Applied Medical Informatics
Original Research Vol. 24, No. 1-2/2009, pp: 12 - 33

A Health-care Application of Goal-driven Software Design

Maria-Eugenia IACOB*, Diederik ROTHENGATTER, and Jos van HILLEGERSBERG

1 University of Twente, School of Management and Governance, Department of Information Systems & Change Management, Capitool 15, Postbus 217, 7500 AE Enschede.
E-mail(s): m.e.iacob@utwente.nl; d.c.f.rothengatter@utwente.nl; j.vanhillegersberg@utwente.nl

* Author to whom correspondence should be addressed; Tel. +31-(0)53-489 4134, Fax. +31-(0)53-489 2159.

Abstract: In this paper we focus on goal engineering by addressing issues such as goal elicitation, specification, structuring and operationalisation. Specification of business goals is regarded as a means to raise the level of abstraction (and automation) at which business logic is incorporated in model driven software design in the context of service oriented architectures. More specifically, the proposed goal modelling approach consists of an abstract syntax (metamodel) and a concrete syntax (graphical notation) for the specification of business goals. We also proposed a framework for the goal-driven design of service-oriented software applications. In particular, we illustrate our approach by means of a case study carried out in the healthcare sector and we explain the role business goals (operationalised in the form of business rules) can play in software design. This research also outlines a number of areas that have significant research potential.

Keywords: Goal modelling; Business Rules; Service Oriented Architecture; Model-driven design.

Introduction

Goal elicitation, modelling and analysis are often critical and complex processes. Business people need therefore ways to express these goals as clearly and precisely as possible, both for their own understanding and for communication with other stakeholders, such as requirements engineers, process designers, business strategists and software developers. To date, there is no de facto standard notation/language for describing goals, and they are often described in myriad of ways going from informal and highly abstract pictures or text that lack a well-defined formal semantics to very precise predicate logic formulae or mathematical functions. This leads to misunderstandings, and makes it very difficult to provide tools for visualisation and analysis of goals. It also makes impossible for organisations to trace back to business processes the causes of success or failure in reaching organisational goals. Consequently, we believe that goal modelling and analysis is critical in early design phases during the identification of requirements for information systems that must support their achievement. Furthermore, in this paper we argue that in the context of model-driven design, goal models may become an integral part of the design of these systems.

Therefore, in this research we aim to develop a framework for goal engineering and a Goal Modelling Language (GML) that has the following characteristics:

- It should be aligned with well-known existing goal modelling languages and methods.
- It should have a solid formal basis expressed in terms of an abstract syntax (i.e., a meta-model) and concrete syntax (i.e., a graphical notation). Preferably, the concrete syntax should be aligned with existing notations for goal modelling and with existing design languages notations.
- It should be easy to integrate with other special purpose specification languages such as business rule specification languages process modelling languages, enterprise architecture languages, and software design languages.

- It should be attractive for a business audience, thus simple to learn, understand and use. Therefore, the aim is to design a language that has an intuitive notation, a minimal set of modelling concepts and relationships, and is, nevertheless, rich enough to have sufficient expressive power.
- It should facilitate the documentation, communication, analysis and reasoning about goals.

In particular, we aim to illustrate and validate our goal modelling approach and framework in practice by means of a case study carried out in the healthcare sector. This is meant to demonstrate that (1) the defined language is expressive enough to capture various aspects of an organisation's goals but also to demonstrate that (2) our idea that goals are not just requirements for the service-oriented software design but can become a part of it is practically feasible.

The paper is organised as follows: in the following section we briefly discuss the problems we address in this paper. Then we provide a few classification criteria for goals and we investigate the existing standards and languages for the formal specification of business goals. In the next section we introduce our approach for goal modelling and we define GML's concrete and abstract syntax. This is followed by a presentation of our framework for combining goals with model-driven design of service oriented systems. The main goal is to analyse in terms of method, specification languages and tools the extent to which business goals can be incorporated in design models and become eventually, at the platform specific level, an executable way of specifying business logic. This is illustrated by means of a case carried out at a large Dutch hospital. Finally, in last section we draw some conclusions and point out future work.

Goals: Definition, Classifications and Modelling

In this section we will give an overview of the topics central in this research, namely the goal concept, goal classification criteria and existing goal modelling approaches.

Definition and classifications

According to [1], goals are high-level objectives of a business, organization or system; they capture the reasons why a system is needed and guide decisions at various levels within the enterprise.

Goals are important in several respects. Goals drive the identification of system requirements to support them. Together with scenarios, they are the driving forces for a systematic requirements elaboration process. Thus, a goal refinement tree provides traceability links from high-level strategic objectives to low-level technical system requirements. Consequently, goal refinement provides not only a natural mechanism for structuring complex requirements documents for increased readability but also the right level of abstraction at which decisions makers can be involved for validating the choices being made during system design or for suggesting other alternatives overlooked so far. Goal models also provide a way to communicate system requirements to different stakeholders and to detect whether they have conflicting goals with respect to the future system.

Goals in organizations are considered to be abstract concepts, like the high level objectives of the business, organization, or system [2]. In this paper we define a goal as a *desirable state to be achieved by the composite system.* We could distinguish in the literature concerning goals, goal-based requirements engineering, and goal oriented analysis, several classifications of goals. The first classification to be discussed here is the distinction between *functional goals*, and *non-functional goals.* This distinction is proposed both by [3] and [4]. Functional goals are services a system delivers such as the registration of a patient, the sending of an invoice, or the assignment of a resource to a specific activity in a process. In most cases, functional goals can be assigned to a specific process, actor, or organizational structure. Non functional goals do not apply to a specific process, actor, or structure. This type of goals is of importance for the whole system. Examples are security, performance, or scalability. Both the functional and the non-functional goals can in principle be clearly defined, and thus are measurable. Another goal classification is the distinction between *hard goals* and *soft goals* [5], or between operational goals and strategic goals [6]. An operational goal can be accomplished by a given process. The strategic goal, however, cannot be achieved by a single process. As stated in [6], soft-goals only refer to a part of the domain, not directly to a given process. Soft-goals are used to specify high level qualitative objectives (e.g., customer satisfaction) that are difficult to measure. Hard-goals clearly define a concrete state/target an actor desires to achieve. For hard goals clear achievement criteria and (quantitative) measures are defined. Some authors equate this classification to the functional versus non-functional distinction (see [7,8]). However we assume in this taxonomy that hard goals are goals which can be determine by a measure, whereas soft goals have no clear-cut criteria as to whether they are satisfied. In the functional and non-functional classification, both types of goals are measurable. A third important classification is on the level of goal assignment. Goals can be assigned to a system or organization, to an actor in the system or organization, or to the relationship between two or more actors in the system or organization. In [8] this distinction is defined as *system goals* and *private goals*, in [5] goals are always defined between two or more actors in the system, and in [3] goals are most often identified on the system level.

Goal elicitation strategies

In an organization goals can be *elicited* in a bottom-up or top-down fashion. *Top-down elicitation* of goals can be regarded as refining and decomposing the system's goals. The top-down approach usually starts from analyzing the corporate strategy and translating it into a set of soft goals, which are gradually refined and decomposed into operational level goals [6]. *Bottom-up elicitation* of goals starts with the analysis of the individual autonomous actors/agents, identifies their "private" operational goals and aggregates them into more abstract and higher level goals that concern the whole organization/system. The former approach appears to require guess work and inventiveness, since there is no systematic way for refining high level goals into concrete quantifiable goals. Furthermore, high level enterprise goals do not always imply what should be the goal, for example, of a concrete low level process [2]. In the latter approach, the challenge resides in the fact that goals do not always comply with one another. Each stakeholder has different requirements and priorities. Very often these interests are conflicting. While the first type of approach is typical/suitable for organization models in which the control is imposed into a centralized and hierarchical fashion, the second approach is more suitable for organizational models in which the control is distributed over several units/departments. Nevertheless, common to both approaches is the fact that in the end both approaches result into a detailed decomposition of goals in the form of a goal tree in which leafs' granularity is sufficiently fine to allow their operationalisation.

Both strategies for goal elicitation - top-down and bottom-up -are incorporated in Goal-Oriented Requirements Engineering (GORE) methodologies. The top-down approach for example is implemented in the KAOS methodology [8], whereas the bottom-up approach is implemented in the TROPOS method [5].

As it will become clear from the presentation of a few goal modelling languages, three types of relations essentially form the basis of the top-down and bottom-up approaches: *decomposition, abstraction and operationalisation*. These relationships constitute essential mechanisms trough which during goal specification one can go, on one hand, from one level of abstraction to the following one (decomposition and abstraction), and, on the other hand, from goals specification to design specifications (operationalisation).

Elicitation by decomposition is an informal technique for finding out sub-goals and requirements by asking HOW questions about the goals which have been already identified [4,9]. Formal goal refinement patterns may also prove effective when goal specifications are formalized; typically, they help finding out sub-goals that were overlooked but are needed to achieve the parent goal. Elicitation by abstraction is an informal technique for finding out more abstract, parent goals by asking WHY questions about operational goals/descriptions already available [9]. Note also that refinement patterns when applied in the reverse way correspond to abstraction patterns that may produce more coarse-grained goals. Elicitation by operationalisation is a formal or informal technique to deriving pre-, post-, and trigger conditions (which are essentially constraints) on system operations so as to ensure the fulfilment of the terminal goals (i.e., the leafs) in a goal tree. The principle is to apply derivation rules whose premise match the goal under consideration [9]. Of particular interest for this research is the relationship between the (terminal) goals and constraint constructs (such as business rules controlling processes). As stated in [8]:

"the link between goals and constraints is captured in the Operationalization meta-relationship defined as follows:

```
Operationalization (C, G)
iff meeting constraint C is among the
operational ways to achieve goal G.
```

Thus, a constraint operationalising of a goal amounts to some abstract "implementation" of this goal. In general a goal can be operationalised through several alternative combinations of constraints; like Reduction, Operationalisation is an And/Or relationship. A meta-constraint here is that a goal operationalised into constraints may not be reduced further."

