PhD Research Plan

Model-Driven Service Level Management

Anacleto Cortez e Correia
accorreia@fct.unl.pt

Supervisor: Prof. Doutor Fernando Brito e Abreu

2010

Abstract

Background: IT Service Management (ITSM) is the set of processes that allow planning, organizing, directing and controlling the provisioning of IT services. Among the concerns of ITSM, namely within the service level management process, are the requirements for services availability, performance, accuracy, capacity and security, which are specified in terms of service-level agreements (SLA).

Research problems: SLA definition and monitoring are open issues within the ITSM domain. In this research plan we are going to focus on three specific problems in this context: (1) SLAs in the context of ITSM are informally specified in natural language; (2) SLAs specifications are not grounded on models of ITSM processes; (3) SLAs compliance verification in IT services is not performed at the same level of abstraction as service design.

Research goals: Our main goals on this research work are to mitigate the above problems by proposing a model-based approach to IT services SLA specification and compliance verification. The specification part will be accomplished by proposing a SLA language - a domain specific language (DSL) for defining quality attributes as non functional requirements (NFRs) in the context of ITSM. Its metamodel will be an extension of the meta-model of the adopted process modeling language. As such, it will be possible to ground SLA definition on the corresponding IT service model constructs. SLA monitoring and compliance validation should occur at a level of abstraction that is understood by all the stakeholders involved in the service specification.

Main research questions: (1.1) Can a formal model of SLA improve the efficiency of building SLA contracts? (1.2) Does the SLA language accurately specify the constraints required for an automatic validation of SLA? (2.1) Can the SLA language improve processes representation, to allow monitoring IT services delivery and an automatic validation of SLAs? (3.1) Can the SLA compliance verification, performed at service design level, be more effective than the one performed at the level of service implementation?

Research methodology: An approach based in the scientific method, comprising three main iterative phases, will be performed: (i) the *foundation* phase for the SLA language designing along with the setting up of the SLA specification and validation environment; (ii) the *serviceable* phase where the environment will be instantiated with a real world case, and operational data collected; (iii) the *evaluation* phase where an empirical assessment of research hypotheses, grounded in collected data, will be performed and conclusions drawn.

Categories and Subject Descriptors (ACM): D.2.1 Software Engineering – Requirements / Specifications - Elicitation methods, Languages, Tools.

Keywords: IT Service Management, ITIL; Service Level Management; Service Level Agreements; Metamodels, MDD; Domain Specific Languages, Process Model, BPMN.

Table of Contents

Acronyms list

AD - Activity Diagram
BPDM - Business Process Definition Metamodel
BPMN - Business Process Modeling Notation
BPEL - Business Process Execution Language
CIM - Common Information Model
CIMOM - CIM Object Manager
CMDB – Configuration Management Database
DCML - Data Center Markup Language
DMTF - Distributed Management Task Force
DSL - Domain-Specific Language
DSM - Domain-Specific Modeling
EPC - Event Driven Process Chain
eTOM – Enhanced Telecommunications Operations Management
H_0 - Null hypothesis
H_1 - Alternative hypothesis
IDEF3 - Integrated DEFinition Method 3
IT - Information Technology
ITSM - Information Technology Service Management
ITIL - Information Technology Infrastructure Library
KPI - Key Performance Indicator
M2DM - Meta-Model Driven Measurement
MDA - Model-Driven Architecture
MDD - Model-Driven Development
MDE - Model-Driven Engineering
MOF - Meta-Object Facility, Managed Object Format, Microsoft Operations Framework
MVC - Model-View-Controller
NFR – Non Functional Requirements
OASIS - Organization for the Advancement of Structure Information Standards
OMG - Object Management Group
OCL - Object Constraint Language
QoS - Quality of Service
QVT - Query/Views/Transformations
RAD - Role Activity Diagram
REST - REpresentational State Transfer
RBSLA - Rule-Based Service Level Agreement
SLA - Service Level Agreement
SLM - Service Level Management
SOA - Service-Oriented Architecture
SOAP - Simple Object Access Protocol
UML - Unified Modeling Language
WBEM - Web-Based Enterprise Management
W3C - World Wide Web Consortium
WSDL - Web Services Description Language
WSLA - Web Service Level Agreement
WSML - Web Service Management Layer

WSMO - Web Service Modeling Ontology
WSOL - Web Services Offering Language
XMI – XML Metadata Interchange
XML - eXtensible Markup Language

1. Introduction

Most organizations rely on Information Technology (IT) services to support their business processes [Laudon and Laudon, 2005]. IT services are built upon the technical infrastructure (servers and network devices) as well as on systems and application software. The set of processes that allow planning, organizing, directing and controlling the provisioning of IT services is usually called an IT Service Management (ITSM) framework [Silva and Martins, 2007]. Several ITSM frameworks have been proposed, such as ITIL [ITIL3Sm, 2007], ISO 20000 [ISO20000-1, 2005] [ISO20000-2, 2005] (based on ITIL v.2), MOF [MS, 2008] (a Microsoft proposal, also influenced by ITIL), ISM [Lindquist, et al., 2007], or HP ITSM [Sallé, 2004]). Specially tailored examples of ITSM frameworks have also been proposed, such as eTOM [TMF, 2007], for the telecom industry. Although these frameworks are mainly prescriptive (of best practices) and abstract, efforts are being made to help practitioners to instantiate them, by using process modeling notations such as Business Process Modeling Notation (BPMN) [BPMN, 2009] [Maps, 2010].

As we can see in Figure 1, the interest of academic community for ITSM/ITIL subjects is still low when compared with others domains of computer science. However we can see a trend that shows a growing interest in the latest years by academia in ITSM.

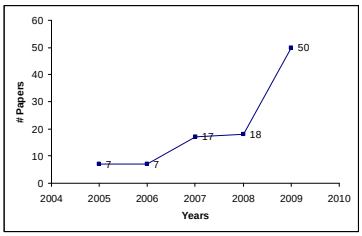

Figure 1 – Papers published in conferences on ITSM/ITIL (Source: www.b-on.pt, Feb. 2010)[1]

In ITIL, the most widely used ITSM framework, some of the processes are: incident management, problem management, change management, service asset and configuration management and service level management [Bon, et al., 2005]. In order to support these processes, IT providers use two kinds of applications as depicted in Figure 2: *service management applications*, which allow to track IT services incidents and problems; and *systems management tools*, for monitoring and control networks, systems or applications [Mayerl, et al., 2005].

Currently, there are many service management ITSM tools available with quite similar functionalities [PinkElephant, 2010], as well as systems management tools [Gläßer, 2005]. Service management applications often overlie and rely, to some extent, on information collected by systems management applications[2]. Systems management tools are outside the scope of this research plan, because we are concerned with an higher level of abstraction (service management).

[1] Query string: "Assunto=(ITIL) Ou Assunto=(ITSM)" em "Eng./Tecnologia"

[2] Systems management tools used to be mainly focused on managing the uptime of network equipment. Nowadays, albeit network equipment availability is still important, systems management tools are mainly focused on managing the applications' performance. So, managing the availability of network equipment or any other component of the IT infrastructure is seen in the broader context of managing application performance.

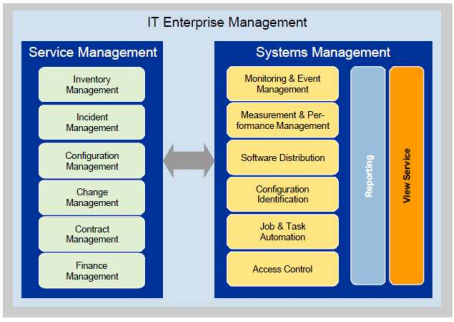

Figure 2 – IT Enterprise Management Applications (adapted from [Gläßer, 2005])

Among the concerns of ITSM, namely within the service level management process, are the requirements for services availability, performance, accuracy, capacity and security, which are specified in terms of service-level agreements (SLA) [Sallé, 2004]. In SLAs, such requirements, built upon quality attributes, are quantitatively bounded (e.g. maximum time to recover or repair, maximum latency or response time, minimal percentage of availability). However, although those requirements are non-functional in nature, and many non-functional requirements techniques have been proposed in the literature [Kassab, 2009, Mylopoulos, et al., 1999, Surhone, et al., 2010], to the best of my knowledge, those techniques are not being used in SLA specification.

SLAs are important during the service design phase [ITIL3D, 2007] and during operation, to guarantee that service delivery matches the agreed-upon levels and that it can be continuously improved [ITIL3C, 2007]. Despite the current availability of applications for supporting service level management, SLA definition and monitoring are still identified as open issues within the ITSM domain [Coyle and Brittain, 2009]. In this research plan we are going to focus our attention on three specific problems in this context:

1. *SLAs in the context of ITSM are informally specified in natural language;*
2. *SLAs definition is not grounded on models of ITSM processes;*
3. *SLA compliance verification is not performed at the same level of abstraction as service design.*

The conjunction of these problems has hard consequences to ITSM practitioners. The first problem arises from the current practice in SLA specification for IT services, mostly based in templates filled with natural language descriptions [Wedemeyer and Menken, 2007]. Such descriptions are not amenable to the automation of SLA compliance verification. Since SLA definition is subjective in nature it renders the second problem: it is not clear to all involved stakeholders which are the activities upon which SLA compliance should be verified, and how their non-compliance will affect the evolution in the process describing the IT service delivery. The third problem concerns a semantic gap which occurs in SLA compliance verification. This gap comes from the fact that compliance checking is based mostly in technical information about hardware, network, and applications (e.g. device performance and failures, network links' availability and latency, applications response times), collected by systems management applications, instead of consolidated data presented on the process model representation of the IT service.

Somehow the first problem is similar to the one object-oriented modeling languages faced previously to the Unified Modeling Language (UML) [MOF, 2006] [UMLs, 2007] unification effort. Diagrammatic notations were not able to capture business rules, so the latter were represented in natural language requirements aside from model diagrams. It was not possible to generate code from

those requirements and traceability to implementation was hampered. With the advent of UML came the Object Constraint Language (OCL) [OCL, 2006], a constraint language that allows expressing business rules textually, based on the modeling concepts (e.g. classes and associations) represented graphically.

The main expected contribution of this research work is mitigating the above problems by proposing a model-based approach to SLA specification and compliance verification. The SLA specification will be derived by way of domain specific languages (DSL). Through SLA specification we seek to elicit quality attributes, applying techniques like the ones from requirements engineering, and specifically from non functional requirements (NFRs) (e.g. NFR, i* and KAOS frameworks), to the context of ITSM. The metamodel for the SLA specification will be an extension of the metamodel of the adopted process modeling language of IT services. As such, it will be possible to ground SLA contracts definition on the corresponding IT service model constructs. The association of DSL with frameworks of non-functional requirements was previously made with other purposes in [Nunes, et al., 2009, Patrício, 2010]

Regarding SLA compliance verification, we plan to use a SLA evaluator tool that will be able to run statements of the SLA language. This component will import SLA contracts expressed in the SLA language, as well as information of IT service instances execution, delivered by a process animation tool. The current state of the delivered IT services will be depicted, with visual tags, by that process animation tool, after accessing to events captured by IT service and system management tools. The SLA evaluator will supply information to a SLA visualizer, to display current levels of service being provided, namely if SLA limits have been surpassed. This environment will allow monitoring process execution and SLA compliance at the abstraction level understood by the stakeholders involved in IT service specification and deployment.

