Nòmos Approach Applied to a Brazilian Regulatory Agency

Cristiane Lima Guadagnin Cardoso,

crislgcardoso@gmail.com

**Abstract.**

***Nowadays organizations are facing a lot of challenges, based mainly in constant changes in the environment where they are inserted. One of biggest drivers of these changes is Government, the way that it manages resources to deliver public services, affects directly organizations - public, private and social. The model Brazil adopts since 1995 is based on regulatory agencies1. This model brings out the necessity of defining suitable methods and techniques to support the design of law-compliant systems. This paper presents a theory review of a Nomos methodology a technique to deal with law-compliance system, brings out initial concepts that support agent-oriented methods.***

**Keywords:** requirement engineering, agent-oriented, Nòmos, Tropos, law-compliance

1. Introduction

Nowadays organizations are facing a lot of challenges, based mainly in constant changes in the environment where they are inserted. One of biggest drivers of these changes is Government, the way that it manages resources to deliver public services, affects directly the organizations - public, private, social. The model Brazil adopts since 1995 is based on regulatory agencies1. A way to evaluate how these rules affect the organization like the one of the statement below may be found in Nòmos[[1]](#footnote-1) approach.

*"The regulatory agencies were created to oversee the provision of public services by private initiative. In addition to controlling the quality in the provision of service, lay down rules for the sector. Currently, there are ten regulatory agencies, deployed between December 1996 and September 2001, but not all carry out supervisory activities."*[[2]](#footnote-2)

The structure of this paper comprise, besides this introduction, a review of Tropos and Nòmos, most recent and relevant literature, presented in section 2, a small but real example of a Brazilian transports services regulatory agency in section 3 and closing this work we present a discussion the strengths of this approach used in the real domain and possibilities in this line of work.

1. Literature Review

Nòmos approach can be classified as an agent-oriented software development method, since it is an extension of Tropos approach, which is agent-oriented. Based on this fact, we decide to present initially a brief explanation about Tropos, its language and method, and them, about Nòmos. In this paper we are interested only with the requirements engineering part of Tropos method, because the law compliance extension deals with this activity.

* 1. Tropos

An agent-orient software system is often required to have properties such as autonomy, social ability, reactivity, and proactivity. Other features, which are sometimes required are mobility, veracity, rationality, and the like [13]. A key idea about software systems possess these properties is that it is conceived and programmed at a knowledge level [8]. Thus, in Agent Oriented Programming (AOP), we talk of *mental states* and beliefs instead of machine states, of *plans and actions* instead of procedures and methods, of communication, negotiation and *social ability* instead of interaction and I/O functionalities, of goals, desires, and so on. Explicit representations of such mental notions provide, at least in part, the software with the extra flexibility needed in order to deal with the intrinsic complexity of applications such as those mentioned earlier [1]. On this explanation there is a statement that we would like to explore for a better understanding: *is conceived and programmed at a knowledge level.*

* + 1. Conceived and Programmed at the Knowledge Level

Conceived and programmed reflect the idea of process, in software development process we have four phases that are well-established established in the Software Engineering literature and are supported by various methodologies and tools.: (1) Requirements, (2) Architectural Design, (3) Detailed Design and (4) Implementation.

So the *first* main point is related to the process describe above, where the initial phase - *requirements* is split in two new phases: early and late requirements. *i*\* framework introduced this approach, which is adopted by Tropos methodology, although not widely practiced by the software development community. The idea behind knowledge level is *mental concepts*, which is the *second* key point, because they reflect agents’ mentalist notions, their beliefs, desires and intentions (BDI).

Putting these two idea ideas together, *mental concepts* and *early/late requirements*, give us the ability to capture *metal state*s, as point by Bresciani et al in [1]: *"we talk of mental states and beliefs instead of machine states. The main motivation underlying this earlier work was to develop a richer conceptual framework for modeling processes which involve multiple participants (both humans and software systems). The rationale of i\*model is that by doing an earlier analysis, one can capture not only the what or the how, but also the why a piece of software is developed. This, in turn, supports a more refined analysis of system dependencies and encourages a uniform treatment of the system’s functional and non-functional requirements."*

*"While developing agent oriented specifications and programs, one uses the same notions and abstractions used to describe the behavior of human or social agents, and the processes involving them. The conceptual gap from what the system must do and why, and what the users interacting with it must do and why, is reduced to a minimum, thus providing (part of) the extra flexibility needed to cope with application complexities."*

* + 1. Tropos meta-model - relashonships

Modeling is a way to represent an idea, then to represent the *mental state*s, is necessary to define specific concepts and their relationship. To represent them a language is necessary. *i*\*/Tropos language is used for this purpose.



Figure – Tropos language meta-model defining actor and goals representations, as well as, their structure and dependency association [11].