Existing goal modelling languages

Several language metamodels providing a conceptual foundation for the modelling of goals have been proposed in the literature. Below, we briefly describe three of the most representative ones – KAOS [8], Tropos [10] and BMM [11] and explain in which way we will further use these approaches in our research.

The ***KAOS*** approach [8] has three components: a conceptual model for structuring requirements models, with an associated textual specification language, a set of strategies for elaborating requirements models in this framework and an automated assistant to provide guidance in the acquisition process (automated support in following an acquisition strategy – built around a requirements database and a requirements knowledge base). The main constructs defined in the conceptual model are objects (agents, entities, events, and relationships), operations (performed by an agent and change the state of one or more objects), actions: a mathematical relation over objects, goals (a non-operational objective to be achieved by the composite system) and constraints (an operational objective to be achieved by the composite system). Goals are classified according to their pattern and their category. Five goal patterns are identified: Achieve, Cease, Maintain, Avoid, Optimize. Furhtermore, KAOS distinguishes between the following goal categories: Satisfaction goals, Information goals, Robustness goals, Consistency goals, Safety goals and privacy goals. The strong points of KAOS are a solid formal basis, wide recognition in the academic community and powerful support for goal analysis and reasoning. It is a framework that facilitates the reasoning about goal satisfaction and the systematic building of complete, conflict-free goal based requirements modelling [9]. Nevertheless, KAOS also fails to meet some of the requirements we set in the introduction for a goal modelling approach, namely no "official" graphical notation has been proposed [1] and it is not suitable for business audience, since it is too formal and requires a background in discrete mathematics and formal logic.

The ***Tropos*** approach aims to guide the development of agent-based software systems [5]. Tropos incorporates goal modelling concepts (such as agents, goals, plans) through the phases of software development. A pivotal role is assigned to requirements analysis when the environment and the system-to-be are analyzed. The methodology covers five phases: (1) *early requirements*: identification of stakeholders (with their objectives). Stakeholders are represented as agents, objectives as goals; (2) *late requirements*: requirements analysis to fit the system-to-be in its environment (including relation with actors in its environment); (3) *architectural design*: all system actors are introduced and assigned to subtasks; (4) *detailed design*: detailed design of system agents, including specification of communication and coordination protocols and (5) *implementation*: the Tropos specification is transformed into a skeleton for implementation (via mapping of the Tropos constructs to those of an agent programming platform). With respect to goal modelling, the Tropos language and metamodel identifies the following core concepts: actor (represents a physical agent, or a software agent, as well as a role or position), goal (in Tropos a distinction is made between hard goals and soft goals), dependencies (links between actors, indicating that one actor depends on another to attain some goal), plan (represents a way of satisfying a goal), resource (a physical of

[1] There exists a commercial requirements engineering tool that implements the KAOS methodology and proposes a graphical notation for KAOS as well (http://www.objectiver.com/).

informational entity), And/Or Decomposition (goal decomposition relation), means/end relationship specifies a means (in terms of a goal, a plan or a resource) to satisfy the goal, and contribution relationship (given a goal, the contribution relationship specifies the goals or plans or resources that can contribute positively or negatively to its achievement; the measure of the degree of contribution is expressed via the qualitative metrics +, ++, -, --). Compared to KAOS, Tropos has the advantage of having an intuitive graphical notation, which makes it suitable for business audience. Similar to KAOS, it has a solid formal basis and facilitates goal analysis. Furthermore, there are several of open source tools (e.g., TAOM4e [2]) available. The most notable disadvantages of the Tropos approach are the unpractical modelling technique (models grow quite rapidly and the overview is lost) and the weak/lack of support for constructs such as processes and business rules/constraints.

The third approach – the ***Business Motivation Model*** (BMM) is a meta-model and a standard for capturing business requirements that was finalized by the OMG in October 2007 [11]. More precisely, BMM is one of the several OMG standards developed by the Business Modelling and Integration Task Force (BMI-TF). Several BMM supporting tools are already available on the market. BMM focuses on capturing semantically rich requirements that are useful for business analysis, querying, impact analysis, change management, and business reasoning. Its main goal is to capture business requirements in such a way that it clearly explains why the business wants to do something, what it is aiming to achieve, how it plans to get there, and how it assesses the result. The main constructs of the meta-model are Ends (what the business wants to accomplish), Means (how the business intends to accomplish its stated ends), Assessment (who and how the means are assessed against the ends) and Influencers (who or what judges/influences the assessment. Each of these constructs is further refined/specialised into several other constructs, a few of which we wish to emphasize. The first construct, the Ends, designates things the enterprise wishes to achieve, namely Goals and Objectives. Among the Means there are things the enterprise will employ to achieve those Ends, for example, Mission, Strategies, Tactics, Business Policies, and Business Rules. The Influencers are internal and external factors that shape the elements of the business plans, and the Assessments are evaluations made about the impacts of such Influencers on Ends and Means (i.e., Strengths, Weaknesses, Opportunities, and Threats). Two important constructs that fall outside the four above-mentioned categories are that business process and organisational unit. Although, both are considered to be referenced elements defined externally (which supposedly fall outside BMM's scope), the BMM does include placeholders for them, to facilitate the integration with other existing and emerging Business Process standards.

BMM has several obvious advantages such as those of being an open standard, very appealing for business audience and easy to integrate with other OMG standards (e.g., SBVR [12], UML [13], etc.). Nevertheless, BMM also does not come with a standard graphical notation, it has a broader scope than just goal modelling and therefore it has too many concepts (some of which are unclear or overlap with each other), it has no strong formal basis and does not address at all goal analysis and reasoning issues.

We conclude this section with a comparison of these three goal modelling languages. The results of this comparison are presented in Table 1 and Table 2. These tables provide a structured overview of the defined concepts, and make the semantic differences between the languages explicit. For example, empty cells in the table illustrate that a particular language does not explicitly support a certain concept. The table may also provide an opportunity to reflect about the way each language defines its own concepts and thus to consider alternative definitions. This is also how we exposed the overlap between the three languages and extracted the essentials in order to create a basis for devising our GML meta-model.

[2] See also http://www.troposproject.org/tools.php.

Table 1. Concepts comparison: KAOS, TROPOS, BMM

INTEROGATIVE	CONCEPT	KAOS	TROPOS	BMM
Who	*actor*	*agent*	*Actor specialised as agent, position, role*	*organization unit*
Why	*goal*	*goal*	*goal*	*desired result*
	soft goal		*soft goal*	*goal*
	hard goal		*hard goal*	*objective*
	system goal	*system goal*		
	private goal	*private goal*		
How	*behaviour*	*action*	*plan*	*Course of action specialized as strategy, tactic;* *business process*
	constraint	*constraint*		*Directive, business rule, business policy*
What	*resource*	*entity*	*resource*	*asset specialised as resource and fixed asset*
When	*event*	*event*		

Table 2. Relationship comparison: KAOS, TROPOS, BMM

RELATION	KAOS	TROPOS	BMM
AndOr-Refinement	*ORreduction(goal, goal), ANDreduction(goal, goal)*	*AND/OR decomposition(goal,goal)*	*includes(desired result, desired result)*
Contribution	*conflict(goal,goal)*	*contribution(goal,goal)*	*amplify(goal,vision)*
Association	*responsibility(agent, constraint)* *concerns(goal,object)* *wish(agent,goal)*	*means-end analysis(goal, plan, actor, resource)* *wants(actor, goal)*	*responsible(organization unit, liability)* *responsible(organization unit, asset)* *defines(organization unit, end)* *defines(organization unit, course of action)*
Assignment	*perform(agent,action)*	*executes(actor, plan)*	*responsible(organization unit, business process)*
Operationalisation	*Operationalization (goal,goal)*		*supports_achievement_of (business rule, goal)* *is_derived_from(business policy, business rule)* *quantifies(objective, goal)* *makes_operative(mission,vision)* *implements(tactic,strategy)*
Enforcement	*ensuring(constraint, object,action)*	*Dependency(goal,actor1, actor2,plan,resource)*	*Governs(Businessrule/policy, business process)*

GML

Based on the comparison of the three goal modelling formalisms (KAOS, Tropos and BMM) and on the requirements formulated in the introduction we are proposing a new language GML that incorporates features of all three languages but also compensates most of their identified shortcomings. In Table 3 one may find a summary of the concrete syntax attached to GML together with the definition of the various language constructs.