We plan to validate this approach in the context of an IT service management project on the domain of financial self-service systems.

The rest of this research plan is organized as follows: (a) in section 2 we present a survey on the current state of the art of SLAs and NFRs; (b) in section 3 we detail the problems to solve, the research questions and the derived research hypotheses, and identify the experiments planned to test the hypotheses. We underline the main challenges of this research, the steps towards our proposal and its architecture, as well as the main expected contributions; finally (c), in section 4, we present our methodology and the detailed plan for our research work.

2. Research Context

When discussing related work, it is useful to have a common taxonomy. A taxonomy-based analysis promotes a more systematic study of different approaches than the one usually achieved through a traditional, non-structured, descriptive review. The taxonomy also helps identifying which approaches are likely to be more adjustable to our own context and the issues that are not handled by current approaches and our proposal intends to tackle.

2.1. Service Level Agreements

For surveying the current state of the art in SLA specification and compliance validation, we propose a taxonomy with four dimensions to describe the entries of the survey:

- **Domain**: it refers to the technological context at which the SLA is defined and evaluated. In a nominal scale we consider:

  (1) *ITSM* – regarding processes of an ITSM framework;
  (2) *SOA* – concerning web services interacting in an Service-Oriented Architecture (SOA) environment [Erl, 2007];
  (3) *System Management* – related with services of IT infrastructure assessed by QoS parameters.

- **Formalization**: it refers to the level of formalism [Wieringa, et al., 1997] used for SLA specification. We consider an ordinal scale that classifies the approaches, in descending order, as:

  (1) *Formal* – the SLA language has formal semantics, i.e. it has an underlying mathematical model that describes the possible computations described by the language;
  (2) *Semi-formal* – the SLA language is depicted through diagrammatic and/or tabular techniques that present information in structured form. Although the representation might not cover all details of the reality, the representation itself and the system description are equivalent;
  (3) *Informal* – SLA is described in natural language, usually using a structured template form.

- **Abstraction Level**: it refers to the kind of representation used in SLA. We consider an ordinal scale that classifies the approaches, in descending order, as:

  (1) *Model-based* – diagrammatic models for SLA elicitation and specification;
  (2) *Hybrid* – a blended approach using a diagrammatic notation supplemented by a textual language;
  (3) *Textual* – specification exclusively based in textual language.

- **Compliance Verification**: it refers to the level where SLA validation occurs. We consider an ordinal scale that classifies the approaches, in descending order, as:

  (1) *Model-based* – the SLA compliance is checked at the level of models;
  (2) *Message-based* – the compliance verification is based in the exchanged messages, among clients and providers, via uncoupled web services using standards such as SOAP/WSDL or REST;
  (3) *Data-based* – the compliance verification is based on information available in databases or other repositories (e.g. CMDB), with known schemas, stored by applications that deal with and support special types of IT infrastructure or specific IT service management processes;
  (4) *Absent* – there is no SLA compliance checking.

2.1.1. SLA in SOA

The field of computer science where SLA specification has gained more attention is the one of SOAs, and in particular web services [Bianco, et al., 2008]. Some examples follow:

- The language and framework named Web Service Level Agreement (WSLA) is a specification and reference implementation that provides detailed SLA specification capabilities and enables the runtime monitoring of SLA compliance [IBM, 2001];

- The Web Services Agreement Specification (WS-Agreement), defines an XML-based language for agreements, as well as a protocol for advertising the capabilities of service providers, creating agreements between service consumers and providers, and monitoring agreement compliance [OGF, et al., 2007]. Like the WSLA, the WS-Agreement provides an XML schema that defines the overall structure of an agreement document. In addition, the WS-Agreement specification

defines a protocol for dynamically negotiating and establishing SLAs, based on web services. The structure of a WS-Agreement XML agreement is extensible. It contains several sections where intended users are expected to define domain-specific elements and properties. For instance in a space reservation manager application, a system for storing and retrieving large amounts of data, relevant properties are space allowed, maximum allocation size, maximum reservation lifetime. WS-Agreement does not go into each domain-specific area, which cannot be addressed uniformly except for specific common definitions for terms such as cost/price, re-negotiation policies, etc. Therefore, domain-specific languages have to be agreed and implemented by WS-Agreement users, or existing languages can be reused. In general, infrastructure provisioning requests may be addressed by embedding Open Virtualization Format (OVF) [OVF, 2010] requests in WS-Agreement offers; for grid storage reservations, Storage Resource Manager (SRM) [SRM, 2009] can be used as a term language; for job submission, Job Submission Description Language (JSDL) [JSDL, 2005] is a candidate. WS-Agreement's extensible characteristics allow the inclusion of any language deemed appropriate.

The following are other approaches intended to the specification of the quality attributes in a SLA:

- The Web Services Offering Language (WSOL): enables the "formal specification of functional constraints, some QoS constraints, simple access rights (for the differentiation of service), price, and relationships with other service offerings of the same web service" [Tosic, et al., 2006];
- The Web Service Management Layer (WSML) proposal defines a layer that goes between an application and web services and allows the specification of nonfunctional characteristics of a service [Verheecke, 2003];
- The Web Service Modeling Ontology (WSMO) is used to specify aspects of semantic web services, through the description of nonfunctional properties (e.g. cost, accuracy, network related QoS, performance) [W3C, 2005];
- The WS-Policy proposal is related to the specification of service qualities that are part of a SLA [W3C, 2007]. WS-Policy allows web services to advertise their policies and web service consumers to specify their policy requirements. WS-Policy is often used in conjunction with other specifications, such as WS-Policy Attachment, for associating specific policies to web services;
- The WS-Policy Attachment is a specification conforming to the WS-Policy, for attaching policy assertions to policy subjects [W3C, 2007b];

Other projects related to SLA specification are:

- Rule-Based Service Level Agreements (RBSLA) is a project that uses knowledge representation concepts for the specification of SLA [Paschke, 2005]. It provides a set of abstract language constructs based on RuleML [RuleML, 2010] to define SLA/policy rules;
- SLAng is a language for SLA definition using a combination of Meta-Object Facility (MOF) [MOF, 2006], OCL, and natural language [Skene, et al., 2004] [Skene, 2007]. According to its authors, the SLAng definition using a meta-modelling approach allows a high degree of precision in the specification of its semantics, traceability from SLA to language specification, and the testing of the language and SLAs to ensure they capture the original intent of the parties. SLAng supports the expression of mutually-monitorable SLAs, for which the determination of compliance depends only on events visible to both client and provider of the service.

As summarized in Table 1, all the mentioned approaches are related in some sort to SOA and web services (Domain – 2). Therefore they are more able of addressing computer based processes, more

rigorously defined (Formalization – 1), than processes that relies in human intervention, such as the ones covered by ITSM processes. Moreover, the compliance verification is made by messages and events exchanged at an implementation level (Compliance Verification – 2), which is difficult to grasp by non technical stakeholders (Abstraction Level – 3/2), more acquaintance with higher levels of abstraction.

Table 1 – Summary of SOA approaches to SLA specification and validation

Approach	Domain	Formalization	Abstraction Level	Compliance Verification
[IBM, 2001] [OGF, et al., 2007] [Tosic, et al., 2006] [Verheecke, 2003] [W3C, 2005] [W3C, 2007] [W3C, 2007b] [Paschke, 2005]	(2)	(1)	(3)	(2)
[Skene, et al., 2004]	(2)	(1)	(2)	(2)

2.1.2.SLA in ITSM

In Table 2 we summarize the current status, regarding the four dimensions, of service management applications and systems management tools.

There are tens of service management applications available, that support a large number of ITIL processes, including service level management [PinkElephant, 2010]. However there are some reported issues [Coyle and Brittain, 2009], about their effectiveness regarding SLAs support. SLAs are informally specified (Domain – 1); SLA definition is not linked with models of ITSM processes (Formalization – 3); and SLA compliance verification in IT services is not performed at the same level of abstraction of service design, but instead is based in dashboards (Abstraction Level – 2) and fine grained metrics collected from scattered and not integrated repositories of data (Compliance Verification – 3).

Concerning systems management tools [Gläßer, 2005] – which monitor network, IT devices and applications at QoS parameters level – the main issue is that they are essentially focused in the technical infrastructure (Domain – 3). So, the data they collect regarding mostly IT services must be previously consolidated and transformed before they can be presented and analysed by IT providers and consumers. Therefore, the rank of Formalization, Abstraction Level and Compliance Verification dimensions follow the pattern previously mentioned of service management applications.

Table 2 – Summary of Service/System Management approaches to SLA specification and validation

Approach	Domain	Formalization	Abstraction Level	Compliance Verification
Service Management	(1)	(3)	(2)	(3)
System Management	(3)	(3)	(2)	(3)

The information summarized in Table 1 and Table 2, allow us to conclude that there is ground for a research work concerned with the specification of a model-based approach to SLA specification and compliance verification of IT services. This research work should allow stakeholders to specify SLAs at the same level of abstraction as SLA compliance verification.

2.2. Quality Attributes and Non Functional Requirements

The IEEE defines, in the Software Engineering glossary, non-functional requirements as "*a software requirement that describes not what the software will do, but how the software will do it, for example, software performance requirements, software external interface requirements, design constraints, and software quality attributes*" [Thayer, 2003]. Different approaches have been proposed for dealing with NFRs in the requirements phase. They range from unstructured and informal text to highly formal mathematical approaches. The approach chosen in each case depends on the goals of the project and the available resources [Matoussi and Laleau, 2008].

To specify SLAs we need to formalize SLA parameters in terms of quality attributes, by means of an SLA language. This language should be able to link quality attributes with processes of IT services. In order to achieve this goal, we need to overcome some challenges in the elicitation of SLA quality attributes, specifically, deal with their diversity, subjective nature, and the conflicts that rise among them.

To derive our SLA language we will focus our attention on available NFRs elicitation techniques, due the similarity among SLA quality attributes and NFRs concepts [Röttger and Zschaler, 2006]. For analysing the current state of the art in NFRs, we propose another taxonomy to describe the entries of the survey, with four dimensions closely resembled with the ones proposed for the SLA survey:

- **Domain**: it refers to the technological context at which the quality attributes/NFRs are defined and evaluated. In a nominal scale we consider:

  (1) *ITSM* – regarding processes of an ITSM framework;
  (2) *Software/Requirements Engineering* – NFRs elicitation is performed in the context of Software Engineering;
  (3) *QoS* – related with QoS parameters of IT infrastructure.

- **Formalization**: it refers to the level of formalism [Wieringa, et al., 1997] used for quality attributes/NFRs specification. We consider an ordinal scale that classifies the approaches, in descending order, as:

  (1) *Formal* –techniques that have an underlying mathematical model;
  (2) *Semi-formal* –diagram techniques and/or tabular techniques that present information in structured form;
  (3) *Informal* – descriptions in natural language usually using a structured template form.

- **Abstraction Level**: it refers to the kind of representation used in the elicitation and specification of quality attributes/NFRs. We consider an ordinal scale that classifies the approaches, in descending order, as:

  (1) *Process-based* – a diagrammatic notation for modelling processes (e.g. BPMN [BPMN, 2009]);
  (2) *Ontology-based* – a diagrammatic notation for modelling concepts and their relationships (e.g. UML class diagram);
  (3) *Specific* – an ad-hoc diagrammatic notation for quality attributes elicitation (e.g. NFR Framework);
  (4) *Textual* – specification are exclusively based in textual language.

