_k, a_{k,u}^{SA})$, labels each edge and represents the QoS advertised by the specialist $a_{k,u}^{SA}$ for performing the task t_k. Note that different paths offering different

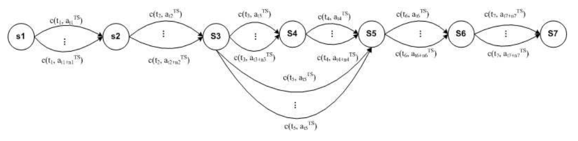


Figure 8: Strategic Dynamic Diagram - Travel Planner.

QoS are available. Indeed, path $< t_1, t_2, t_3, t_4, t_6, t_7 >$ offers alternative ways of fulfilling the service. The following list gives a descritpion of each task used in the Hypergraph.

- t_1:The customer proposes a range of dates fitting for his travel.
- t_2:The customer expresses the maximal budget he is able to allow for his travel.
- t_3:The Resource Provider gets a list of proposition for commodities on the basis of the customer constraints.
- t_4:The customer selects its preferred commodities.
- t_5:On the basis of the constraints defined, the Service provider organises a trip plan for the customer. Finally, the Service Provider books the commodities (i.e., flight, hotel, ...).
- t_6:The customer accepts a list (date, budget, flight and hotel) proposed.
- t_7:The Resource Provider provides the resource to the customer.

5.4 Lessons Learned

This simplified case study allowed illustrating:

- The use of different levels of knowledge:
  - A first level materialized through the SSD where stakeholders can have an aggregate view of the services the system offers as well as the threats and opportunities;
  - A second level materialized through the SDM, SDM and DSD to decompose the service in a static and dynamic manner. In this view analysts can see respectively the actors involved as well as the multiple realization paths.
- The alternative paths in the hypergraph with different QoS.

To demonstrate the real contribution of the framework it should however be applied to a broader case study implying multiple services. Due to the lack of space this was not possible here.

6 CONCLUSIONS

As identified several times in literature, the Tropos methodology in general and i* framework in particular lacks proper elements for driving the soft process. Indeed, even if the qualities and advantages of the strategic dependency and strategic rationale models has been recognized, redundancy, various levels of aggregation and alternative modeling ways make them hard to use in the context of model driven software development. That is why a broader framework for analysis stages of i*/Tropos is needed.

The framework furnishes multiple views for model driven software development. This framework includes a high level service view allowing to identify in a non redundant way the services that must be fulfilled by the system as well as elements raising or lowering QoS. Such framework is much broader than only allowing organisational modeling and requirements engineering. Indeed the framework can be extended for risk, quality or time management. The aim is consequently to create a complete software project management dimension for Tropos based on the models of the framework; this work is in progress; a CASE Tool is also under development.

Future research directions will include systematic forward engineering rules based on the models in the framework. This work has partially been done in (Do et al., 2003) but needs to be extended to include the benefits of dynamic models as the Strategic Services Hypergraph. Such systematic approach will provide guidelines to software designers to facilitate the framework adoption.

REFERENCES

Anda, B., Angevik, E., and Ribu, K. (2002). Improving estimation practices by applying use case models. *PROFES*, pages 383–397.

Do, T., Kolp, M., and Pirotte, A. (2003). Social patterns for designing multi-agent systems. *In Proceedings of the 15thInternational Conference on Software Engineering and Knowledge Engineering (SEKE 2003), San Francisco, USA, July*.

Dubois, E., Rifaut, A., and Mayer, N. (2007). Improving risk-based security analysis with i*. *in Giorgini P., Maiden N., Mylopoulos J., Eric Yu editors, Social Modeling for Requirements Engineering, in Cooperative Information Systems series, MIT Press*.

Estrada, H., Rebollar, A., Pastor, O., and Mylopoulos, J. (2006). An empirical evaluation of the i* framework in a model-based software generation environment. *Proceedings of CAiSE*, pages 513–527.

for Intelligent Physical Agents, F. F. (2002). Fipa quality of service ontology specification. *Doc.No. SC00094A. FIPA (http://www.fipa.org)*.

IBM (2003). The rational unified process. *Rational Software Corporation, Version 2003.06.00.65*.

Jacobson, I., Booch, G., and Rumbaugh, J. (1999). The unified software development process. *Addision-Wesley*.

Jacobson, I. and Bylund, S. (2000). The road to the unified software development process. *Cambridge University Press*.

Kruchten, P. (2003). The rational unified process : An introduction. *Longman (Wokingham), Addison-Wesley, December*.

Mouratidis, H. and Giorgini, P. (2007). Extending i* and tropos to model security. *in Giorgini P., Maiden N., Mylopoulos J., Eric Yu editors, Social Modeling for Requirements Engineering, in Cooperative Information Systems series, MIT Press*.

Pastor, O., Estrada, H., and Martínez, A. (2007). The strengths and weaknesses of the i* framework: an experimental evaluation. *in Giorgini P., Maiden N., Mylopoulos J., Eric Yu editors, Social Modeling for Requirements Engineering, in Cooperative Information Systems series, MIT Press*.

Schneider, G. and Winters, J. (1998). Applying use cases - a practical guide. *Addison Wesley Longman, Inc.*

Schneider, G. and Winters, J. (2001). Applied use cases. *Second Edition, A Practical Guide, Addison-Wesley, Reading, MA*.

Susi, A., Perini, A., Giorgini, P., and Mylopoulos, J. (2005). The tropos metamodel and its use. *Informatica, 29(4):401–408*.

Wautelet, Y., Achbany, Y., and Kolp, M. (2007). A unified framework for multi-agent services-driven organisational modeling. http://www.isys.ucl.ac.be/staff/yves/articles/FAMOS07.pdf

Yu, E. (1995). Modeling strategic relationships for process reengineering. *PhD thesis, University of Toronto, Department of Computer Science, Canada*.