Table 3. Summary of the concrete syntax

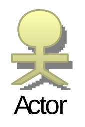
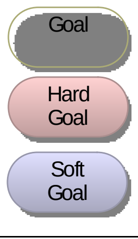
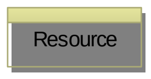
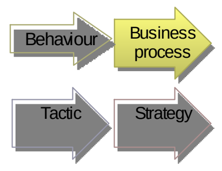
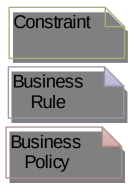
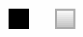
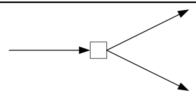
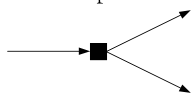
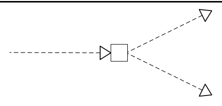
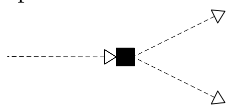
Name	Definition	Notation
Actor	An entity that has strategic goals and intentionality within the system or the organizational setting. An actor may represent a physical, social or software agent as well as a role, position or organisation [7]. Actors can be specialised as agents and/or roles.	
Goal	A goal is a strategic and/or tactical objective to be achieved by the composite system. These can be goals of organizational and human actors. The goal concept can be specialised into soft (non-operational) goals and hard goals. A goal has a type that can take one of the following values: system goal, private goal, functional goal, non-functional goal, etc. Besides the attributes inherited from the parent concept "goal", hard goals must be measurable, and norms must be specified in relation with them.	
Resource	A resource represents a physical or an informational entity [7]. A resource can be specialised as informational resource or physical resource.	
Behaviour	A (collection of logically related) unit(s) of internal activities, leading to a clear result (e.g., a set of products and/or services). Exactly one actor is responsible for those activities. A complex behaviour could be composed out of several sub-behaviours. Often different actors are responsible for these different sub-behaviours. For a consumer only the behaviour results (e.g., products and services) are relevant, while the required activities delivering them is merely a black box, hence the designation: internal. A behaviour can be specialised as business process, as a tactic or as a strategy.	
Constraint	A constraint is a statement that defines or constrains some aspect (e.g., information, behaviour or structure) of a system. A constraint can be further specialised as a business rule or as a business policy.	
And/Or junction	Part of a n-ary decomposition or operationalisation relation. A junction is not a separate construct defined in the GML metamodel.	
And/Or Decomposition relationship	And/Or decomposition-combines AND and OR decompositions of a root goal/behaviour into sub-goals/sub-behaviours, modelling a finer goal/behaviour structure.	 'or' decomposition  'and' decomposition
And/OR Operationali-sation relationship	A constraint operationalizing a goal indicates that the constraint is an abstract "implementation" of that goal (n.b., a constraint may be a business rule or a business policy). In general a goal can be operationalised through several alternative combinations of constraints. Like decomposition, Operationalization is an And/Or relationship. A meta-constraint here is that a goal operationalised into constraints may not be reduced further.	 'or' operationalisation  'and' operationalisation

Table 3. Continuation

Name	Definition	Notation
Contribution relationship	Given a goal, the contribution relationship is a directed relationship that specifies the goals or behaviours or resources that can contribute positively or negatively to its achievement. The measure of the positive or negative degree of contribution is expressed via the qualitative metrics +, ++, -, --.	$\xrightarrow{\text{--, -, +, ++}}$
Acess relationship	The access relationship models the access of behavioural concepts to resources.	┈┈┈┈┈>
Assignment relationship	The assignment relationship reflects the ownership of a goal, behaviour, resource and/or constraint by an actor.	■────■
Constrain relationship	The constrain relationship expresses the fact that a constraint controls/influence/limits/defines some aspect of the behaviour or a certain resource.	⟶

Figure 1 depicts the GML metamodel, which is the formal abstract syntax of the goal modelling language we are proposing. As one can easily see five constructs together with the relationships between them form the core of the language: goal, actor, resource, constraint and behaviour. Each of the above mentioned constructs can be then further specialised.

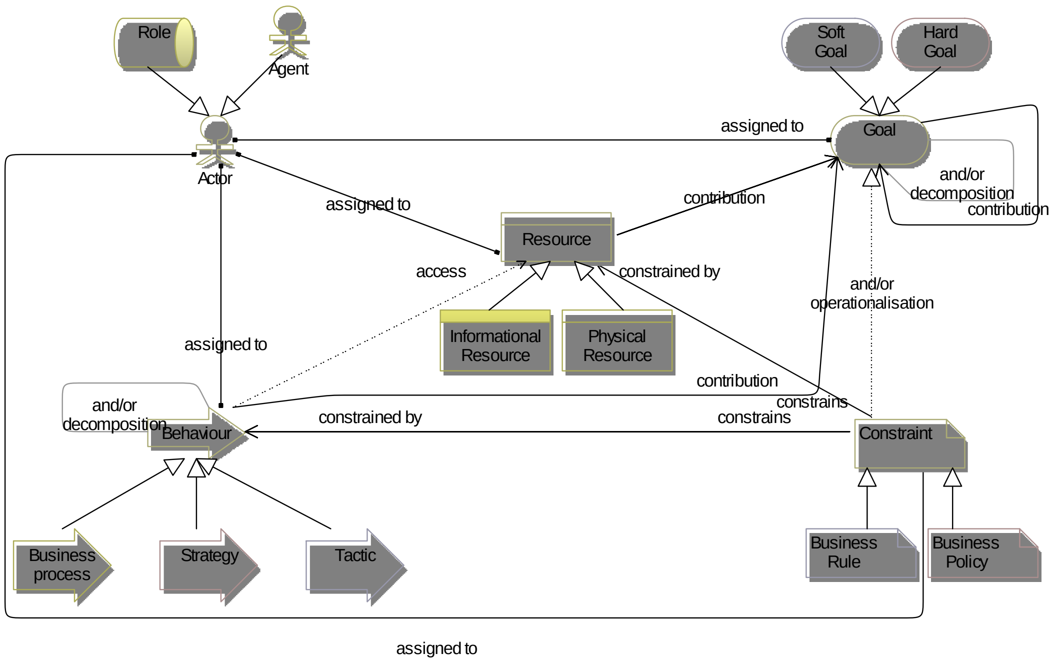


Figure 1. GML metamodel

An example of a GML model is depicted in Figure 2.

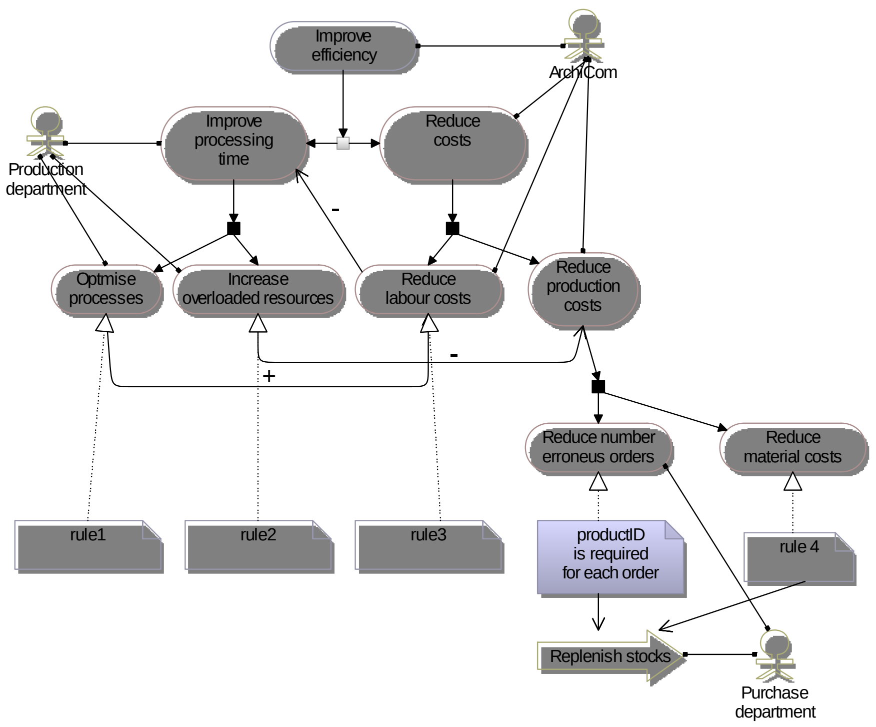


Figure 2. GML model

One important advantage of GML to be stressed is its seamless integration with design languages, in particular with the enterprise architecture language ArchiMate [14]. In order to illustrate this idea we refer to Figure 3 in which an example is given of how goals and their operationalisation as business rules can be incorporated in process and application design expressed in ArchiMate.

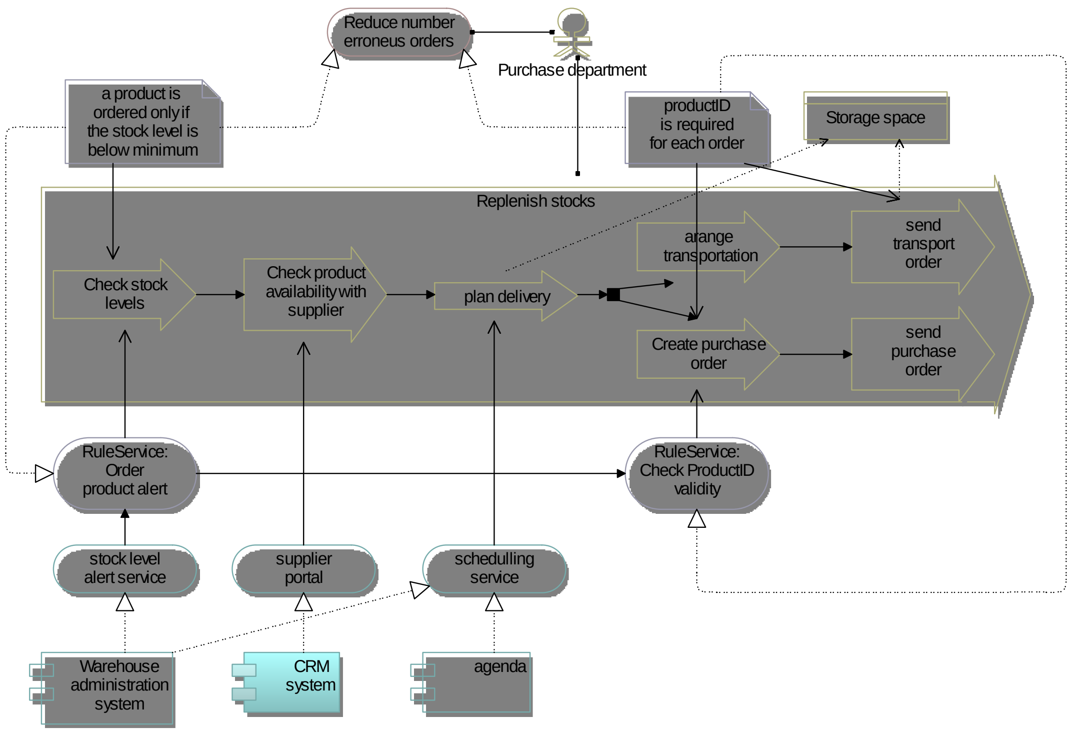

Reduce number erroneus orders
Purchase department
a product is ordered only if the stock level is below minimum
productID is required for each order
Storage space
Replenish stocks
arange transportation
send transport order
Check stock levels
Check product availability with supplier
plan delivery
Create purchase order
send purchase order
RuleService: Order product alert
RuleService: Check ProductID validity
stock level alert service
supplier portal
schedulling service
Warehouse administration system
CRM system
agenda

Figure 3. Goal and architecture modelling

In this particular case, one can see that business rules can be realized by rule services that are subsequently used by business processes. In turn such rule services may use other application services.