- **Coverage**: it refers to the number of quality attributes/NFRs considered by the approach. We consider an ordinal scale that classifies the approaches, in descending order, as:

(1) *Broad* – the approach includes four or more measurable quality attributes / NFRs;
(2) *Few* – the approach includes two or three measurable quality attributes / NFRs;
(3) *Single* – the approach includes only one measurable quality attribute / NFRs;

2.2.1. Informal and Semi-formal Approaches

Informal and semi-formal approaches are more human-nature oriented than formal ones, but do not ensure that the obtained results are non ambiguous. We will now survey some of those approaches to specify NFRs.

The *NFR Framework* [Chung, et al., 2000] considers NFRs as *soft goals* (goals difficult to express) to be addressed during the development process. NFRs, design decisions and the relations among them are depicted in a goal graph where the nodes are either NFRs or design decisions. Goals can be refined into detailed concrete goals. The relationships between NFRs and intended decisions are made explicit, so, according to its authors, it is easier to understand the positive or negative impact of design decisions in multiple NFRs. This framework is reused by other models to handle NFRs, such as for instance in [Kassab, et al., 2007] that presents a semi-formal model that extends the taxonomy of the NFR Framework by integrating the concept of ”hard goals NFRs”. This model details the sequence of systematic activities towards an early consideration of identifying, specifying and splitting functional requirements and NFRs.

In the informal procedure presented by [Kaiya and Kaijiri, 1999] the evaluation of stakeholders is relevant. Some stakeholder requirements conflict with others, since many kinds of stakeholders participate in a system development. Therefore, a criterion shared by the stakeholders is required to coordinate the requirements among them or to establish trade-offs. NFRs have this role because of their influence in most of the stakeholders concerns. The NFRs are also used to check the stakeholders' satisfaction.

The *Quality Attributes taxonomy* [Barbacci, et al., 1995] is a semi-formal proposal which measures the quality attributes impact in the software architecture. The terminology used in the taxonomy serves as a vocabulary to specify NFRs and drives the design of the architecture. The taxonomy is divided into four areas: performance, dependability, security, and safety. All quality attributes are analyzed through three dimensions: (1) *concerns* - the parameters by which the attributes of a system are specified; (2) *factors* - the properties of the system and its environment that have an impact on the concerns; and (3) *methods* - how to address the concerns.

[Drr, et al., 2003] presents a semi-formal approach, inspired in the previous taxonomy, for eliciting and documenting *efficiency* requirements with Use Cases and a high-level architecture. Quality attributes and quality models are used to capture general knowledge on NFRs, while specific NFRs are captured in a template. This approach is particularly interesting for our research work, since it uses a generalized efficiency metamodel, as well as a quality model that can then be used for other NFRs.

According to its authors, using the *Performance Requirements Evolution Model* (PREM) [Ho, et al., 2006] as a guideline, a software development team can identify and specify performance requirements incrementally, from casual descriptions, to the adequate level of detail and precision. This model addresses essentially the specification and testing of the NFR “performance”.

Misuse cases (aka functions that the system should not allow) [Herrmann and Paech, 2005] [Sindre and Opdahl, 2005] [Firesmith, 2003], based on UML, handles security NFR.

KAOS (Knowledge Acquisition in autOmated Specification) [Lamsweerde, 2001] [Letier, 2001], a semi-formal approach, was initially targeted to the elicitation, analysis and modelling of functional requirements. KAOS requirements model is based on first-order temporal logic and it is composed of several sub-models related through inter-model consistency rules. Later, KAOS was extended to handle NFRs such as security requirements, so it can elicit security goals such as confidentiality, integrity, availability, privacy, authentication and non-repudiation [Lamsweerde, 2004].

2.2.2. Formal approaches

In this section we present some research works on the formalization of NFRs. The latter include:

Formal Design Analysis Framework (FDAF) [Cooper, et al., 2003] intends to support the systematic design of a system that meets its NFRs. The UML (standard, formalized with OCL, and extended) and other formal methods have already been used to describe architectures. According [Matoussi and Laleau, 2008], a major limitation of FDAF is the variety of formal notations used. It becomes a hindrance to the study of different interactions and conflicts detection between NFRs.

[Aagedal, 2001] [Zschaler, 2004a] [Zschaler, 2004b] proposed formal specification languages for *NFRs of component-based systems*. [Zschaler, 2004a] introduced a formal specification using extended temporal logic of actions (TLA+) to describe the system. The logic is a temporal logic where states are represented as values assigned to state variables. CQML (Component Quality Modeling Language) [Aagedal, 2001] is another specification language for components QoS properties. CQML+ [Röttger and Zschaler, 2006], an extension of CQML, builds on quality characteristics, which essentially are definitions of measurements. Quality statements are used to specify constraints on characteristics. Both quality characteristics and quality statements are parameterized and can, therefore, be reused in different contexts. CQML+ uses a modified version of OCL for the specification of the formal meaning of a characteristic. CQML+ comprises both measurement definition and measurement usage. For measurement usage it was defined a graphical notation allowing to attach constraints to parts of the functional model.

The language *NoFun* (NOn-FUNctional) [Botella, et al., 2001], another NFR formalization proposal, consists of three different parts: (1) software quality characteristics and attributes definition; (2) the assignment of values to component quality basic attributes; (3) quality requirements can be stated over components, both context-free and context-dependent. The language contains structuring mechanisms, type definition elements and other constructs that give an appropriate support for defining non-trivial quality models. A mapping of these concepts to UML is made using extension mechanisms.

As summarized in Table 3 the mentioned NFRs elicitation approaches are mostly related with requirements engineering (Domain – 2), with formalization that ranges from formal to informal methods (Formalization – 1 to 3), and covering from one to several quality attributes (Coverage – 1 to 3). However, their notations do not include the representation of quality attributes in processes, an abstraction more suitable for filling the gap between design and monitoring of ITSM processes.

Table 3 – Summary of NFRs elicitation approaches

Approach	Domain	Formalization	Abstraction Level	Coverage
[Chung, et al., 2000] [Kassab, et al., 2007]	(2)	(2)	(3)	(1)
[Kaiya and Kaijiri, 1999]	(2)	(3)	(4)	(2)
[Barbacci, et al., 1995]	(2)	(2)	(4)	(2)
[Drr, et al., 2003]	(2)	(2)	(2)	(3)
[Ho, et al., 2006]	(2)	(3)	(4)	(3)
[Herrmann and Paech, 2005] [Sindre and Opdahl, 2005] [Firesmith, 2003]	(2)	(2)	(3)	(3)
[Lamsweerde, 2001] [Letier, 2001]	(2)	(2)	(3)	(3)
[Cooper, et al., 2003]	(2)	(1)	(2)	(1)
[Aagedal, 2001] [Zschaler, 2004a] [Zschaler, 2004b] [Röttger and Zschaler, 2006]	(3)	(1)	(4)	(1)
[Botella, et al., 2001]	(2)	(1)	(2)	(1)

In Requirements Engineering, informal and semi-formal approaches are the most used to specify NFRs. However, a common problem in using formal methods for specifying NFRs is the high cost and difficulty of applying them. The cost could be probably justified, only to crucial NFRs such as security requirements. However, being NFRs one of the critical features of a system, failures in its specification, may result in serious losses in terms of time, money and data [Matoussi and Laleau, 2008].

The kind of problems, previously stated regarding NFRs and Requirements Engineering, is also common to the quality attributes of IT services, in the context of ITSM. However, we have to bear in mind that IT services quality attributes and NFRs have different levels of granularity. The former refer to coarse grained properties. The NFRs are related to more fine grained characteristics of software systems.

Nevertheless, the already known Requirements Engineering techniques of NFRs elicitation, seems to be a good starting point to figure out how to draw out quality attributes in the context of ITSM, for SLA definition. So, our SLA language must be able to express IT services quality attributes in order to improve the rigor of SLA specification. In the meanwhile we must also address the problem of traceability among quality attributes of IT services applications and technical infrastructure.

3. Research Statement

3.1. Research Drivers

The starting point of our research process is to state which are the problems we are trying to address (3.1.1). This will drive us to our research questions (3.1.2). To answer these questions we have to formulate some hypotheses. The explanatory theory of the problems posed by the research questions is embodied in the research hypotheses (3.1.3). To evaluate our contribution to the problem's solution, we have to make some experiments (3.1.4). If the experiments are properly set up, we

should be able to draw some conclusions about the effectiveness of our proposal [Wohlin, et al., 2000].

3.1.1. Problems to Solve

A major drawback of SLA in organizations implementing ITIL is concerned with the fact that SLAs are specified using general guidelines, and templates. Unlike other research areas such as SOA [Bianco, et al., 2008], there is no formal specification proposed for SLAs in the ITSM domain.

An example of general guideline is specified in the ITIL's process of Service Level Management (SLM). It states that "*all of the appropriate and relevant customer requirements, at all levels, must be identified and incorporated in SLAs*". This include the requirements of staff such as senior managers, which "*may rarely use a service and may be more interested in issues such as value for money and output*", as well as junior members that "*may use the service throughout the day, and may be more interested in issues such as responsiveness, usability and reliability*" [ITIL3D, 2007].

The SLM process also defines the SLA life cycle, spread out by service phases of Service Design [ITIL3D, 2007] and Continual Service Improvement [ITIL3C, 2007]. In Service Design there is the SLA *negotiation* phase (cf. Figure 3), where the SLA's structure is designed and the customer requirements' elicitation is made in order to produce a high level specification of the SLA. After this initial phase, the IT service (or a specific instance of it) is made available for consumption. This may require the reconfiguration of the resources that support service execution in order to meet SLA parameters. It is in Continual Service Improvement that occur all the other next SLA phases. The *monitoring* and *reporting* happen during the service execution, i.e., the actual operation of the service. In the *evaluating* phase the SLA provided to consumer is assessed for service quality validation, and violation checking. The *improving* phase periodically reviews for each SLA, consumer satisfaction, potential improvements, and changing requirements. The information needed for mentioned Continual Service Improvement phases are collected by system management and service management tools, which allow the realignment of service goals and operations, or the definition of different service levels, when problems in service support are identified. In Figure 3, we coloured the regions where are located the states concerning the SLA Specification and Validation, the main focus of this research.

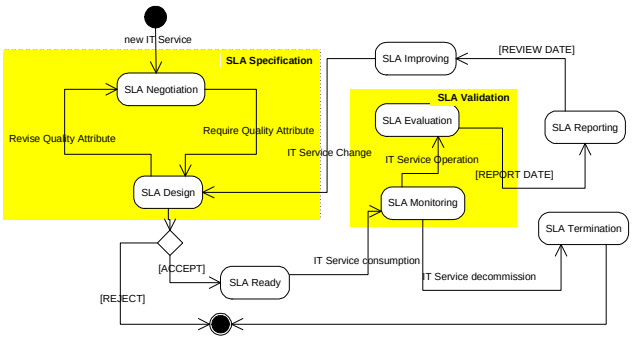

Figure 3 – State chart of SLA life cycle according ITIL v3

ITIL provides a template of an SLA contract ([ITIL3D, 2007], Appendix F), which establishes that for the SLA parameter of service availability, it must be specified the target availability levels (expressed as percentages), measurement periods, the methods and calculations. However, the

template filling instructions acknowledge the difficulty in linking the target availability percentage with service quality, or with customer business activities. Since the ITIL does not establish other more prescriptive guidelines for SLA specification, the automation of SLA life cycle by organizations, as specified in the SLM process, is difficult to achieve, and became an hindrance for the ITIL full implementation, given the central role of the SLM process in the framework [Coyle and Brittain, 2009].