Tropos language key concepts are: *Actor*, *Goal*, *Plan*, *Resource*, *Dependency*, *Capability*, and *Belief*, as presented in Figure 1[[3]](#footnote-3). Below, there is a brief description of these concepts; a detailed description can be find in [11].

* ***Actor*:** is an entity (human or software) that has strategic goals and intentionality within the system or the organizational setting;
* **Goal:** represents actors’ strategic interests, where there two kinds: *hard goal and soft goals*. This last one is usually (but no always) used to define non-functional goals;
* ***Plan*:** represent a way of doing something to achieve a goal, hard or soft;
* ***Resource*:** represents a physical or an informational entity;
* ***Dependency*:** represent the relation between two actors to achieve a goal. This implicates in the definition of three other concepts; the *depender*, which is the "owner" the original goal, the one who needs to achieve the goal, passing the responsibility to a *dependee*, which is the "hands on", the one who will do something to achieve the goal. The *dependum*, is the reification of the dependency relationship;
* ***Capability*:** represents the ability of an actor to fulfillment of a goal;
* ***Belief*:** represents actor knowledge of the world.



Figure 2 - Tropos language meta-model for representing goal decomposition, as well as, supporting elements to contribution and means-end analysis modeling [11].

Complementing the semantics of actor dependencies, actors and goals taxonomy and other elements, such as plans and resources, Tropos language also offers expressive relationships. They give support decomposition, contributions and means-end analyses. These features are illustrated in Figure 2. Goals may be decomposed in “AND” relationships. When we decompose a goal, the downward path tells *what* states of the world do stakeholders “want”. Going upward, means *why* stakeholders want that world states. Goals and decompositions entail that for its satisfaction, all of its decompositions must also hold satisfaction. “OR” decompositions are also admissible, meaning that if any of the decompositions hold, the decomposed goal is satisfied. Goals may contribute to other goals satisfaction, as well as, plans and resources. Furthermore a goal, plan or resource may be a means to satisfy a goal. All these analysis are possible only inside an actor point of view, represented in the meta-model by the relationship of the same name.

* + 1. Tropos Modeling Method

Independently of modeling phase (early or late) Tropos offers two kinds of diagrams to represent the organization and system goals, namely, *Actor Diagram* and *Goal Diagram*. **Actor modeling** consists of identifying and analyzing both the actors of the environment (early requirements) and the system’s agents. During the early requirement phase the focus is on modeling the application domain, stakeholders and their intentions as social actors which want to achieve goals (system-as is), and during late requirement, the focus is on the definition of the system-to be. Dependency modeling deals with the analysis of actors and agents dependencies to satisfy a goal, achieve a plan or get some resource. It is performed during the actor modeling activity. **Goal modeling** rests on the analysis of actor goals, conducted from the point of view of the actor. There are three basic reasoning techniques to use in this activity: means-end analysis, contribution analysis, and AND/OR decomposition. Means-end analysis focus on "means" (resources, plans, soft goals) to achieve a goal, while contribution analysis identifies goals that can contribute positively or negatively to the target goal that is being analyzed. Goal decomposition shows the relation that refinements are parts of refined goal that need to satisfied in order to satisfy it. Plan modeling is based on the same techniques for goal modeling, means-end analysis, contribution analysis, and AND/OR decomposition, but the focus is on plans.

* + 1. Example - e-Cultural Project - Trentino Government

The concepts and the relationships are represented in the diagram below, which is a fragment of a real application developed for the government of Trentino (Provincia Autonoma di Trento - PAT), eCultural project [1].

The concepts and the relationships are represented in the diagram below, which is a fragment of a real application developed for the government of Trentino (Provincia Autonoma di Trento - PAT), eCultural project , presented in BRESCIANI ET AL. paper



**Figure 1. Actor diagram modeling the stakeholders of the eCultural project.**

****

**Figure 2. Goal diagrams for Citizen and Visitor. Notice the goal and plan decomposition, the means-end analysis and the (positive) softgoal contribution.**

*"The figure 2 shows the actor diagram for the eCulture domain. In particular, Citizen is associated with a single relevant goal: get cultural information, while Visitor has an associated softgoal enjoy visit. Along similar lines, PAT wants to increase internet use while Museum wants to provide cultural services. Finally, the diagram includes one softgoal dependency where Citizen depends on PAT to fulfill the taxes well spent softgoal."* [1]

* 1. Nòmos

Information and communication technologies steadily evolved during the last decades, evidencing the replacement of the concept of software as data processing tool by that of a socio-technical system [2], consisting of software, human and organizational agents and business processes, running on an open network, fulfilling relevant functions for organizations. Such systems got attention of governmental bodies, which are responsible for regulating them through laws, regulations and policies ensuring their compliance with security, privacy, governance and other important concerns to citizens and governments alike.