Goals and Service-Oriented Software Design

In order to react swiftly and coherently to changes, the agile SOA architecture must provide a capability to capture how services realise business motivations (vision, goals, objectives, missions, strategies, tactics, policies, regulations, etc.). One way to incorporate the business view in SOA design is to express this view formally in terms of goals, refine them, translate them into business rules (BR) and, then, integrate them in the design and composition of services. Using business rules as vehicle to achieve this has the advantage of allowing the decoupling of the business logic (expressed as goals) from business operations, such as business rules, processes and their supporting applications. Furthermore, the effects of changes of the business logic (e.g., a new business strategy, new laws and regulation or change of the internal business policy etc.) can be thus isolated, affecting the business operations only to a limited and controllable extent (since business goals can be modelled and maintained separately from rules and process models). In this way, it becomes possible for organisations to explicitly manage and maintain goals and business rules, which are no longer hidden and hard-coded in processes and applications [15], and to achieve higher business process and software agility.

In this section we will explain that business rules are also very well positioned to be technically combined with and incorporated in the model-driven design of SOAs, and, thus, to ensure goal satisfaction by the system design. The idea of combining business rules with SOA (in particular in relation to web service technology) has been already around for a while (e.g., [16,17]). Currently, several commercial software platforms (e.g., the Oracle SOA suite, BEA Aqualogic, Web Methods etc.) support the use of business rules for controlling services and the orchestration of services. Thus, combining business rules with SOA is to some extent technically already possible. However the BR specification languages used by these tools are in most cases proprietary and have significant limitations. Furthermore, it should be noted that combining SOA and BRs is only possible at this platform-specific level, which does not yet fulfil the promise of SOA being an architectural style in which software design is driven by and fully aligned with the business needs. Fulfilling this promise would assume that the (partial) specification of both business rules and goals is possible independent of specific implementation platforms in an intuitively understandable manner, accessible to the primary user/creator of these specifications: the non-technical business person. The idea of developing means to specify business rules in nearly natural language is therefore essential. To raise the level of abstraction at which business rules are specified, the availability of a model-driven approach for business rules (as advocated by the MDA paradigm) is a prerequisite.

In this sense, the idea of applying the principles of model-driven design not only to software but also to business rules has recently captured the attention of standardisation bodies such as the OMG and W3C. Work is currently done to finalise standards for BR specification languages in all MDA layers of models (e.g., SBVR [18], RIF [19], PRR [12]). Furthermore, results have been reported with respect to the definition of model transformations between BR specifications languages positioned in the different MDA abstraction layers (e.g., [20]). However, although the two model-driven approaches (for business rules and for software design) follow the same principle, they seem to evolve in parallel and somewhat independently from each other. We argue that they must be combined and complemented with goal specification techniques, which will eventually result into a model-driven approach for SOA in which business rules constitute the expression of goals as reflection of the business logic and through which the decoupling of the business logic from business operations and from applications can be effectively achieved. Thus, in the remainder of this section we propose a framework for the integration of the SOA, model-driven design (MDD), business rules and goals and we outline some open research directions that emerge as a consequence of this integration.

Besides controlling the orchestration of services, one other way to use business rules in the context of SOA is to *provide and invoke them in the form of independent web services*. Thus, rule engines may expose the effect of business rules resulting in decision or derivation (web) services as depicted in Figure 4.

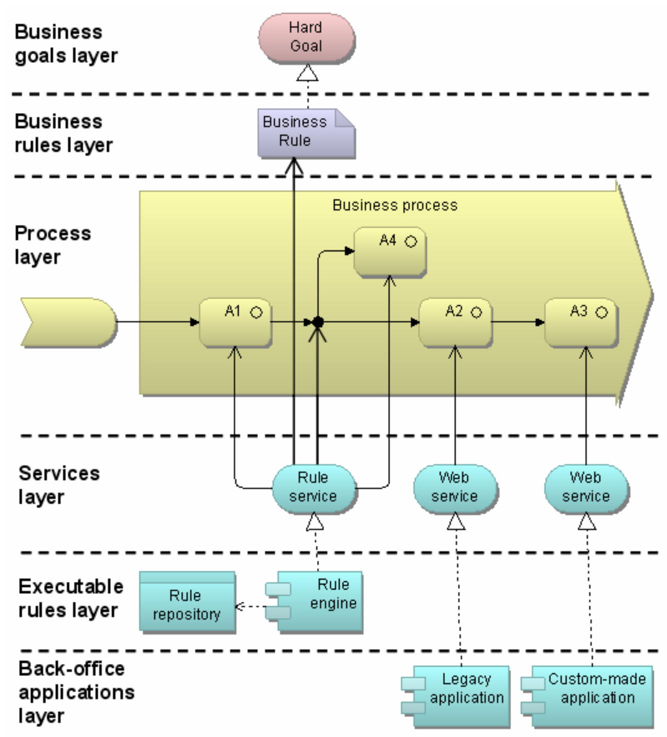


Figure 4. Business rules exposed as a service

Model-driven and goal-based design

The combination of MDA with SOA design is an area that has been extensively researched in the Freeband A-MUSE project (http://a-muse.freeband.nl), which has proposed and validated a design methodology in this sense [21]. As sketched already in the two previous paragraphs, new interesting areas of research emerge from the combination of the two aforementioned paradigms and business rules, which could reuse and extend the Freeband A-MUSE results. As we have shown, business rules can be derived as the operationalisation of an organisation's goals and strategies. As such, rules may not just play a role in capturing business goals but also in incorporating them in the design of application services and in the design and control of the service orchestration. Furthermore, if they resulted as operationalisation of some non-functional goal they could also play a role in specifying and controlling the non-functional properties of the resulting composite service (e.g., performance). Furthermore, we argue that this should be possible throughout the whole stack of MDA models, from high level computation-independent models to platform-specific models. In MDA, model transformations play a central role. Transformations are used to maintain relationships between models at different abstraction levels in the MDA model stack (see the left-column of Figure 5). Typically, one of the languages from OMG's Query-View-Transformation (QVT) standard [22] is used as the language to specify these transformations. The middle column of Figure 5 (which is a "service-oriented" version of MDA) illustrates this. As in the top-down transformations information is added (i.e., the lower-level models are refinements of the higher-level models), it is still unclear to what extent these transformations can be performed fully automatically.

In Figure 5 a distinction has been made between the *design space* (the left column), with models expressed in design languages such as UML, business process modelling languages or architectural description languages, and the *goal&business rule space* (the right column), with goals/rules expressed in special-purpose specification languages (see Figure 5 for examples of such languages). The integration of design models and rule specifications can also be seen as a special type of (horizontal) model transformations – model merging [23].

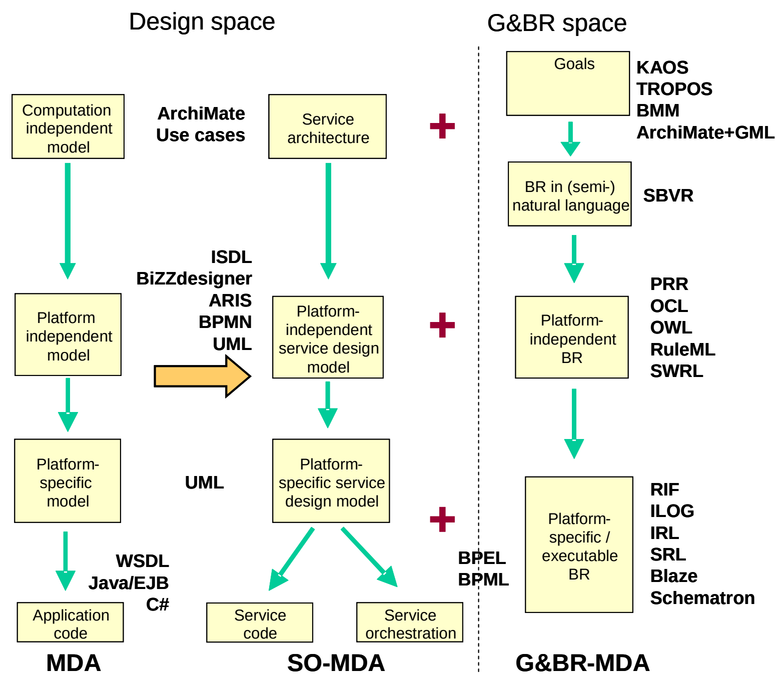


Figure 5. A model-driven view on the integration of service design enhanced with goals/business rules and their specification languages

As the Figure 5 suggests, there is a strong symmetry between the design space and the rule space: for any design model, there may be a corresponding rule set specification. Furthermore, a rule specified at a higher abstraction level may be refined (i.e., transformed) into a rule (set) specification at a lower abstraction level.