The main difficulties around the SLAs in the context of ITIL can be summarized as follow. On one hand, there is a lack of more precise SLA specification of IT services in ITIL, which hampers the automatic processing of SLA life cycle. On the other hand, and related with the previous difficulty, no mapping is usually built among SLA and ITSM processes that execute IT services, in order to perform an effective monitoring of SLAs. Finally, the SLA compliance verification is hampered because the monitoring of IT services is made at an implementation level, in service management and systems management applications, instead of being consolidated in high level models, resulting in a semantic gap for the compliance verification of SLAs.

Based on the abovementioned SLAs issues, the aim of our research is to provide contributions for facing three major problems[3]:

- (**P1**) SLAs in the context of ITSM are informally specified in natural language;
- (**P2**) SLAs definition is not grounded on detailed models of ITSM processes; and
- (**P3**) SLAs compliance verification is not performed at the service design level.

3.1.2.Research Questions

With the elicitation of real-world problems in the previous section, we can now come up with related research questions that will direct our research.

- **(P1.Q1)** Can a formal model of SLA improve the efficiency of building SLA contracts by stakeholders?
- **(P1.Q2)** Does the SLA language allow to accurately specify the constraints required for an automatic validation of SLA parameters?
- **(P2.Q1)** Does the SLA language improve processes representation, by allowing to monitor IT services delivery and automatic validation of SLAs?
- **(P3.Q1)** Can SLAs compliance verification, performed at service design level, be as effective as the one performed at the level of service implementation?

3.1.3. Research Hypotheses

We are now able to formulate the research hypotheses, which can be implicitly deduced from the research questions. By testing the research hypotheses, we intend to validate the degree of accomplishment of our research work. Here we have only specified the H_0 hypothesis, the one that we want to disprove using empirical evidence, grounded in real data. The H_1, which can be easily derived

[3] We label the problems as P1, P2 and P3, for the sake of traceability with research questions, and research hypotheses.

by denying each H_0, express what we are actually trying to show. If H_0 is not accepted, we can conclude that there is no empirical evidence to disprove H_1.

(P1.Q1.Y1)

- H_0: The formal model of SLA is as efficient[4], in defining SLA parameters, as traditional methods based in technical knowledge of IT infrastructure, processes and services.

(P1.Q2.Y1)

- H_0: The SLA parameters, expressed in the SLA language, for an automatic validation of SLA parameters are, as detailed and accurate[5] as the ones expressed by traditional methods.

(P2.Q1.Y1)

- H_0: The IT services process monitoring, supplemented by extensions introduced by the SLA language, provides the same level of effectiveness[6] on catching SLA violations as the traditional methods.

(P3.Q1.Y1)

- H_0: The process of SLA compliance verification, performed at service design level, is as efficient[7] as the one performed at the level of service implementation.

3.1.4. Experiments

As part of the scientific method utilization, we will have to do an empirical assessment in order to test the research hypotheses. These assessments will go through the following steps:

1. Experimental Design – In this step it will be planned the information gathering of some treatments on some objects (user groups, IT services). Dependent and independent variables of the problem must be determined for controlled experiments, quasi-experiments or statistical surveys.
2. Data Collection – This step will be accomplished by acquisition and registering of events captured during the monitoring of IT services related with real operation of financial self-service terminals. The events are concerned the opening, handling and closure of terminals' incidents;
3. Data Filtering – In this step, the erroneous data (e.g. invalid data, missing values) will be filtered out;
4. Data Computation – Required variables will be computed from existing data. Collected data will be organized and analysed in order to identify factors influencing SLAs parameters. Data related to compliance of established SLAs will be computed;
5. Statistical Analysis –The resulting dataset is loaded into the statistical data analysis tool where research hypotheses will be tested.

---

[4] Efficiency can be measured in several ways, such as time required for defining and justifying a SLA parameter and interaction reduction in communication among clients and IT service providers.

[5] Accuracy is related with produced figures regarding of SLA parameters and their traceability in the process implementation.

[6] Effectiveness relates to getting the right things done, i.e. SLA violations captured when agreed level of services, specified by thresholds of SLAs parameters, are not reached.

[7] The efficiency in this context can be measured in the amount of time and resources required to SLA compliance verification

Some of the experiments to be performed in order to conclude about the research hypotheses of the previous section are the following:

(P1.Q1.Y1.T1)

- Two independent groups[8] of IT users with equivalent knowledge and experience about SLA and IT services will try to establish and justify SLA parameters (e.g. availability) of SLA contracts. One of the groups will use the SLA model, built in the context of this research, for SLA formalization. The other will use traditional methods based on technical information about hardware devices, network links and peripherals, applications performance and IT processes and services. The efficiency attained by each group can be measured by the time spent to specify the SLA parameter and its level of detail. To run the experiment we plan to use the following variables:[9]

✓ Dependent variable: *Total Time Spent* (ratio[10]) in the SLA quality attributes elicitation of SLA contracts;

✓ Independent variable: *IT Service Complexity* (ratio) attained by a process metric; *Technical Infrastructure Complexity* (ratio) attained by an appropriate metric; SLAVE (nominal) states if the group used or not the SLAVE tool (made available by this research work), or service management and systems management applications were used.

(P1.Q2.Y1.T1)

- Given a sample of statistically relevant SLA violations of IT services instances, the SLAVE tool can accurately identify in the process model the source of violations, in the same way that service management and systems management applications traditionally used for monitoring. The accuracy attained is measured by the level of detail at which is possible to trace the origin of the violation. To run the experiment we plan to use the following variables:

✓ Dependent variable: *Process Level* (ordinal) at which the SLA violation was detected;

✓ Independent variable: *IT Service Complexity* (ratio) attained by a process metric; #Service Management *Incidents* (ratio); #*Technical Infrastructure Incidents* (ratio); SLAVE (nominal) states if the SLAVE tool, or service management and systems management applications were used in the diagnosis.

(P2.Q1.Y1.T1)

- Given a sample of statistically relevant IT services instances, the SLA model can detected the same number of positive SLA violations, as the traditional methods of SLA monitoring, based in reports extracted from service management and systems management applications. To run the experiment we plan to use the following variables:

✓ Dependent variable: *#SLA Violation* (ratio) as the number of SLA Violations detected;

---

[8] The experimental design will be preferably based in experiments using different groups of individuals (between-groups), minimizing the possibility of the members of a group (within-groups), learn with previous experiments and thus, biasing the results.

[9] In a conducted experiment the types of variables are: (1) *independent variables* (or *factors*) which are manipulated by the experimenter; and (2) *dependent variables* are measured from the subjects.

[10] Variables can be *quantitative* and *qualitative* (or *categorical*). The former variables can be measured on an *ordinal*, *interval*, or *ratio* scale; qualitative variables are measured on a *nominal* scale.

- ✓ Independent variable: *IT Service Complexity* (ratio) attained by a process metric; #Service Management *Incidents* (ratio); #*Technical Infrastructure Incidents* (ratio); SLAVE (nominal) states if the group used or not the SLAVE tool, or service management and systems management applications were used in the diagnosis.

(P3.Q1.Y1.T1)

- Two independent groups of non IT stakeholders with identical knowledge and experience about SLA and IT services will try to explain the source of SLA violations. The time needed to understand of SLA violations will be measured by two groups of stakeholders. One of them will use information provided by the SLA model (service design level) and the other information extracted from service management and systems management applications (service implementation level). To run the experiment we plan to use the following variables:

- ✓ Dependent variable: *Total Time Spent* (ratio) for explaining the source of SLA violations;

- ✓ Independent variable: *IT Service Complexity* (ratio) attained by a process metric; *User IT Literacy* (ordinal); *Incident Origin* (nominal) due to a technical or human failure; SLAVE (nominal) states if the group used or not the SLAVE tool, made available by this research work, for SLA specification.

3.2. Challenges for Problem Solving

The challenges for addressing the research problems (3.1.1) have been grouped in several sets. The first group of challenges is related to the expressiveness power of the SLA language, for an accurate specification and automatic validation of SLA parameters. The second category is concerned with addressing a higher level of abstraction, in the monitoring of IT services, through a process model representation. Finally the third category is related with the need of data collection and consolidation from disparate sources and levels of granularity.

Regarding the first group of challenges we must considerer:

- As happens with functional requirements and NFRs [Bass, et al., 2003] in software systems, ITSM functionalities and quality attributes are orthogonal. This means that it is possible to choose the desired level of each quality attribute in an independent way. However, this not means that any level of a certain quality attribute is achievable for a specific IT service. For a specific function, the kind of IT service implementation will determine its relative level of quality. The choice of service implementation can be enforced by a SLA contract. Therefore SLA specification has to consider that quality attributes cannot be achieved in isolation. The quest of a certain level of any quality attribute will have a positive or negative effect, on the level attained by other quality attribute. The requirements engineering techniques for dealing with conflicting non-functional requirements [Mylopoulos, et al., 1999] can help us to address quality attributes in the context of ITSM. In the following examples, we illustrate the tension among quality attributes:

  - security and reliability - a most secure system has the fewest points of failure, however a most reliable system has the most points of failure (e.g. a set of redundant processes or processors where the failure of any one will not cause the system to fail);

  - performance and portability - performance is negatively affected by almost every quality attribute, namely portability. A technique for achieve portability is to isolate system

dependencies, which introduces overhead into the system's execution, and hurts its performance.

- Quality attributes are more easily specified in a SLA, if we are able to measure or verify it[11]. There are qualities more suitable for being automatically measured (e.g. scalability, performance) using quantitative metrics than others more difficult to automate (e.g. credibility[12], auditability[13]).
- Many organizations provide IT services that depend on services from other organizations. For example, an online shop system may interact with (1) banks to authorize credit cards, (2) the warehouse to check availability and reserve the items to be sold, (3) the delivering company for checking availability of transport. A kind of similar challenge rises when an organization providing a service may, in turn, use services from other organizations (commonly called layered or composite services). This is the case for instance when the technical infrastructure is based on cloud computing, where shared resources (e.g. software, information) are provided on-demand, like a public utility. The unavailability or poor performance of one of the services may compromise the whole IT service.

The second group of challenges is concerned with addressing a higher level of abstraction, in the SLA specification, as well as in the monitoring of IT services.

- The SLA specification can be addressed by a general purpose modeling language or a more specific and constrained approach; each of them has advantages and disadvantages, namely concerning the ability of dealing with new concepts and services not explicitly addressed in the language specification.
- Since the focus of this research is presenting and monitoring IT services at higher levels of abstraction, a process technique is required that is able of representing IT services at service design level and also trace SLA non compliance at processes' detailed level.