Regulations impact over Software Engineering is profound, as much as on business practices. As examples, Healthcare organizations estimated $17.6 billion expenditures over a number of years to align their systems and procedures with a single law, the Health Insurance Portability and Accountability Act (HIPAA), introduced in 1996[[4]](#footnote-4). Furthermore, Sarbanes-Oxley Act [3, 12] estimated impact on organizations’ expenditures was $5.8 billion, in 2005 alone.

In order to cope with this context, requirements engineers are faced with the additional challenges in eliciting requirements, i.e., compliant with relevant legal prescriptions. However, unlike stakeholders’ requirements, which can be validated by themselves, requirements introduced for compliance purposes, need to be objectively evaluated for alignment against their originating prescriptions, without the aid of who has produced them. Nòmos is an extension of *i*\*, to build models of legal prescriptions together with intentional elements, and derive requirements that fulfill stakeholder needs and comply with relevant regulations, as well [6].

* + 1. Concepts

The core element of legal prescriptions is normative proposition (NPs). It is the most atomic proposition able to carry normative meaning. NPs contain information concerning the subject, who is addressed by the NP itself, the legal modality and the description of the object of such modality. The legal modality is one of the 8 elementary rights, classified by Hohfeld [4, 5] as: privilege, claim, power, immunity, no-claim, duty, liability, and disability. *Claim* qualifies a person to have something done from another person, who has therefore a *Duty* of doing it. *Privilege* is the entitlement for a person to discretionally perform an action, independent of the will of others who may not claim her to perform that action, and, therefore, have a *No-claim*. *Power* is the (legal) capability to produce changes in the legal system towards another subject, who has the corresponding *Liability*. *Immunity* is the right of being kept untouched from other person performing an action, who has therefore a *Disability*. Complex legal prescriptions are created in law documents by structuring NPs through conditions, exceptions, and other conditional elements. Such elements are captured in Nòmos by introducing priorities between NPs. For example, a data processor may be allowed (i.e., it has a privilege) to process the data of a subject; but the right of the subject to keep her data closed of third parties has a higher priority on the privilege, thus constraining the way data is used by the processor.



Figure - The meta-model of Nòmos language.

* + 1. Meta-model - relashonships

Figure 3 presents an UML [9] class diagram of Nòmos’ meta-model, and shows how it integrates the representation of NPs with the representation of goals. The picture is divided in three areas (legend at the bottom of the diagram); first, *i*\* meta-model, delimited by the dashed line. The elements that form a NP are inside the dotted line compartment. The link between the goal-oriented and the legal part of the meta-model is the class Actor, which is at the same time stakeholder in the domain and subject of NPs [10].

Actors “hold” rights, as well as “want” goals, “perform” tasks and “own” resources; on the other hand, they are addressed (“counterparty”) by rights carried by NPs. Hohfeld legal taxonomy [4, 5] states that rights are related by correlativity relations: for example, if someone has the claim to refunding in a commercial transaction, then some party involved in the transaction will have the duty of providing the funds. Thus, duty and claim are correlatives; the semantics is similar for privilege-noclaim, power-liability and immunity-disability: they describe the same reality from two different points of view. The meta-model represents each pair of types as a single class, e.g., ClaimDuty, each of them sub-class of the abstract class Right. The meta-model also represent priorities between rights by means of the Dominance class, which connects two rights. ActionCharacterization class contains the actual object of the NP; such action is bound to the behavior of actors by means of the Realization class, specifying that a determined goal is wanted by the actor in order to accomplish the action prescribed by law [10].

* + 1. Example - Law 10.233 of June 5, 2001

This is the law sanctioned from Brazil President Cabinet that created the country’s ground and water transportation regulatory agencies, ANTT (ground) and ANTAQ (water). For example,

Chapter IV, *Principles of and Guidelines to the Ground and Water Transportation*, Section II, *General Guidelines*, Article 14, *Except for provisions in specific legislation, the provisions of the Article 13 shall apply according to the following guidelines*, Item III, depends on authorization, Sub item (a), passengers’ ground transportation under charter regimen.

We can rephrase the legal fragment to: *Passengers’ ground transportation under charter regimen depends on authorization*. This means that whichever companies willing to do charter trips to transporting passengers, must be authorized by the agency. Thus, this fragment of the law implies that the companies depend on the agency to authorize them, i.e. there may be a goal “charter travel permits granted”.

The Nòmos solution would be a ClaimDuty relationship (“travel permit”) from the third party transportation service supplier to the agency. Moreover, we should represent the aligned goal of charter travel permits issued and proposed process for authorizing a certain company, realizing that ClaimDuty. Unfortunately, there is no available tool to represent Nòmos visual language. We used a simple UML extension to represent the elements of this small fragment in the goal diagram presented in Figure 5.



Figure 5 - Goal that realize the ClaimDuty.