In summary, the following types of model transformations are relevant (see Figure 5):

- Vertical model-to-model and model-to-code transformations in the design space as identified in the MDA.
- Horizontal model merging transformations between design models and goal/rule specifications, either at the architectural, platform-independent or platform-specific level.
- Vertical transformations in the goal/rule space going from (and refining) goals into rules expressed in near-natural language down to executable rule specifications.

In this section we have presented our vision on what is needed in order to be able to design service-oriented applications in a model-driven and goal-based way.

The framework we defined also led us to a number of areas having significant research potential. In particular, we believe more work is needed, on one hand, in the area of integration between design languages and BR specification languages in all layers of the MDA stack and, on the other hand, between goal specification languages and rule languages. Furthermore, we argue that business rules could play a significant role in the operationalisation of non-functional goals and in the specification and analysis of non-functional properties of services. Also, although current MDA, SOA, BRMS and other modelling tools can be used to partly support the framework presented in this section, several integration issues and gaps can be identified:

- Integration between goal modelling and business rule specification languages and tools is still missing.
- Integration between design and BR specifications has been only partly realised, and only at the platform-specific level (in software platforms, such as, Oracle SOA suite or IBM's Websphere, etc.).

Finally, one important issue to be addressed is the development of more methodological and tool support for the specification of model transformations between the different layers of models in the MDA stack.

The HNP case at the Medical Spectrum Twente

In order to validate the proposed goal modelling approach we carried out a case study in a health care setting. Goal of this case is twofold: first to validate the GML language in a real case situation and second to illustrate the framework presented in the previous section.

Case presentation

For this case study, we selected a mid-size 'top-clinical' hospital located in the east of the Netherlands. This hospital supplies approximately 820 beds for the region, divided over five locations. We focused on the diagnosis and treatment process of Hernia nuclei pulposi (HNP) or "spinal disk herniation". In this case process, several specialists work together, in order to fulfill the complete patient treatment. Coordinating and collaborating in this process is a major challenge, considering the different measures for optimal performance (e.g. patient satisfaction, costs, treatment quality, and through put time). The selected patient treatment process, which we observed via interviews, participation, and group discussions, over a period of 3 months, is based on a standardized clinical trial. A clinical trial is a instrument used to organize the multi-disciplinary care process for the patient, in order to manage and improve the quality of the treatment process.

Of all patients diagnosed with HNP, only 5 to 6 percent receive medical treatment. Yearly 600 patients are treated in this hospital. The average treatment time (or throughput time) varies between four to twelve months before full recovery, depending on the medical background, and the physical state of the patient. The actual time spend by the patient within the hospital is limited to approximately three fulltime days. As already stated, the HNP treatment process in this hospital department is standardized according to a uniform clinical trial. This standard trial is multi-disciplinary care plan including all key interventions (i.e., interventions that are applied to 80% of all patients) and targets. In the HNP treatment, disciplines involved are, next to general practitioners, physiotherapists, anesthetists, neurosurgeons, radiologist, lab-assistants, administrative staff, and nursery staff.

For the application of our goal analysis method we zoom in on a crossing of two processes of two involved disciplines in the HNP treatment; the neurosurgery department, and radiology department. Both these departments can be considered value shops, with a typical value shop process (problem finding and acquisition, problem solving, choice, execution, and control / evaluation).

The main department involved with this HNP procedure is the neurosurgery department. This department is responsible for initiating the clinical trial and the administrative settlement. In this case study we focus on the operations of both the neurosurgery department and the radiology department. The issue in this case study is the automation and improvement concerning the ordering, requesting, and planning of the medical diagnostics as provided here by the radiology department, for a treating specialism (here the neurosurgery department).

The current flow of the basic process is initiated by either a general practitioner, or a specialist in a hospital, diagnosing a patient HNP. After this diagnosis, a complex treating process is started to heal the patient from his injury. At a certain point in the process the neurosurgeon forwards the patient to the radiologist for a scan of the internals of the patient. This operation can comprise several scans, ranging from a simple 'bucky' scan (Röntgen scan) to the more complex CT scans or MRI scans (more detail provided at the description of the radiology process). Here the patient is send to the radiology secretary to make an appointment for the advised scan. The response for this request for a scan can be given directly (as an scheduled appointment), or with an acknowledgment for the request (upon which the scheduled appointment is communicated later with the patient). Thus, in this process there is a clear distinction between requesting and scheduling. On the day of the scan, the patient registers at the central radiology desk, and is scanned after a short period of time. The radiologist checks the quality of the scan, and the scan is being processed into the system.

After the radiologist concludes with a final diagnosis, the results are send to the requesting department (i.e. the neurosurgery department). Now the neurosurgeon can finalize his diagnosis, and decide whether to perform medical surgery. After the surgery, the patient is put up for post surgery treatment, and outpatient recovery. At this point the patient should be fully recovered.

For case analysis we will focus on the interaction between the neurosurgery department, and the radiology department. Problems that are encountered in this current process flow are:

- Sub-optimal usage of the radiology resources
- Long waiting lists for radiology scans
- Difficulties for requesting and scheduling patient treatment
- Unknown costs for the neurosurgery per treatment per patient
- No transparency and flexibility in the radiology planning process

Both the neurosurgery process as well as the radiology process are described in more detail below.

The case description will be provided by means of a process model of both the neurosurgery treatment process of HNP, and the process model of the medical diagnostics as provided by the radiology department.

The process overview of the clinical trial can be modeled as in Figure 6. The modelling language we used for this formal specification is The Open Group's standard ArchiMate [14] (as implemented in the BiZZdesign Architect tool [24]). In this overview five phases are involved, ranging from the diagnosis to the final recovery. Only the medical surgery and the post-surgery treatment are fully conducted within the hospital (clinical). The actors involved in each process part are linked to each step.

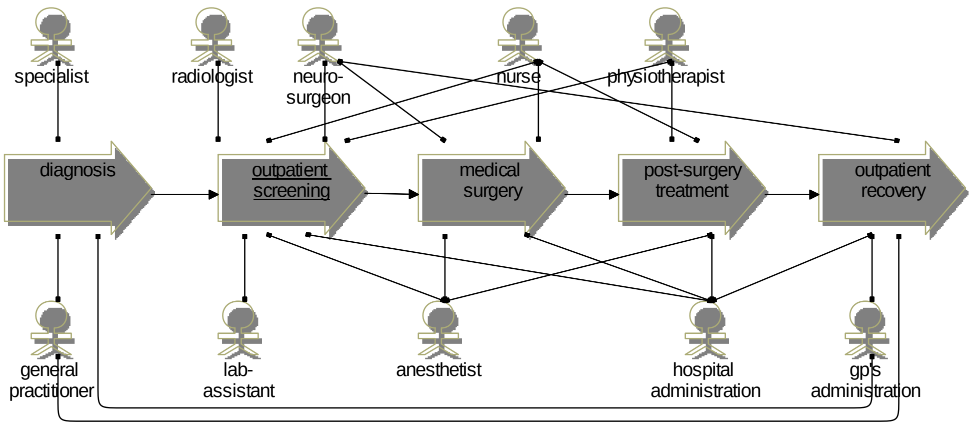


Figure 6. Overview process model HNP treatment

When zooming in on the outpatient screening (second module in the process overview), we get the overview depicted in Figure 7. In this part of the treatment 10 major process steps can be identified, ranging from anamnesis (collecting patient information) to the final preparation and check on the patient file.

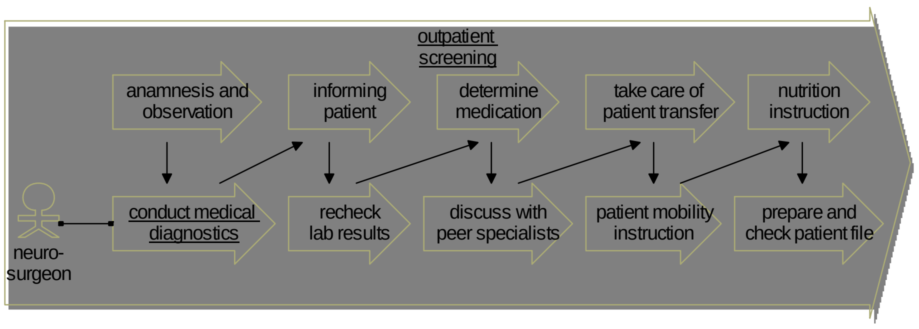

outpatient screening
anamnesis and observation
informing patient
determine medication
take care of patient transfer
nutrition instruction
conduct medical diagnostics
recheck lab results
discuss with peer specialists
patient mobility instruction
prepare and check patient file
neuro-surgeon

Figure 7. Overview process model outpatient screening, HNP treatment

Finally, when zooming in on the 'conduct medical diagnostics' process module, as part of the outpatient screening, the process model can be displayed as in Figure 8. In this process, two diagnostic operations are performed; the lab (e.g. body fluids) testing, and the radiology (body imaging) testing.

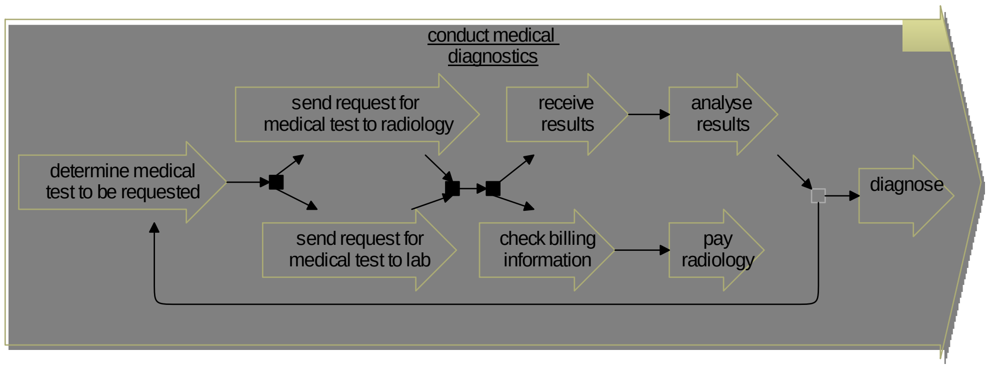

conduct medical diagnostics
determine medical test to be requested
send request for medical test to radiology
send request for medical test to lab
receive results
analyse results
check billing information
pay radiology
diagnose

Figure 8. Medical diagnostics process, part of outpatient screening, part of HNP treatment

In both the lab and radiology testing, basically, the requests for testing are accepted (or denied), the test itself is scheduled, the diagnostics are performed, and the test results are processed. After this step, the results are sent to the requesting department (in this case the neurosurgery).