Finally the third group of challenges is related with the need to ensure the SLA monitoring at model level, data collected from disparate sources that need to be integrated and consolidated. The sources, from which we expect to collect data with several levels of granularity, using standards for data storage and interchange, are the following:

- By means of system management tools one can access fine grained data about the technical infrastructure and applications behavior that support IT services execution.
- In service management applications one can access IT services specification and its current status in order to check SLA compliance.

3.3. Approaches and Tools for Problem Solving

This research work is framed by the Model-Driven Engineering (MDE) paradigm. MDE was operationalized by the OMG's initiative Model-Driven Architecture (MDA) [MDA, 2001], which

[11] We are not addressing in this research work legal or accountability subjects of SLAs.
[12] It refers to the believability of a source or message, and has two key components: (a) trustworthiness, based more on subjective factors, but can include objective measurements such as established reliability; and (b) expertise that can also be subjectively perceived, and includes relatively objective characteristics of the source or message (e.g., certification or information quality) (http://en.wikipedia.org/wiki/Credibility).
[13] A non-functional requirement concerned with the transparency of a system with regards to external audits.

refers to the systematic use of models as primary engineering artefacts, to express domain concepts effectively, throughout the engineering lifecycle. The MDE assumes that any problem can be represented by a model which captures its essence [Frankel, 2003] [Mellor, et al., 2004].

In the present research work, the problem domain is related with the SLA specification, and compliance verification in the context of IT services. In order to address the problem, our intention is to adapt NFR techniques from requirements engineering domain [Chung, et al., 2000] to the ITSM domain. The NFR approaches allow elicitation of non-functional requirements, grounded in knowledge bases and functional requirements documentation [Cysneiros, et al., 2001]. By following the MDE approach, a set of model transformations from the solution specification models (the SLA contracts) to the implementation models (the SLA language) will be accomplished. This will allow the generation of validation statements in the SLA language, based on information of SLA contracts.

Several alternatives exist for expressing contracts upon IT service process models. One would be to extend the general-purpose UML modeling language with a profile that would restrict the scope to a particular domain with stereotypes, tagged values and constraints. This alternative has the advantage of tools availability, although the effort for creating a new profile is usually considerable. However, the main disfavor of this alternative is that the representation richness of UML renders the corresponding modeling tools excessively complex for IT service designers and clients.

To mitigate the previous problem we adopted a Domain-Specific Modeling (DSM) approach [Langlois, et al., 2007]. In other words, we are developing a Domain Specific Language (DSL) for expressing SLAs. DSLs allow solutions to be expressed in the idiom and at the level of abstraction of the problem domain. Consequently, domain experts can more easily understand, modify and validate specifications at the domain level. Summing up, we believe that this choice will reduce the effort to express a contract specification. Moreover, a DSM environment, aka graphical DSL tool, can automate the creation of tools that are costly to build from scratch, such as the editor required for specifying SLA contracts [Kalnina and Kalnins, 2008].

So, we need as domain expert, to specify the domain specific constructs and rules, in order that the DSM environment provides a separate stand-alone program: the SLA editor tailored for the service level management domain. To define the SLA contracts supported by the editor, we will need to specify the domain. This will be achieved by producing a metamodel for SLA contracts, as an extension to the metamodel of the process modelling language itself (see section 3.4). These two metamodels can be built using a meta-metamodel language very close of the Essential MOF (EMOF) [MOF, 2006], the ECore[14]. So, the domain-specific language will be composed by of two parts – the domain specification and the SLA editor. By setting up the domain-specific language with the mentioned metamodels, it will be possible to operate with familiar terms related with service level management (e.g. availability, performance), instead of using QoS terms more implementation oriented (e.g. bit rate, bit error, latency).

There are several graphical DSL tool platforms, for domain specific languages development [Pfeiffer and Pichler, 2008]. Some of the currently available tools to be evaluated [Vasudevan and Tratt, 2010] in the context of this work are: MetaEdit+ [MetaEdit+, 2010], Eclipse GMF [GMF, 2010], AToM3 [AToM3, 2010], Microsoft DSL Tools [MSDSL, 2010], DiaGen/DiaMeta [DiaGen/DiaMeta, 2010], Pounamu [Zhu, et al., 2007], Marama [Grundy, et al., 2008], METAclipse [METAclipse, 2010], and GrTP [Barzdins, et al., 2007].

[14] The ECore has been defined in the Eclipse Modeling Framework (http://www.eclipse.org/modeling/emf/)

The next step will be to produce a model transformation of the SLA. Model transformation is a critical component of MDA. Transformations take model specifications as input (the SLA contracts expressed in the SLA language, in our case) and perform actions upon them, resulting in a different model specification (the contract validation clauses, in our case).

The Query/Views/Transformations (QVT) is the current standard compatible with the MDA recommendation suite (UML, MOF, and OCL). Queries and transformations take models as input and perform actions upon them, resulting in a different new model in case of a transformation. *Views* are models, created by some transformation, which are inherently related, by a projection, to the model from which they are created. Several model transformation languages are presently available, with different levels of compliance to the QVT standard. In the present research work the transformations are exogenous, i.e. their source and target metamodels are different[15]. Some tools to be evaluated [Czarnecki and Helsen, 2003] are: GReAT [Balasubramanian, et al., 2006], UMLX [Willink, 2003], VIATRA [OptXware, 2006], MOLA [MOLA, 2010], ATL [INRIA and OBEO, 2010], Kermeta[16], Stratego/XT[17], MT [Tratt, 2005] and Tefkat[18].

Concerning process modelling, there are several well known languages in research and industry that should be evaluated. Among them we highlight the following:

- the UML 2.0 Activity Diagram (AD) [UMLs, 2007] – designed for modelling business processes and flows in software systems, in the software development process. The main concepts are actions and swimlanes (representing roles or actors);
- the Business Process Definition Metamodel (BPDM) [BPDM, 2008] – it is specified as a UML 2.0 profile and does not provide its own graphical notation; it offers a generic meta-model for business processes, in order to support the mapping between different tools and languages;
- Business Process Modelling Notation [BPMN, 2009] – it inherits concepts from other notations namely the Activity Diagram, and besides the support for modelling business processes, it also allows the transformation of models into an execution language (BPEL);
- Event Driven Process Chain (EPC) [Scheer, 1999] – the main goal was to be easily used by non-IT users. The basic concepts of EPC are the function (activity in a business process) and the event (created by processing functions or actors outside of the model);
- Integrated DEFinition Method 3 (IDEF3) [Mayer, et al., 1995] – it provides two perspectives: the process schematics (model of the process sequence) and the object schematics (model of objects and their changing states throughout a particular process). IDEF3 is designed to model business processes and sequences of a system;
- Petri Net [Petri, 1962] – it is a directed graph that mainly uses two different components: places (possible states of the system) and transitions (events or actions which cause the change of state). Petri Nets are designed for modelling, analysis and simulation of dynamic systems with concurrent and nondeterministic procedures. A Petri Net can be used for modelling workflows;
- Role Activity Diagram (RAD) [Holt, et al., 1983] – It shows roles, their activities and interactions, together with external events. The RAD notation has been adopted in many applications involving business processes, namely modelling, simulation and enactment.

[15] In an endogenous transformation, the process of model transformation is made by conversion of a model Ma, conforming to metamodel MMa, into a model Mb conforming to metamodel MMb, where MMa=MMb.
[16] http://www.kermeta.org/
[17] http://strategoxt.org/
[18] http://tefkat.sourceforge.net/

To collect information for SLA monitoring, a set of standards can be used in data consolidation. The first is the OASIS's Data Center Markup Language (DCML) [OASIS, 2005]. The standard is an XML specification for representing components of the data center and information used to manage those components. The DCML is a vendor-neutral language that enables tools to exchange policy, configuration, and operating information in a reliable and standard way.

Another standard to collect information is the Distributed Management Task Force (DMTF)'s Common Information Model (CIM) [CIM, 2010]. CIM is an object-oriented model for unifying and extending existing standards such as SNMP. CIM takes objects from the management domain and divides them into classes based on several criteria, including characteristics and features, relationships, and behaviors. CIM includes the ability to show relationships among objects, including their dependencies.

The CIM Object Manager (CIMOM) allows management of CIM objects on a Web-Based Enterprise Management (WBEM)-enabled system [WBEM, 2010]. A CIM object is a representation, or model, of a managed resource, such as a printer, disk drive, or CPU. In order for hardware devices to be managed using CIM, a management console application must communicate with a CIMOM server. The connection with the CIMOM, which is linked to real hardware, allows the mining of all class schemas. These class schemas can be converted and saved as Managed Object Format (MOF) files.

3.4. Expected Contributions to the Solution

Grounded in the research hypotheses drawn in section 3.1.3 the purpose of the research is summarized in the following thesis statement:

The main objective of this research is to provide a formal specification of SLA for compliance validation, enabled by a Model-Driven approach in the context of the ITSM domain.

To illustrate the feasibility of our approach for SLA specification and compliance validation upon ITSM services modeled in BPMN, we intend to build the *SLA specification and Validation Environment* (SLAVE). Its architecture addresses two main stages in the SLA life cycle: the SLA specification stage, when a *negotiation phase* takes place among the stakeholders; and the *validation stage*, which comprises the SLA life cycle phases of monitoring, reporting, evaluating and improving [ITIL3D, 2007]. Each of those stages is implemented by a different subsystem, schematically depicted by the component diagrams in Figure 4. Grayed components are produced by a third-party.

Our approach considers that ITSM processes were previously modeled with BPMN, by the *ITSM Manager* role, with the support of a BPMN modeling tool, as represented in Figure 4. The resulting *Process Model* can be persisted in a standard format (e.g. XMI).

The *negotiation phase* aims to define the conditions of the SLA contract, agreed between the IT service provider and the customer, and expressed with the SLA language, which will be inspired in techniques of NFRs elicitation [Matoussi and Laleau, 2008]. An *SLA Editor* supports this activity. This tool is used by the SLA Manager role and serves to specify the SLA terms. This editor will be built using a DSL tool.

The output of the SLA Editor will be an *SLA Contract* complying with the SLA metamodel, with hooks on the corresponding BPMN process model. Those hooks specify where in the process a specific contractual clause (which will refer to given SLA quality attributes) applies (e.g. diagram

elements such as activities, gateways or data objects). The *Validation Clauses Generator* is a component that performs a transformation between SLA language and OCL and it will also be built with a DSL tool. This generator takes the *SLA Contract* (expressed in the SLA language) as input and generates the *Contract Validation Clauses* (expressed in OCL) as output.