* + 1. Nomos - a perspective

NOMOS methodology, basic idea, is to define the key concepts related to normative domains (laws, rules regulation, etc.) and them, define the relationship of this concepts with the stakeholders strategic goals.

Figure 3 is very illustrative w.r.t the world states sets in this context. An interesting aspect to capture from the illustration is that, the set of admissible states of the world specified to the system is a subset of the set resulting from the intersection of the stakeholders’ intended states and law imposed ones.



Figure 3 - D = Domain assumptions;

G = Set of states of the world represented by stakeholders' goals; L = set of states of the world described by law prescriptions; S = set of states of the worlds specified to the system[[5]](#footnote-5).

This intersection presents a new perspective that is worth to be analyzed: modeling a stakeholders’ intended states, and relating his state with "another state" in this specific example - LAW, the result, intersection can be seeing as tentative of goal alignment?

As pointed in Alberto Siena [10]: *" Our main point of departure is that have explicit, separate constructs for legal concepts and strategic concepts. This allows to have a representation of law that is close to the textual source, and to link it with goal models of requirements."*

This method, proposed by Nomos of define concepts from a specific domain *(close to the textual source)*, and relate than with strategic concepts (*goal models of requirements),* seems to be a way for doing strategic planning, where this method would be applied to map the impact of a specific domain ( ex. regulation, sustainability, political, economic environment., etc.) on the stakeholders (organization) goals.

1. Conclusion

To better understand Nomos Methodology, a good strategic is to have a previous understand of i\* and Tropos Methodology.

The concepts of Nomos are complex, since they derive from a specific domain, Laws. Then, a previous knowledge on this field facilitates the understanding and therefore its use. However, it is worth noting that this use is questioned since there are no tools to use for represent Nomo's language.

As mentioned, this methodology is used at requirements phase analysis, when the application of it brings out the relation between the desire state of the Law domain and the intentions of the organization/ stakeholders, named - Law - Compliancy .

Another point is that result of this methodology should be evaluated , at the perspective of using this method to do strategic planning.

References

[1] Bresciani, P. et al. 2004. TROPOS: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*. 8, 3 (May. 2004), 203–236.

[2] Bryl, V. et al. 2009. Designing socio-technical systems: from stakeholder goals to social networks. *Requir. Eng.* 14, 1 (2009), 47–70.

[3] Hall, J.A. and Liedtka, S.L. 2007. The Sarbanes-Oxley Act: implications for large-scale IT outsourcing. *Commun. ACM*. 50, 3 (Mar. 2007), 95–100.

[4] Hohfeld, W.N. 1913. Some Fundamental Legal Conceptions as Applied in Judicial Reasoning I. *Yale Law Journal*. 23, (1913), 16–59.

[5] Hohfeld, W.N. 1917. Some Fundamental Legal Conceptions as Applied in Judicial Reasoning II. *Yale Law Journal*. 26, (1917), 710–770.

[6] Ingolfo, S. et al. 2011. Nòmos: from Strategic Dependencies to Obligations. *iStar* (2011), 72–77.

[7] Morandini, M. et al. 2010. On the use of the Goal-Oriented Paradigm for System Design and Law Compliance Reasoning. *iStar* (2010), 71–75.

[8] Newell, A. 1982. The Knowledge Level. *Artif. Intell.* 18, 1 (1982), 87–127.

[9] OMG 2006. UML Specification, Version 2.0.

[10] Siena, A. et al. 2009. A Meta-Model for Modelling Law-Compliant Requirements. *RELAW* (2009), 45–51.

[11] Susi, A. et al. 2005. The Tropos Metamodel and its Use. *Informatica (Slovenia)*. 29, 4 (2005), 401–408.

[12] Sutton, S.G. and Arnold, V. 2005. The Sarbanes&Oxley Act and the changing role of the CIO and IT function. *Int. J. Bus. Inf. Syst.* 1, 1/2 (Jul. 2005), 118–128.

[13] Wooldridge, M. and Ciancarini, P. 2001. Agent-oriented software engineering: the state of the art. *First international workshop, AOSE 2000 on Agent-oriented software engineering* (Secaucus, NJ, USA, 2001), 1–28.

1. From the Greek word Nóµoς, meaning “norm”. [↑](#footnote-ref-1)
2. http://www.brasil.gov.br/sobre/o-brasil/estrutura/agencias-reguladoras. [↑](#footnote-ref-2)
3. *Capability* and *Belief* are Tropos language extended elements, not presented in the basic meta-model. [↑](#footnote-ref-3)
4. Medical privacy - National standards to protect the privacy of personal health information. Office for Civil Rights, US Department of Health and Human Services, 2000. [↑](#footnote-ref-4)
5. This picture was taken from a presentation from Luca Sabatucci, presented in 2010, related to the paper [7]. [↑](#footnote-ref-5)