The second viewpoint in this clinical trial, we zoom in on the radiology department, as part of the HNP treatment process. In this paper we will focus on one location of the hospital (out of 3 radiology departments in the overall hospital group). In total, this hospital group conducts yearly over 200,000 radiology procedures of three types: Radiographies, Magnetic Resonance Imaging (MRI), and Computed Tomography (CT).

The radiology department process is depicted in Figure 9 and it starts from the moment the patient is referred to by another department (in our case study: the neurosurgery department). Next the patient is tested by means of a body scan. Hereafter, the results from the tests are processed and sent to the requesting department. Finally the diagnosis process is finished.

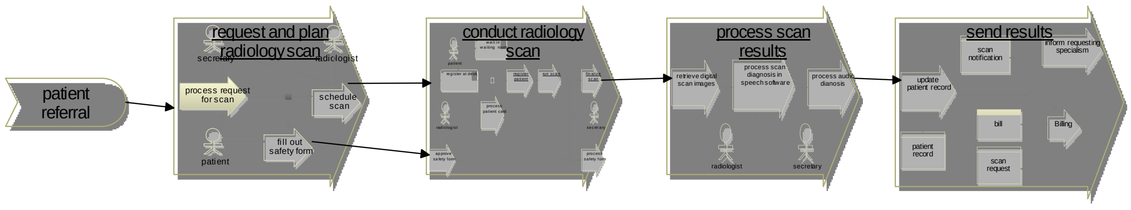


Figure 9. Radiology department process

This whole overview of the radiology process model compromises the *outpatient screening* part of the neurosurgery process model. Due to space limitations we will only zoom in the last sub-process, 'send results', which will be later used again. Al other processes are described in detail in [25]. The actor involved in the sub-process, 'send results' is the secretary.

This process model depicts the general setting of the treatment of a patient, when send from the neurosurgery to the radiology department.

At this intersection of both the neurosurgery treatment of HNP, and the radiology process of body imaging, we applied our goal analysis method. On this intersection, actors from both process chains are involved, with their corresponding goals. As already mentioned, analyzing these goals can be done as well with a bottom-up approach as in a top-down fashion. Both these analyses will be presented in the following paragraph.

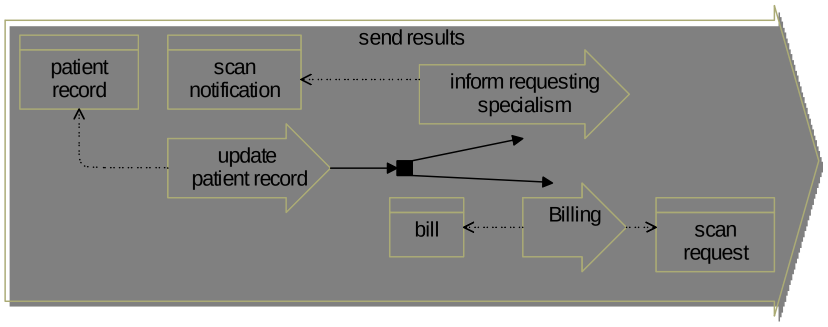


Figure 10. Send results process

Goal elicitation and system design

When conducting the goal elicitation in a bottom-up approach, first the stakeholders are to be identified. As explained before, stakeholders can be defined as actors who have an interest in the issue under consideration, who are affected by the issue, or who - because of their position - have or could have an active or passive influence on the decision making and implementation processes [26]. In the case under consideration, the stakeholders identified are the neurosurgeon, the radiologist, the radiology administration and the patient. The goal analysis is performed via interviews with the relevant stakeholders and group sessions discussing the results from the individual interviews.

Figure 11 displays an overview of bottom-up goal model with the identified stakeholders along with their private and composite (inter-actor and intra-actor) goals.

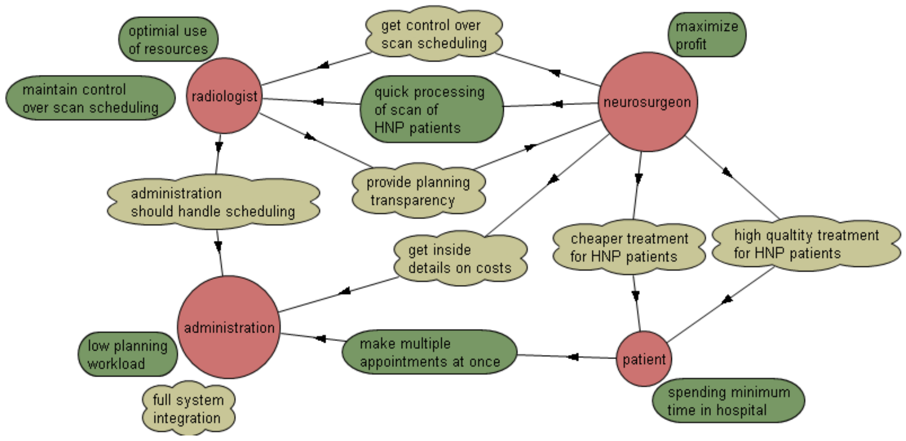

optimial use of resources
get control over scan scheduling
maximize profit
maintain control over scan scheduling
radiologist
quick processing of scan of HNP patients
neurosurgeon
administration should handle scheduling
provide planning transparency
get inside details on costs
cheaper treatment for HNP patients
high qualtity treatment for HNP patients
administration
low planning workload
make multiple appointments at once
patient
full system integration
spending minimum time in hospital

Figure 11. Tropos goal model outpatient screening

This graphical model is constructed with the aid of a software tool in which the Tropos methodology is embedded; TAOM4E [27-29]. In this model, the circles represent the identified actors, the rounded rectangles the hard goals, and the cloud shapes represent the soft goals. When a goal is situated between two actors A and B, this means actor A depends on actor B for attaining this goal. The same model has been expressed using the GML language and can be found in Figure 12.

For the sake of simplicity in the GML model we only represented the assignment of goal to actors and we chose to omit the dependencies relationships from the Tropos model. Nevertheless these could be modeled as well in GML using the pattern shown at the bottom of Figure 13.

Since a dependency in Tropos is a link between actors, indicating that one actor depends on another to attain some goal, in GML the same thing is expressed even more precisely by indicating which behaviour (in the case of the given example "schedule scan") of the *dependee* (the radiologist) contributes to the achievement of the goal ("get control over scan scheduling") of the *depender* (the neurosurgeon) for which the actors the dependency relation was established.

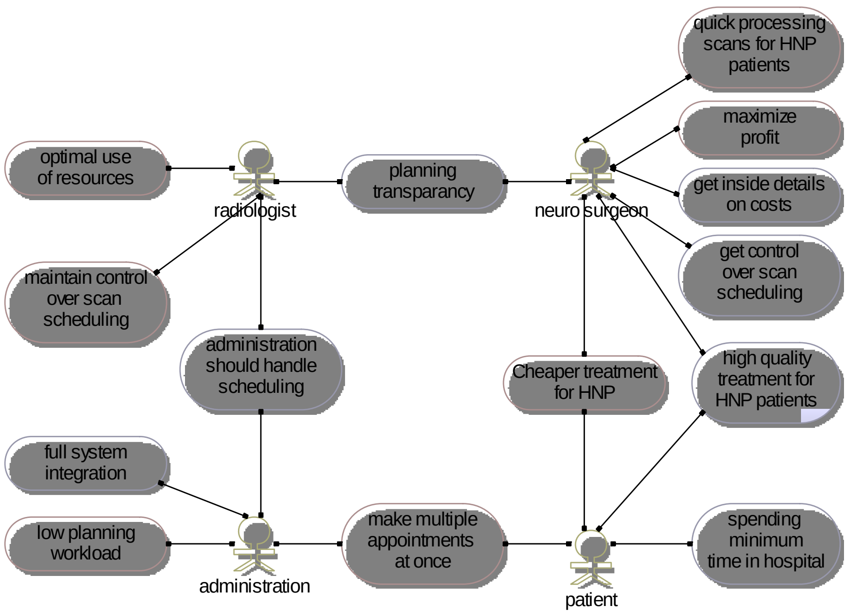

quick processing scans for HNP patients
maximize profit
get inside details on costs
get control over scan scheduling
high quality treatment for HNP patients
spending minimum time in hospital
optimal use of resources
radiologist
planning transparancy
neuro surgeon
maintain control over scan scheduling
administration should handle scheduling
Cheaper treatment for HNP
full system integration
low planning workload
administration
make multiple appointments at once
patient

Figure 12. GML goal model outpatient screening

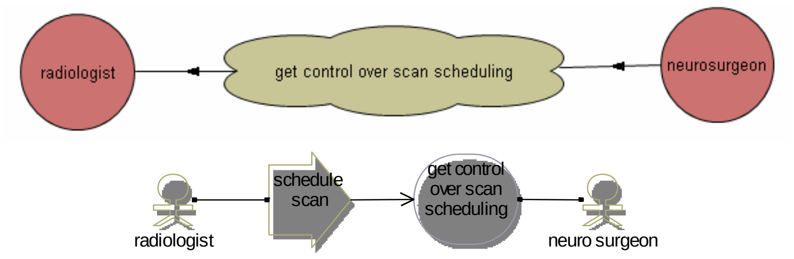

radiologist
get control over scan scheduling
neurosurgeon
schedule scan
get control over scan scheduling
radiologist
neuro surgeon

Figure 13. Tropos dependency modelled in GML

In the HNP case study, several goals can be identified. The efficiency of the HNP patient scan planning and scheduling at the radiology department is critical for the performance of the radiology department. The goal of the radiologists is to maximize the utilization of the resources (like MRI scanners), while guaranteeing best quality to their patients. At the same time the neurosurgeons want their patients diagnosed as fast as possible. For this they want insights and editing rights in the radiology schedule. Next to this, it is important for the neurosurgery department to have clear view on the costs attached to treating patients, and service their patients with the highest level of quality at low costs (thus maximize its profits). The administration prefers a fully integrated system (the radiology system, the neurosurgery system, and the hospital information system are not fully integrated), and a work load evenly spread over the whole day. The goal of the patient is to make multiple appointment (e.g. lab tests, radiology scans) at the same time, and have them succeeding.