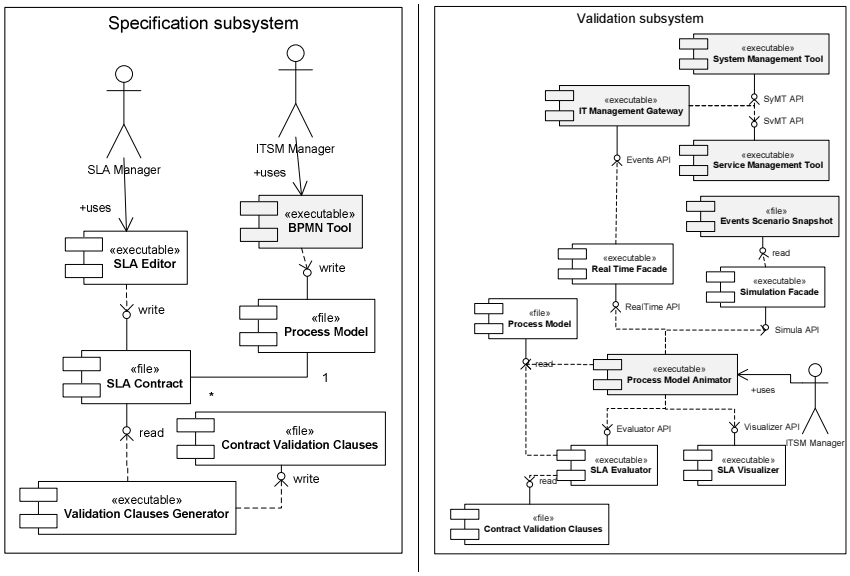

Figure 4 – SLA specification and Validation Environment (SLAVE)

In the validation stage, the *ITSM Manager* role interacts with the *Process Model Animator*. The latter allows observing the dynamics of the execution of the ITSM process on a BPMN diagram. To do so it takes the *Process Model* as input and processes its events received from one of two façades [Gamma, et al., 1995]. The *Real-Time Façade* is a front-end to an existing *IT Management Gateway* that performs a centralized processing of events coming either from *System Management Tools* (e.g. network management, software distribution) or *Service Management Tools* (concerning the execution of ITIL processes such as incident management or problem management). The *Simulation Façade* reads previously recorded *Event Scenario Snapshots*. The latter can be artificially generated (e.g. taking as basis the process model) or created by logging *IT Management Gateway* sessions and storing events in a standard format such as CIMOM [CIM, 2010], WBEM [WBEM, 2010] or a CMDB [CMDBf 2010].

The *Process Model Animator* (a third party tool such as STP[19] or ProM[20]) will have two plugins: the *SLA Evaluator* and the *SLA Visualizer*. The former is basically an OCL interpreter that is able to check the *Contract Validation Clauses* against the instantiated *Process Model*, thereby being able to deliver information regarding SLA conformance or non-conformance. The SLA Evaluator is the cornerstone of the SLAVE environment since it joins the specification and validation stages.

Finally, the *SLA Visualizer* will extend the *Process Model Animator* with graphical views (e. g. dashboard or balanced scorecard) upon the running or simulated IT service processes. For instance, they will be filtered by SLA fulfillment or non-fulfillment in a specified timeslot.

[19] http://www.eclipse.org/stp/
[20] http://prom.win.tue.nl/tools/prom/

In summary, the main expected contribution of this research work is mitigating the problems stated in 3.1.1, by proposing a model-based approach to IT services SLA specification and compliance verification. The specification part will be accomplished by proposing a SLA language for the context of ITSM. Its metamodel will be an extension of the metamodel of the adopted process modeling language. As such, it will be possible to ground SLA definition on the corresponding IT service model constructs.

Regarding the SLA compliance verification, an evaluation tool will be used, fed by the events that are captured by IT service and system management tools. The SLA evaluator will be integrated with the process model animator to allow visual information related to current levels of service being provided, namely when SLA limits are exceeded. Therefore, process execution and SLA compliance validation will be made at the level of abstraction that is understood by all the stakeholders involved in IT service specification and deployment.

3.5. Publications and Scientific Venues

Besides the empirical research validation, we intend to peer validate our research work by means of its publication in conferences and journals ranked at least with B by the Computing Research and Education Association of Australasia, (CORE)[21]. The ones that we found relevant in the software engineering area, where research community publishes about ITSM/ITIL and in the topics of this research work, are presented next:

Conferences:

- SERVICES - IEEE Congress on Services (http://www.servicescongress.org/2010/) – B
- SCC - IEEE International Conference on Services Computing (http://conferences.computer.org/scc/2010/) – A
- HICSS – Conference on System Sciences (http://www.hicss.hawaii.edu/) - A
- CISTM – Conference on Information Science Technology and Management (http://www.cistm.org/) – B
- BPM – International Conference in Business Process Management (http://www.bpm2010.org/) - A
- Quality of Information and Communications Technology (http://www.fe.up.pt/quatic2010) – Track of Quality in ICT Service Management organized by FCT/UNL.

Journals:

- Business Process Management Journal (http://www.emeraldinsight.com/products/journals/journals.htm?id=bpmj) – B
- Information and Management - The International Journal of Information Systems Applications (http://www.elsevier.com/wps/find/journaldescription.cws_home/505553/description) – A*
- Information Processing and Management (http://www.elsevier.com/wps/find/journaldescription.cws_home/244/description) – A
- Information Systems (http://www.elsevier.com/wps/find/journaldescription.cws_home/236/description) – A
- Information Systems Management (http://www.informaworld.com/smpp/title~content=t768221794) – B

[21] http://core.edu.au/index.php/

The scientific production that we intend to submit to some of the above-mentioned conferences, which is scheduled in 4.3, is the following:

- workshop paper about “Metrics definition for SLA Quality Attributes”;
- workshop paper about “Expressing Quality Attributes in the context of BPMN”;
- conference paper about “Quality Attributes elicitation in Service Level Management”;
- conference paper about “A DSL for ITSM Quality Attributes elicitation”;
- conference paper about “SLA representation in the context of BPMN“

We intend to produce 2 technical reports about work developed, regarding this research work, in a QREN project (detailed in 4.2) where we expect to conduct the empirical studies.

We also intend to submit 3 paper concerning chapters of our dissertation to journals to be selected. The topics of the intended papers are:

- “A conceptual model of SLA”;
- “SLA extension to the BPMN metamodel”
- “A SLA language for ITSM”

Until now, and in order to validate the focus of this work, three publications have already been produced:

- *Jorge Freitas, Anacleto Correia, and Fernando Brito e Abreu, "An Ontology for IT Services", In Proc. of the 13th Conference on Software Engineering and Databases (JISBD'2008) Gijon, Spain. October, 2008* – proposal of a formal ontology for IT services with preciseness granted by mean of well-formedness rules written in the OCL constraint language. The proposed ontology was instantiated to illustrate its validity in addressing realistic examples.
- *Correia, Anacleto & Brito e Abreu, Fernando, “Model-Driven Service Level Management”, 4th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2010), PhD Student Workshop, June 21-25, 2010, University of Zurich, Switzerland* – a preliminary version of the current PhD proposal was presented there. The aim of the conference was the exchange of ideas across different communities, namely in topics such as service management and networking. As feedback, we got the confirmation by academic experts present in the public presentation that in fact, this paper touched one of the open issues in the field of ITSM: the gap between SLA specification and validation at business and technical levels. However, we were advised that the instrumental part of our research, regarding the presentation of SLA compliance in process models, is probably already implemented in available solutions, since there are SLM solutions in the market quite advanced.
- *Correia, Anacleto & Brito e Abreu, Fernando, “Defining and Observing the Compliance of Service Level Agreements: A Model Driven Approach”, 7th International Conference on the Quality of Information and Communications Technology (QUATIC 2010), Faculty of Engineering, Porto University, Oporto, Portugal, 29/9- 2/10/2010* (accepted).

The 1st International Workshop on ITSM, a Thematic Track promoted under the 7th International Conference on the Quality of Information and Communications Technology (QUATIC 2010), is an attempt to establishing an academic forum for regular presentation and discussion of recent ITSM research developments. With the regular organization of this workshop, linked each year to a recognized conference in the fields of services or software engineering, we intend to surpass the lack of this kind of meeting for the academic community, since that ITSM events that take place are intended for practitioners and solutions vendors.

4. Work Plan

4.1. Methodology

This research could be generally classified as in the intersection of Information Systems (IS) and Computer Science (CS). These two knowledge areas are both concerned with the analysis, development and use of computer-based products, so there is a large area of overlap between them. The aspects of IS which are relevant to this research are the study of the processes by which systems are analyzed and designed, as well as the contributions they make to business effectiveness. From CS we adopted Software Engineering tools and techniques.

Our research will be grounded on the study of a real world project (aka empirical assessment of systems), in order to collect evidence that allow validation of this research in ITSM. Therefore, the research will be focalized in empirical assessment and evaluation, so that we can have evidence-based conclusions about our research hypotheses.

A research approach, based in the scientific method and comprising three main iterative phases, will be performed:

(1) *foundation* phase - for designing the SLA language along with the setting up of the SLA specification and validation environment;

(2) *serviceable* phase - where the environment will be instantiated with a real world case, and operational data collected;

(3) *evaluation* phase will draw conclusions after empirical assessment of research hypotheses, grounded in current and historical data. For conducting the results validation, we will follow the experimental software engineering process summarized in [Goulão and Brito e Abreu, 2007]. The process encompasses five phases: requirements definition, design planning, variables selection, data selection, data analysis, and results packaging, recognizing the validity threats. The results achieved in the aforementioned process, will be reported in the format proposed in [Jedlitschka and Pfahl, 2005].

This research approach heavily depends on the resources and data to be collected in the QREN project detailed in section 4.2, in order to perform the experiments mentioned in 3.1.4. Nevertheless, we are aware that, we are relying on a project which is not yet approved. Therefore we considered also the alternative of applying a research approach for information systems: action-research [Cunha and Figueiredo, 2002]. Due to the fact that action-research is more appropriate for a study supported by qualitative data, we should be able to overcome the contingency of lacking of quantitative data and resources needed to conduct the experiments based in scientific method.

4.2. Validation

To support this research it was established a joint research work, as part of a QREN project, between the CITI/QUASAR group (where this PhD work is being developed) and SoftFinança an industrial partner, whose main operational processes will provide an adequate background for this research work.

The mentioned QREN project is expected to study, analyze and implement a new events management system for financial self-service terminals networks (ATMs, Kiosks and POS). Events

management is concerned with faults or malfunctions related to the use or operation of terminals, either by final users or by support team members (supervisors, maintenance and replenishment agents and helpdesk). Events are caused by total or partial unavailability of terminals (due to hardware or software problems), as well as, other causes not related with the availability of the terminals (e.g. paper jam, printer's lack of ink, fake banknotes deposit).

The project's broad scope will extend the current version of a solution (Expand Care), which manages hundreds of ATMs in Portugal, to enable its simultaneous use, spanning several countries, with tens of thousands of terminals. An innovative approach to the delivery and processing of messages will take place in this project. The goal is to implement a model where the dispatching of events will result from the conjunction of rules, namely settled in service level agreements contracts. This will differ from solutions deployed in the market today, that are based in the paradigm of division of the total of terminals in groups, by one or two criteria, allocating each group to one professional helpdesk.

The new system should be able to monitor and verify the SLAs compliance considering the following features:

- Be able to capture events that start and end the assessment of SLA compliance (e.g. service provider notification, incident close down);
- Allow for different SLA monitoring methods depending on the type of provider and service (e.g. cash replenishment, peripherals' faults);
- Allow different levels of SLA monitoring (e.g. SLA per ATM, network supplier SLA);

4.3. Detailed Plan

In this section we describe with more detail the set of activities to be accomplished from the initial survey work until the presentation of the main results in the PhD dissertation.

Year 1 (ongoing until September 2010)

- ***Activities:***
  - Finish the survey on the state of the art on SLA, NFRs, DSLs, and process modeling;
  - Detail the research methodology to be adopted;
  - Verify and confirm research questions and plan the design of experiments;
  - Domain Analysis of SLAs in ATM industry, in order to derive a feature model, which will be the foundation of the DSL Design Process (language metamodel, syntax and notations). DSL evaluation;
  - Organize and promote a 1st International Workshop on ITSM-based research topics
  - Participate in the submission of a research project to the Portuguese Foundation for Science and Technology (FCT/MCTES) in case of the QREN project, mentioned in 4.2, don't get approval;
  - Disseminate results internationally and collect feedback of first findings.
- ***Publications:***
  - 1 technical report about the QREN project;
  - Workshop paper: "Metrics definition for SLA Quality Attributes";
  - Conference paper: "Quality Attributes elicitation in Service Level Management";

Year 2 (2010/2011)

- ***Activities:***
  - Conclude the SLAVE implementation;
  - Data collection process for hypotheses testing;
  - Conduct planned experiments;
  - Preliminary assessment and refinement of research questions;
  - Identify threats to the validity of results and ways to mitigate them;
  - Build and integrate SLAVE components (described in section 3.4). Start collecting field data for SLAVE validation;
  - Disseminate results and collect feedback by international peers.
- ***Publications:***
  - 1 technical report about the QREN project;
  - Workshop paper: "Expressing Quality Attributes in the context of BPMN";
  - Conference paper: "A DSL for ITSM Quality Attributes elicitation";
  - Journal paper: "A conceptual model of SLA";
  - Journal paper: "SLA extension to BPMN metamodel".

Year 3 (2011/2012)

- ***Activities:***
  - Finalize the data collection process;
  - Conclude planned experiments, data analysis and hypotheses testing;
  - Final assessment of achieved results taking into account the raised research questions;
  - Results packaging: results interpretation, validity threats discussion, inference and conclusions;
  - Organize and promote a 2nd International Workshop on ITSM-based research topics;
  - Disseminate results internationally and collect feedback;
  - Write and revise dissertation.
- ***Publications:***
  - Conference paper: "SLA representation in the context of BPMN"
  - Journal paper: "A SLA language for ITSM"
  - PhD dissertation

5. Bibliography

[Aagedal, 2001] Aagedal, J.O.: *Quality of Service Support in Development of Distributed Systems*, PhD thesis, University of Oslo, 2001.

[AToM3, 2010] 'A tool for multi-paradigm modelling', *http://atom3.cs.mcgill.ca/*, accessed on 10-01-2010.

[Balasubramanian, et al., 2006] Balasubramanian, D., Narayanan, A., vanBuskirk, C., and Karsai, G.: 'The Graph Rewriting and Transformation Language: GReAT', *Proc. of Proceedings of the Third International Workshop on Graph Based Tools (GraBaTs 2006), Electronic Communications of the EASST Volume 1*, 2006.

[Barbacci, et al., 1995] Barbacci, M., Klein, M.H., Longstaff, T.A., and Winstock., C.B.: '*Quality Attributes. Technical Report'*, Report No. CMU/SEI-95-TR-021, Software Engineering Institute, Carnegie Mellon University, 1995.

[Barzdins, et al., 2007] Barzdins, J., Zarins, A., Cerans, K., Kalnins, A., Rencis, E., Lace, L., Liepins, R., and Sprogis, A.: 'GrTP: Transformation Based Graphical Tool Building Platform', *Proc. of MoDELS`07,Workshop: Model Driven Development of Advanced User Interfaces (MDDAUI-2007), Vol 297*, 2007.

[Bass, et al., 2003] Bass, L., Clements, P., and Kazman, R.: *Software Architecture in Practice*, Addison Wesley, Second Edition edn, 2003.

[Bianco, et al., 2008] Bianco, P., Lewis, G.A., and Merson, P.: '*Service Level Agreements in Service-Oriented Architecture Environments'*, Report No. CMU/SEI-2008-TN-021, Software Engineering Institute - Carnegie Mellon University, 2008.

[Bon, et al., 2005] Bon, J.v., Pieper, M., and Veen, A.v.d. Eds.: *Foundations of IT Service Management: Based on ITIL, ITIL Version 2*, Van Haren Publishing, 2005.

[Botella, et al., 2001] Botella, P., Burgues, X., Franch, X., Huerta, M., and Salazar, G.: 'Modelling Non-Functional Requirements', *Proc. of Jornadas Ingeniera de Requisitos Aplicados (JIRA)*, Sevilla, Spain, 2001.

[BPDM, 2008] BPDM, O.: '*Business Process Definition MetaModel (BPDM), Version 1.0'*, Report, OMG, November 2008, 2008.

[BPMN, 2009] BPMN, OMG: '*Business Process Model and Notation (BPMN) '*, Report, dtc/2009-08-14, 2009.

[Chung, et al., 2000] Chung, L., Nixon, B.A., Yu, E., and Mylopoulos, J.: *Non-Functional Requirements in Software Engineering*, Boston, 2000.

[CIM, 2010] 'Common Information Model (CIM) Standards, CIM 2.24.0', DMTF-Distributed Management Task Force, Inc., *http://www.dmtf.org/standards/cim/*, accessed on 30-01-2010.

[CMDBf 2010] 'CMDBf Specification', *http://www.dmtf.org/standards/published_documents/DSP0252_1.0.0c.pdf*, 30-01-2010.

[Cooper, et al., 2003] Cooper, K., Dai, L., Deng, Y., and Dong, J.: 'Developing a Formal Design Analysis Framework.' *Software Engineering Research and Practice*, pp. 68-73, 2003.

[Coyle and Brittain, 2009] Coyle, D.M., and Brittain, K.: '*Magic Quadrant for IT Service Desk'*, Report, Gartner 16-10-2009, 2009.

[Cunha and Figueiredo, 2002] Cunha, P.R.d., and Figueiredo, A.D.d.: 'Action-research And Critical Rationalism : A Virtuous Marriage', *Proc. of ECIS 2002 - The Xth European Conference on Information Systems*, Gdansk, Poland, June 6-8, 2002.

[Cysneiros, et al., 2001] Cysneiros, L.M., Leite, J.C.S.d.P., and Neto, J.d.M.S.: 'A Framework for Integrating Non-Functional Requirements into Conceptual Models ', *Requirements Engineering*, Springer-Verlag London Limited(6), pp. 97-115, 2001.

[Czarnecki and Helsen, 2003] Czarnecki, K., and Helsen, S.: 'Classification of Model Transformation Approaches', *Proc. of OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.

[DiaGen/DiaMeta, 2010] 'The Diagram Editor Generator', http://www.unibw.de/inf2/DiaGen/, accessed on 10-01-2010.
[Drr, et al., 2003] Drr, J., Kerkow, D., Knethen, A.V., and Paecha, B.: 'Eliciting Efficiency Requirements with Use Cases.' *Proc. of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03)*, 2003.
[Erl, 2007] Erl, T.: *SOA: Principles of Service Design*, Prentice Hall, 2007.
[Firesmith, 2003] Firesmith, D.: 'Security Use Cases', *Journal of Object Technology*, 2(3), pp. 53-64, 2003.
[Frankel, 2003] Frankel, D.S.: *Model Driven Architecture Applying MDA to Enterprise Computing*, Wiley Publishing, Inc, 2003.
[Gamma, et al., 1995] Gamma, E., Helm, R., Johnson, R., and Vlissides, J.: *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.
[Gläßer, 2005] Gläßer, L.: '*Development and Operation of Application Solutions 2005/2006*', Report, Siemens Business Services, 14.09.2005, 2005.
[GMF, 2010] 'The Eclipse Graphical Modeling Framework (GMF)', http://www.eclipse.org/modeling/gmf/, accessed on 10-01-2010.
[Goulão and Brito e Abreu, 2007] Goulão, M., and Brito e Abreu, F.: 'Modeling the Experimental Software Engineering Process', *Proc. of 6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007)*, Lisboa, Portugal, 2007.
[Grundy, et al., 2008] Grundy, J., Hosking, J., Huh, J., and Na-Liu, K.: 'Marama: An Eclipse Meta-Toolset for Generating Multi-View Environments', *Proc. of International Conference on Software Engineering*, Leipzig, Germany, 2008.
[Herrmann and Paech, 2005] Herrmann, A., and Paech, B.: 'Quality Misuse', *Proc. of REFSQ - International Workshop on Requirements Engineering: Foundation for Software Quality*, 2005.
[Ho, et al., 2006] Ho, C.-W., Johnson, M.J., Williams, L., and Maximilien, E.M.: 'On Agile Performance Requirements Specification and Testing', *Proc. of the conference AGILE '06*, 2006.
[Holt, et al., 1983] Holt, A., Ramsey, R., and Grimes, J.: 'Coordinating System Technology as the Basis for a Programming Environment', *Electrical Communication*, 57(4), pp. 307-314, 1983.
[IBM, 2001] IBM: '*WSLA Language Specification, version 1.0*', Report, 2001.
[INRIA and OBEO, 2010] 'ATLAS Transformation Language', http://wiki.eclipse.org/index.php/ATL, accessed on 10-01-2010.
[ISO20000-1, 2005] ISO20000-1, International Organization for Standardization: *ISO/IEC 20000-1:2005 - Information technology -- Service management -- Part 1: Specification*, International Organization for Standardization, 2005.
[ISO20000-2, 2005] ISO20000-2, International Organization for Standardization: *ISO/IEC 20000-2:2005 - Information technology -- Service management -- Part 2: Code of practice*, International Organization for Standardization, 2005.
[ITIL3C, 2007] ITIL3C, OGC-Office of Government Commerce: *Continual Service Improvement, ITIL Version 3*, ITSMF- IT Service Management Forum, 2007.
[ITIL3D, 2007] ITIL3D, OGC-Office of Government Commerce: *Service Design, ITIL Version 3*, ITSMF- IT Service Management Forum, 2007.
[ITIL3Sm, 2007] ITIL3Sm, OGC-Office of Government Commerce: *Summary, ITIL Version 3*, ITSMF- IT Service Management Forum, 2007.
[Jedlitschka and Pfahl, 2005] Jedlitschka, A., and Pfahl, D.: 'Reporting Guidelines for Controlled Experiments in Software Engineering', *Proc. of 2005 International Symposium on Empirical Software Engineering*, Queensland, Australia, 17-18 Nov. 2005, 2005.
[JSDL, 2005] 'Job Submission Description Language (JSDL) Specification, Version 1.0', Global Grid Forum, http://www.ogf.org/documents/GFD.56.pdf, accessed on 30-01-2010.

[Kaiya and Kaijiri, 1999] Kaiya, H., and Kaijiri, K.: 'Refining Behavioral Specification for Satisfying Non-functional Requirements of Stakeholders', *EICE Transactions on Information and Systems*, E85-D, No.4(20020401), pp. 623-636, 1999.

[Kalnina and Kalnins, 2008] Kalnina, E., and Kalnins, A.: 'DSL tool development with transformations and static mappings', *Proc. of MODELS 2008 Workshops - Doctoral Symposium*, Toulouse, 29.9.2008, 2008.

[Kassab, 2009] Kassab, M.: *Non-Functional Requirements: Modeling and Assessment*, VDM Verlag, 2009.

[Kassab, et al., 2007] Kassab, M., Daneva, M., and Ormandjieva, O.: 'Scope Management of Non-Functional Requirements', *IEEE Computer Society*, The 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO '07), pp. 409-417, 2007.