In order to illustrate our framework we choose to focus on the neurosurgeon, and in particular, on its goal "maximize profit". The first step in applying our method is to decompose and refine this goal. At the CIM level a goal tree model is proposed (see Figure 15), using the *and/or decomposition* relation of the GML language. This model also shows the other goals of this actor and how the sub-goals derived from "maximize profit" may contribute to other goals as well. Furthermore, some leafs in this goal tree are operationalised by means of business rules. Each of these business rules constrain some behaviour (in this case some business processes assigned either with the neurosurgeon ("check billing information" and "send request for medical test to radiology") or with the radiology ("billing").

At this level, business rules are specified in (near-) natural language. Note that the chosen example contains both rules for controlling the process flow (e.g., Biasing algorithm) and rules that can be implemented and used as independent rule services (e.g., the calculation of tariffs based on processing time).

At the PIM level we focus on how the tariff calculation business rule constrains the "send results" process and show how this part of the goal tree can be refined and transformed into a behaviour model expressed in the ArchiMate modelling language which has been extended with the GML constructs (see Figure 14).

Finally, at the PSM level, the previous process model can be transformed into the BPEL specification depicted in Figure 16 and realized using the Oracle SOA suite that integrates among others a BPEL engine with a BR authoring tool, engine and repository. Please note that the in the "send results" process the calculation of the amount that has to be billed has been externalised by invoking a so-called decision service (e.g., "DecisionServiceCalculateAmountBilledPL"). This decision service wraps the business rules stored in a dictionary in a rule repository.

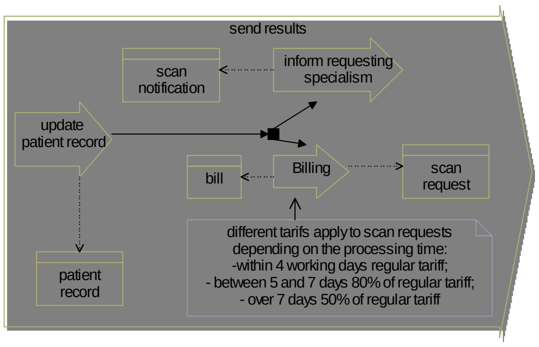

send results
scan notification
inform requesting specialism
update patient record
bill
Billing
scan request
patient record
different tarifs apply to scan requests depending on the processing time: -within 4 working days regular tariff; - between 5 and 7 days 80% of regular tariff; - over 7 days 50% of regular tariff

Figure 14. The "send results" process annotated with the tariff calculation business rule

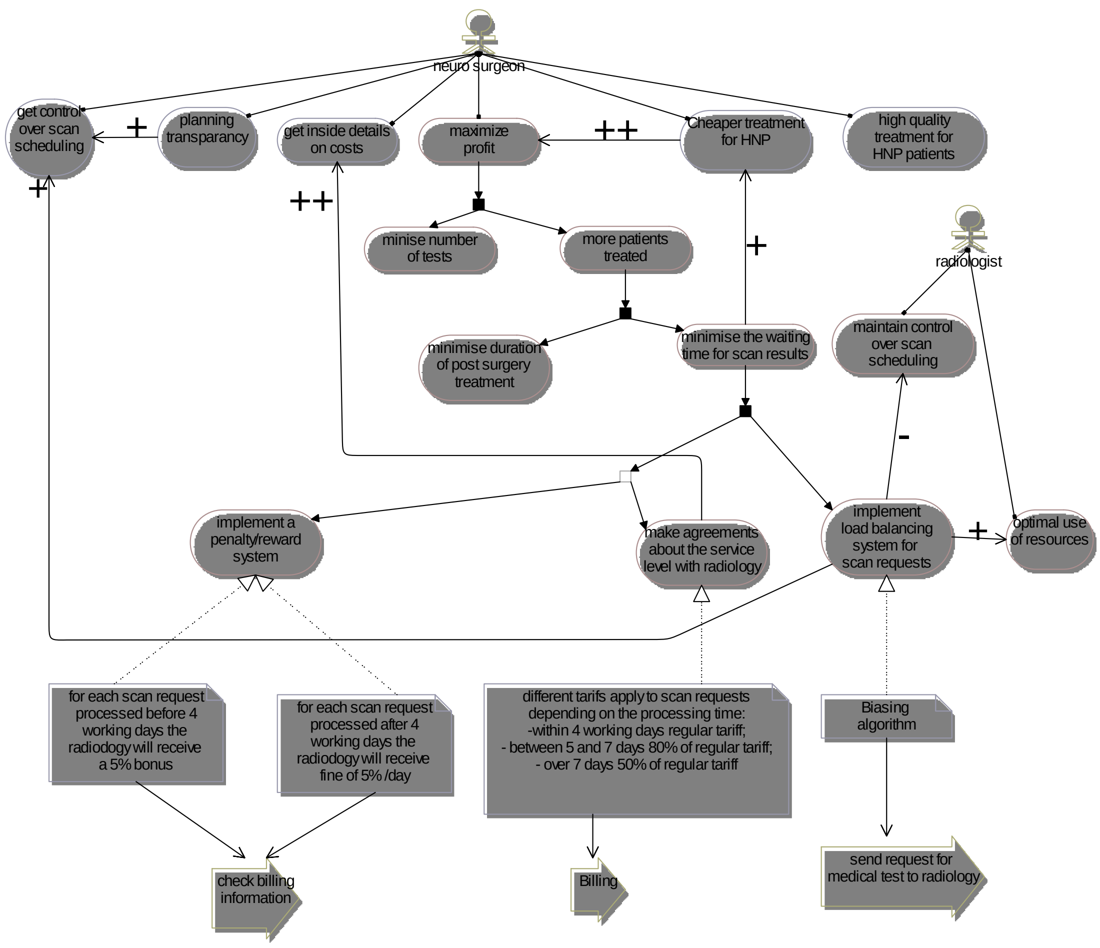

neuro surgeon
get control over scan scheduling
+
planning transparancy
get inside details on costs
maximize profit
++
Cheaper treatment for HNP
high quality treatment for HNP patients
+
++
minise number of tests
more patients treated
+
radiologist
minimise duration of post surgery treatment
minimise the waiting time for scan results
maintain control over scan scheduling
-
implement a penalty/reward system
make agreements about the service level with radiology
implement load balancing system for scan requests
+
optimal use of resources
for each scan request processed before 4 working days the radiodogy will receive a 5% bonus
for each scan request processed after 4 working days the radiodogy will receive fine of 5% /day
different tarifs apply to scan requests depending on the processing time: -within 4 working days regular tariff; - between 5 and 7 days 80% of regular tariff; - over 7 days 50% of regular tariff
Biasing algorithm
check billing information
Billing
send request for medical test to radiology