[Lamsweerde, 2001] Lamsweerde, A.V.: 'Goal-Oriented Requirements Engineering: A Guided Tour', *Proc. of the Fifth IEEE International Symposium on Requirements Engineering (RE '01)*, Washington, DC, USA, 2001.

[Lamsweerde, 2004] Lamsweerde, A.V.: 'Elaborating Security Requirements by Construction of Intentional Anti-Models', *Proc. of the 26th International Conference on Software Engineering (CSE '04)*, Washington, DC, USA, 2004.

[Langlois, et al., 2007] Langlois, B., Jitia, C.-E., and Jouenne, E.: 'DSL Classification', *Proc. of OOPSLA'07 – Domain-Specific Modeling Workshop*, October 21-22, 2007, 2007.

[Laudon and Laudon, 2005] Laudon, K.C., and Laudon, J.P.: *Management Information Systems: Managing the Digital Firm*, Prentice Hall, 9th edn, 2005.

[Letier, 2001] Letier, E.: *Reasoning About Agents in Goal-Oriented Requirements Engineering.*, Ph.D. thesis, 2001.

[Lindquist, et al., 2007] Lindquist, D., Madduri, H., Paul, C.J., and Rajaraman, B.: 'IBM Service Management architecture', *IBM Systems Journal*, 46(3), pp. 423-440, 2007.

[Maps, 2010] 'IT Process Maps', http://en.it-processmaps.com/, accessed on 30-01-2010, 2010.

[Matoussi and Laleau, 2008] Matoussi, A., and Laleau, R.: '*A Survey of Non-Functional Requirements in Software Development Process'*, Report No. TR-LACL-2008-7, Departement d'Informatique Universite Paris 12, 2008.

[Mayer, et al., 1995] Mayer, R., Menzel, C., Painter, M., Perakath, B., de Witte, P., and Blinn, T.: '*Information Integration For Concurrent Engineering (IICE) - IDEF3 Process Description Capture Method Report'*, Technical ReportReport, Armstrong Laboratory, September 1995, 1995.

[Mayerl, et al., 2005] Mayerl, C., Vogel, T., and Abeck, S.: 'SOA-based Integration of Service Management Applications', *Proc. of the IEEE International Conference on Web Services*, 2005.

[MDA, 2001] MDA, OMG: '*Model Driven Architecture (MDA)'*, Report No. ormsc/2001-07-01, July 9, 2001, 2001.

[Mellor, et al., 2004] Mellor, S.J., Scott, K., Uhl, A., and Weise, D.: *MDA Distilled: Principles of Model-Driven Architecture*, Addison Wesley, 2004.

[METAclipse, 2010] 'MOLA Tool Architecture', http://mola.mii.lu.lv/tool_description.html, accessed on 10-01-2010.

[MetaEdit+, 2010] 'Domain-Specific Modeling (DSM) environment', http://www.metacase.com/, accessed on 10-01-2010.

[MOF, 2006] MOF: '*Meta Object Facility (MOF) Core Specification - Version 2.0'*, Report No. formal/06-01-01, OMG-Object Management Group, January, 2006.

[MOLA, 2010] 'MOLA - MOdel transformation LAnguage ', Institute of Mathematics and Computer Science University of Latvia, http://mola.mii.lu.lv/.

[MS, 2008] MS, M.C.-. *Microsoft Operations Framework (MOF) 4.0*, Microsoft Corporation, 2008.

[MSDSL, 2010] 'Domain-Specific Language Tools', http://msdn.microsoft.com/en-us/library/bb126235.aspx, accessed on 10-02-2010.

[Mylopoulos, et al., 1999] Mylopoulos, J., Yu, E., Nixon, B.A., and Chung, L.: *Non-Functional Requirements in Software Engineering*, Springer, (October 1, 1999) 1st edn, 1999.

[Nunes, et al., 2009] Nunes, C., Araújo, J., Amaral, V., and Silva, C.T.L.L.: 'A Domain Specific Language for the I* Framework', *Proc. of ICEIS 2009 - 11th International Conference on Enterprise Information Systems*, Milan, Italy, May 6-10, 2009, 2009.

[OASIS, 2005] 'The DCML Framework Specification', OASIS, http://www.dcml.org/, accessed on 30-01-2010.

[OCL, 2006] OCL, OMG- Object Management Group: *OCL - Object Constraint Language Version 2.0*, 2006.

[OGF, et al., 2007] OGF, Alain Andrieux, Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., and Xu, M.: '*Web Services Agreement Specification (WS-Agreement)'*, Report, Open Grid Forum, 2007.

[OptXware, 2006] OptXware: '*The Viatra-I Model Transformation Framework Pattern Language Specification'*, Report, August 26th, 2006, 2006.

[OVF, 2010] 'DMTF OVF Specification V1.0.0 (preliminary standard)', Distributed Management Task Force (DMTF), http://www.dmtf.org/standards/published_documents/DSP0243_1.0.0.pdf, accessed on 30-01-2010.

[Paschke, 2005] Paschke, A.: 'RBSLA - A Declarative Rule-based Service Level Agreement Language based on RuleML', *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, 02, pp. 308 - 314, 2005.

[Patrício, 2010] Patrício, P.M.B.: *Unificação de linguagens orientadas a objectivos: o caso do i* e o KAOS*, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2010.

[Petri, 1962] Petri, C.A.: *Kommunikation mit Automaten. Dissertation, Schriften des IIM Nr. 2,* , Universität Bonn, 1962.

[Pfeiffer and Pichler, 2008] Pfeiffer, M., and Pichler, J.: 'A Comparison of Tool Support for Textual Domain-Specific Languages', *Proc. of the 8th OOPSLA Workshop on Domain-Specific Modeling (DSM' 08)*, October 2008, 2008.

[PinkElephant, 2010] 'PinkVERIFY 3.0 Toolsets', http://www.pinkelephant.com/pinkverify/pinkverifytoolsetv3.htm, accessed on 30-01-2010.

[Röttger and Zschaler, 2006] Röttger, S., and Zschaler, S.: 'Tool support for refinement of non-functional specifications', *Softw Syst Model*, pp. 185–204, 2006.

[RuleML, 2010] 'The Rule Markup Initiative', http://ruleml.org/, accessed on 30-01-2010.

[Sallé, 2004] Sallé, M.: '*IT Service Management and IT Governance: Review, Comparative Analysis and their Impact on Utility Computing'*, Report, Hewlett-Packard Research Labs, 2004.

[Scheer, 1999] Scheer, A.: *ARIS – Business Process Modeling*, Springer Verlag, 1999.

[Silva and Martins, 2007] Silva, M.M.d., and Martins, J.S.: *IT Governance - A Gestão da Informática*, FCA, 2007.

[Sindre and Opdahl, 2005] Sindre, G., and Opdahl, L.: 'Eliciting Security Requirements with Misuse Cases', *Requirements Engineering*, 10(1), pp. 34-44, 2005.

[Skene, 2007] 'SLAng Language', http://uclslang.sourceforge.net/specifications/slang1.1.php, accessed on 30-01-2010.

[Skene, et al., 2004] Skene, J., Lamanna, D., and Emmerich, W.: 'Precise Service Level Agreements', *Proc. of the 26th International Conference on Software Engineering (ICSE'04)*, Edinburgh, Scotland, May 2004, 2004.

[SRM, 2009] SRM: '*Storage Resource Manager Interface Specification V2.2 Implementations Experience Report'*, Report No. GFD-E.154, Open Grid Forum, 2009.

[Surhone, et al., 2010] Surhone, L.M., Timpledon, M.T., and Marseken, S.F.: *Non-functional Requirement: Systems Engineering, Requirements Engineering, Requirement, Functional Requirements, Constraints*, Betascript Publishing, 2010.

[Thayer, 2003] Thayer, R.H.: '*Software Engineering Glossary'*, Report No. doi:10.1109/MS.2003.10009, IEEE Software, May/June, 2003.

[TMF, 2007] TMF, TeleManagement Forum: *Enhanced Telecom Operations Map® (eTOM) - The Business Process Framework - Version 7.1*, 2007.
[Tosic, et al., 2006] Tosic, V., Lutfiyya, H., and Tang, Y.: 'Web Service Offerings Language (WSOL) Support for Context Management of Mobile/Embedded XML Web Services', *Telecommunications, 2006. AICT-ICIW '06. International Conference on Internet and Web Applications and Services/Advanced International*, pp. 156 - 156, 2006 .
[Tratt, 2005] Tratt, L.: '*The MT model transformation language'*, Report No. TR-05-02, Department of Computer Science, King's College London, 2005.
[UMLs, 2007] UMLs, OMG-Object Management Group: *OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2*, 2007.
[Vasudevan and Tratt, 2010] Vasudevan, N., and Tratt, L.: 'Comparative Study of DSL Tools', *Proc. of The European Joint Conferences on Theory and Practice of Software (ETAPS) - Workshop on Generative Technologies*, Paphos, Cyprus, March 27, 2010, 2010.
[Verheecke, 2003] 'Web Services Management Layer (WSML)', *http://ssel.vub.ac.be/wsml/*, accessed on 10-01-2010.
[W3C, 2005] 'Web Service Modeling Ontology (WSMO)', *http://www.w3.org/Submission/WSMO/*, 30-01-2010.
[W3C, 2007] 'Web Services Policy 1.5 - Framework', *http://www.w3.org/TR/ws-policy/H*, accessed on 30-01-2010.
[W3C, 2007b] 'Web Services Policy 1.5 - Attachment', *http://www.w3.org/TR/ws-policy-attach/*, accessed on 30-01-2010.
[WBEM, 2010] 'Web-Based Enterprise Management (WBEM)', *http://www.dmtf.org/standards/wbem/*, accessed on 30-01-2010.
[Wedemeyer and Menken, 2007] Wedemeyer, M., and Menken, I.: 'The ITIL V3 Service Management Awareness Pocket Guide': (The Art of Service, 2007).
[Wieringa, et al., 1997] Wieringa, R., Dubois, E., and Huyts, S.: 'Integrating semi-formal and formal requirements'Ed.^Eds.: *Advanced Information Systems Engineering* (Springer Berlin / Heidelberg, 1997), pp. 19-32.
[Willink, 2003] Willink, E.D.: '*UMLX : A graphical transformation language for MDA'*, Report, 4 September 2003, 2003.
[Wohlin, et al., 2000] Wohlin, C., Runeson, P., Host, M., Ohlsson, M.C., Regnell, B., and Wesslén, A.: *Experimentation in Software Engineering : An Introduction*, 2000.
[Zhu, et al., 2007] Zhu, N., Grundy, J., Hosking, J., Liu, N., Cao, S., and Mehra, A.: 'Pounamu: a meta-tool for exploratory domain-specific visual language tool development', *Journal of Systems and Software 80 (2007)* pp. 1390-1407, 2007.
[Zschaler, 2004a] Zschaler, S.: 'Formal Specification of Non-functional Properties of Component-Based Software', in Zschaler, S., et al. Eds., TUDFI04-Workshop on Models for Non-functional Aspects of Component-Based Software, 2004a.
[Zschaler, 2004b] Zschaler, S.: 'Towards a Semantic Framework for Non-functional Specifications of Component-Based Systems', in Steinmetz, R., and Mauthe, A. Eds., IEEE Computer Society, EUROMICRO, 2004b.