Figure 15. Neurosurgeon goal tree

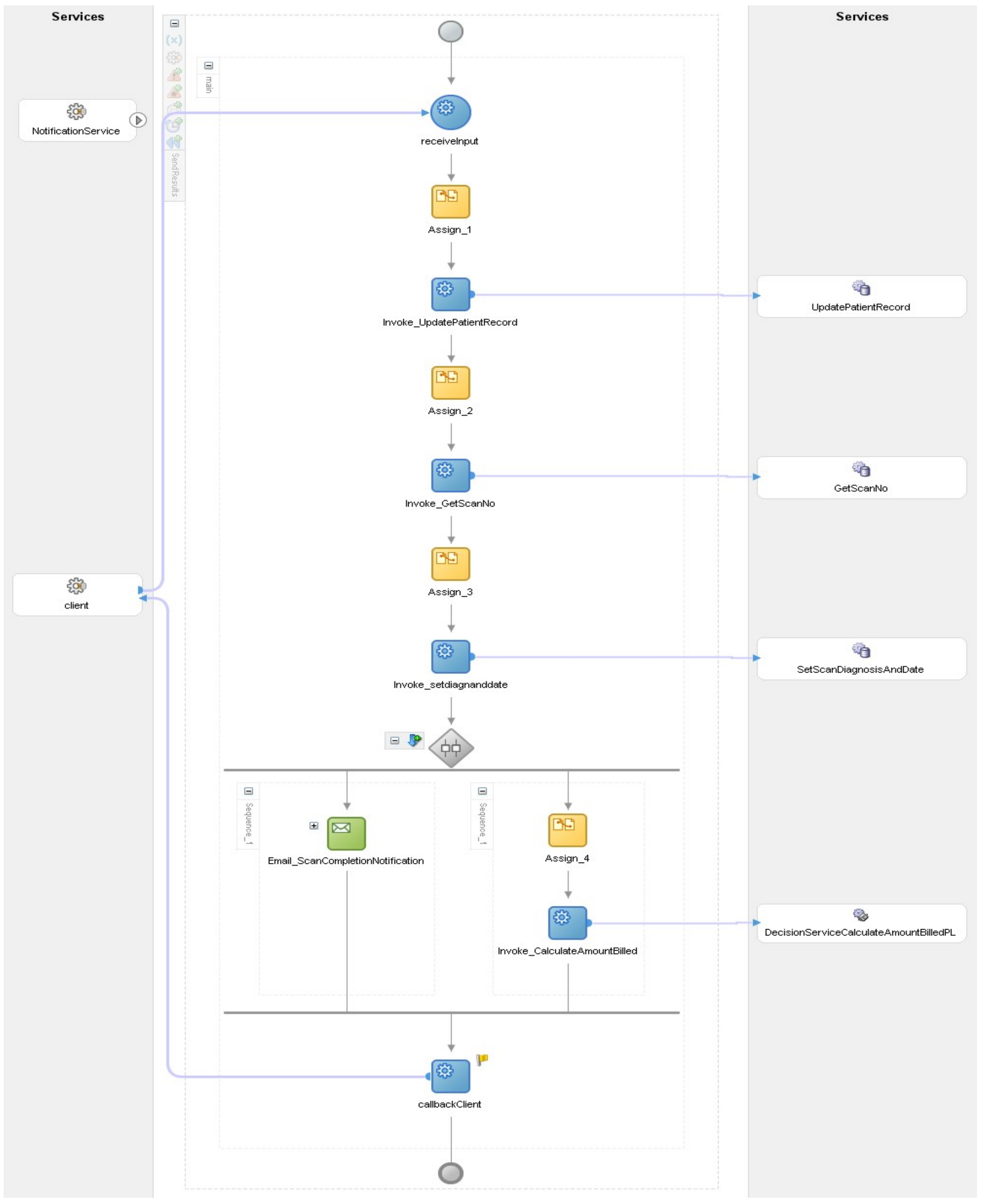


Figure 16. BPEL process

Conclusions

In this paper we have proposed a goal modelling approach and defined the abstract syntax and the concrete syntax of a modelling language (GML) for the specification of business goals. The conceptual foundation of the GML language is the result of an analysis of the metamodels of three representative languages that have been proposed in the literature: KAOS, Tropos and BMM. This means that GML is aligned with well-known existing goal modelling languages and methods. Furthermore, the GML syntax is easy to integrate with design languages such as business process modelling languages (e.g., BiZZdesigner [30]), architecture languages (e.g., ArchiMate [14]) and business rule specification languages (e.g., SBVR [12]). Simplicity was also an important principle that we have followed in the design of GML. Subsequently, GML is suitable for non-technical users since it is intuitive and it has a limited number of modelling concepts and relationships. They are, nevertheless, sufficient to ensure that GML has the enough expressive power.

Next to GML, we have proposed a model-driven framework for the goal-driven design of service-oriented software applications showing how GML specifications could be used during software design. We also provided an example to illustrate this framework and to demonstrate the role goals operationalised by means of business rules can play in the context of MDD of SOAs.

This research also pointed out a number of areas having significant research potential. In particular, we believe more work is needed in the following areas:

- Development methodological support for the elicitation and refinement of goals. In this sense we believe that the development of a goal-ontology and of a catalogue of goal refinement patterns may play an important role.
- Development goal analysis techniques, in particular in the area of non-functional goals but also in relation with reasoning techniques concerning goal achievement.
- Integration between design languages and BR specification languages in all layers of the MDA stack. Furthermore, we argue that the operationalisation of goals by means of business rules could play a significant role in the specification and analysis of non-functional properties of services and therefore should be further investigated.
- Implementing modelling tool support for GML, possibly by extending ArchiMate [14]/Architect [24] with support for the GML constructs.

- Extending ArchiMate/Architect with support for the specification of rules in nearly natural language (e.g., in SVBR)
- Defining model transformations between ArchiMate+GML and platform specific languages (e.g., BPEL combined with a rule specification language).
- Investigating the relation between our approach for goal modelling and Multi-Agent-System design, which are very well positioned to handle the negotiation of actor coordination terms and decentralized and dynamic scheduling coordination problems (that can not be solved optimally with classical operations research techniques).

References

1. Anton AI, McCracken WM, Potts C. Goal decomposition and scenario analysis in business process reengineering, Proceedings of the 6th international conference on Advanced information systems engineering table of contents, 1994, pp. 94-104.
2. Anton AI. Goal-based requirements analysis, Second IEEE International Conference on Requirements Engineering (ICRE `96) , Colorado Springs, Colorado, pp. 136-144, 15-18 April 1996.
3. Keller SE, Kahn LG, Panara RB. Specifying Software Quality Requirements with Metrics. System and Software Requirements Engineering 1990, p. 145-163.
4. Kueng P, Kawalek P. Goal-based business process models: creation and evaluation. Business Process Management Journal 1997;3(1):17-38.
5. Mylopoulos J, Chung L, Yu E. From object-oriented to goal-oriented requirements analysis, Communications of the Acm 1999;42(1):31-37.
6. Soffer P, Wand Y. On the notion of soft-goals in business process modelling, 2005.
7. Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J. Tropos: An Agent-Oriented Software Development Methodology. Autonomous Agents and Multi-Agent Systems 2004;8(3):203-236.

8. Dardenne A, Lamsweerde A. van, Fickas S. Goal-directed requirements acquisition. The Science of Computer Programming 1993;20(1-2):3-50.
9. Lamsweerde A. van. Goal-Oriented Requirements Engineering: A Guided Tour, In: Proceedings of RE'01 - 5th IEEE International Symposium on Requirements Engineering, Toronto, 2001, IEEE Press, 2001, p. 249-263.
10. Bertolini D, Perini A, Susi A, Mouratidis H. The Tropos Visual Language. A MOF 1.4 compliant metamodel. AgentLlink III AOSE TFG 2, 2005 [cited 2009 April]. Available from: URL: http://www.troposproject.org/papers_files/tropos.pdf.
11. Object Management Group, Business Motivation Model (BMM) Specification dtc/07-08-03, September 2007[cited 2009 April]. Available from: URL: http://www.omg.org/docs/dtc/07-08-03.pdf.
12. Object Management Group, Production Rule Representation: Request for Proposal, br/2003-09-03, Sept. 2003 [cited 2009 April]. Available from: URL: http://www.omg.org/docs/br/03-09-03.pdf.
13. Object Management Group, UML 2.0 OCL Specification, ptc/03-10-14, Oct. 2003 [cited 2009 April]. Available from: URL: http://www.omg.org/docs/ptc/03-10-14.pdf
14. The Open Group, Technical Standard ArchiMate® 1.0 Specification, ISBN: 1-931624-80-1, Document Number: C091, Published by The Open Group, February 2009.
15. Hermans L, Lemahieu W, Vanthienen J. Real agility and transparency requires a combination of BPM/SOA, EDA and BRA, In Proceedings of the 6th European Business Rules Conference, Düssseldorf (Germany), Jun. 18-19, 2007.
16. Geminiuc K. A Services-Oriented Approach to Business Rules Development, SOA Best Practices: The BPEL Cookbook (Oracle white paper), retrieved on April, 6-th, 2008. Available from: URL: http://www.oracle.com/technology/pub/articles/bpel_cookbook/geminiuc.html.
17. Rosenberg F, Dustdar S. Business Rules Integration in BPEL – A Service-Oriented Approach, in Proc. 7th IEEE International Conference on E-Commerce Technology (CEC'05), Munich, Germany, July 2005.
18. Object Management Group, Semantics of Business Vocabulary and Business Rules Specification, OMG Adopted Specification, 2006.
19. RIF Working group. [online series] [cited 2009 April]. Available from: URL: http://www.w3.org/2005/rules/wiki/RIF_Working_Group.
20. Linehan MH. Semantics in Model-Driven Business Design, IBM T.J. Watson Research Center, New York, 2006.
21. Almeida JPA, Iacob ME, Jonkers H, Quartel D. Model-Driven Development of Context-Aware Services, In: Frank Eliassen, Alberto Montresor (Eds.) Distributed Applications and Interoperable Systems: 6th IFIP WG 6.1 International Conference, DAIS 2006 Lecture Notes in Computer Science, Volume 4025, pp.213-227, 2006, ISSN: 0302-9743, Springer-Verlag.
22. Object Management Group, Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, Final Adopted Specification ptc/05-11-01. [cited 2009 April]. Available from: URL: http://www.omg.org/docs/ptc/05-11-01.pdf.
23. Kolovos DS, Paige RF, Polack FAC. Merging Models with the Epsilon Merging Language (EML). Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2006;4199:215-229.
24. BiZZdesign B.V., Tool Manual Architect, Enschede, October 2006.
25. Iacob ME, Rothengatter D, Hillegersberg J. van. Goal specification and analysis in value-shop systems, Technical report, Telematica Instituut, January 2009.
26. Varvasovszky Z, Brugha R. How to do (or not to do)… A stakeholder analysis. Health Policy and Planning 2000;15(3):338-345.
27. Morandini M, Penserini L, Perini A. Automated Mapping from Goal Models to Self-Adaptive Systems. ASE 2008:485-486.

28. Penserini L, Perini A, Susi A, Mylopoulos J. High variability design for software agents: Extending Tropos. ACM Transactions on Autonomous and Adaptive Systems 2007;2(4):article no 6.
29. Perini A, Susi A. Developing Tools for Agent-Oriented Visual Modeling. MATES 2004: 169-182.
30. Berg H. van den, Franken H, Jonkers H. Handbook Business Process Engineering, BiZZdesign Academy, 2008.