A Controlled Natural Language Approach for Integrating Requirements and Model-Driven Engineering

David de Almeida Ferreira, Alberto Rodrigues da Silva
INESC-ID / Instituto Superior Técnico
Rua Alves Redol 9, 1000-029 Lisboa, Portugal
david.ferreira@inesc-id.pt, alberto.silva@acm.org

***Abstract*—Despite the efforts made during the last decades, Software Engineering still presents several issues concerning software products' quality. Requirements Engineering plays a important role regarding software quality, since it deals with the clear definition of the target system's scope. Moreover, Requirements Engineering is crucial to deal with change management, which is required to ensure that the final product reflects the stakeholders' expectations, namely the client and end-users business-related needs. We advocate the need to address the open issues regarding the requirements development process, namely to mitigate the drawbacks of using informal natural language, such as ambiguity and inconsistency. Moreover, we recognize the importance of automation to enhance productivity by avoiding repetitive and error-prone activities. In this paper, we propose a new socio-technical approach to overcome these software quality problems, consisting on the deep integration of Requirements Engineering with Model-Driven Engineering processes. This approach is based upon a controlled natural language for requirements specification, supporting the automatic extraction and verification of requirements models with Natural Language Processing techniques. The current results consist on the development of a Wiki-based tool prototype to validate our research ideas.**

***Keywords*-Software Requirements; Software Engineering Tools and Methods.**

I. Introduction

The software development process is rather complex. It strongly depends on several aspects, especially those related with technical, social, and organizational issues. The advances that have emerged in last decades, not only those regarding methodologies and processes, but also concerning to programming languages and development environments, have contributed with their best practices and tools for developing software more efficiently. However, we still witness a high rate of failure among IT projects [1]. Many of these problems could be avoided, or at least mitigated, if there was a greater concern about strictness during the project's early activities, specifically those regarding RE (Requirement Engineering). Without a proper scope definition, it is impossible to address most of the Project Management goals, such as budget, schedule, and quality, which are required to successfully conclude a project.

In order to overcome some of these limitations, the Information Systems Group (GSI) of INESC-ID has been promoting, since 2004, an initiative called ProjectIT [2], focused on Software Engineering's techniques and approaches. Under this initiative, we developed a project that yielded some positive results regarding the integrated vision of RE and MDE (Model-Driven Engineering) [3]. In particular, during this project we proposed two languages [3], one for each of these topics. We also developed a CASE tool [3], which was used as a prototype during our experiments to validate the proposed goals.

Although the achieved results were relevant, there are several open research issues. Regarding RE, the ProjectIT requirements specification language, ProjectIT-RSL [4], does not support aspects related to the definition of behavior and doesn't have a graphical notation. Moreover, it is currently oriented towards the domain of interactive systems [3]. Hence, the current RE component of ProjectIT initiative still has some limitations regarding expressiveness, coverage, and communication. Also, despite the growing number of proposals for RE approaches addressing different applicational areas, there is none (to our knowledge) that integrates the RE and MDE paradigms. Instead of beginning the MDE process at the design level, using as input the final requirements specifications, we advocate the early integration of automatically-extracted requirements models to streamline the process of designing the abstract solution. Therefore, the main goal of this doctoral proposal is to contribute to the advance of the state of the art regarding RE, namely to promote requirements models to first-class citizens during the RE process, and to extract these models from natural language specifications with a domain-oriented and extensible controlled natural language. Finally, regarding the assessment of this proposal, since RE is intrinsically social, collaborative, and competitive we argue the benefits of adopting a Wiki-based platform as a CSCW (Computer Supported Cooperative Work) complementary tool.

The remainder of this paper is structured as follows. Section 2 presents a reviews of related research, as well as industry standards and tools. In Section 3, we continue with a presentation of this proposal's research objectives and approach. In Section 4, we turn our attention to the current work and preliminary results of this doctoral proposal, namely its theoretical foundations and the tool's

overview. Section 5 is devoted to the presentation of the tentative work plan and to the discussion of its implications. Finally, Section 6 concludes this paper, highlighting the contributions that this doctoral proposal will bring to the scientific community, through the seamless integration of RE with the MDE paradigm.

II. State-of-the-Art

The idea of providing a complete software development workbench throughout the entire life cycle is not new, but only some recent attempts have partially succeeded. The ProjectIT initiative [2], on which this doctoral proposal relies, seeks to contribute in achieving such a platform. According to this initiative, the emphasis of the software development process should be on higher-level activities such as those related with Project Management, Requirements Engineering, and Architectural Design. The production effort, namely programming and testing activities, should be minimized and performed automatically. Within the context of this initiative, we began to research on topics related with the RE and MDE areas. With the insight gained on these research fields, we defined the ProjectIT approach [3], supported by a set of modeling languages. Within the context of this paper, we emphasize ProjectIT-RSL [4], which consists of a controlled natural language, based on linguistic patterns, for capturing textual requirements. Furthermore, we developed proof-of-concept tools, which were integrated with each other into a plugin-based platform, ProjectIT-Studio [3]. This desktop-based platform covers the entire software development life-cycle and provides guidance throughout the process, according to the software engineering best practices and methods. Regarding RE, the focus of this proposal, this platform provides ProjectIT-Studio/Requirements [5], a ProjectIT-RSL rich-text editor to address our vision of a Word-like processor that provides on-the-fly feedback while the user writes the requirements specifications (e.g., warnings, errors, auto-completion). This tool assists non-technical stakeholders during the requirements specification activity by providing context-aware feedback with Natural Language Processing (NLP) techniques, thus allowing them to validate their own contributions. However, and despite the concrete results achieved with our research, we intend to continue exploring some open issues regarding ProjectIT-RSL, namely to extend its expressiveness and endow the supporting tools with reusability mechanisms. Additionally, the endeavor of this doctoral proposal is to capture formal models from free-form textual specifications, and to provide a low-level intermediate language to support the seamless integration of RE with the MDE paradigm. Furthermore, we intend to provide a more adequate platform to the nature of the RE process, which is intrinsically social.

In terms of research projects, there are few proposals similar to ours. Also, there is (to our knowledge) no proposal that begins with the requirements development process, using natural language specifications, and ends with the application of generative techniques for producing artifacts, such as documentation and source code. The most relevant projects and approaches regarding to this thesis proposal are the following. Ambriola and Gervasi [6] propose a "lightweight formal method" approach, supported by the use of modeling and model-checking techniques to produce a formal validation of the requirements written in natural language. Their project, called CIRCE [6], uses natural language as the specification language and provides feedback to the user with a multi-perspective approach. They also use fuzzy matching domain-based parsing techniques to extract knowledge from requirements documents, which are later used to provide different views and models to analyze the information captured from the textual specifications. Although CIRCE and ProjectIT-Studio/Requirements share some similarities, there are many differences between them, namely in the architecture, concepts and algorithms used, and above all, in the adopted strategy: the main goal of CIRCE project is to provide requirements validation, and only recently it shifted its focus onto the integration with model-driven approaches; whereas our goal with requirements specification has always been to obtain a consistent requirements model that is in conformance with a predefined metamodel [4], which in turn is aligned with the model-driven techniques of the ProjectIT initiative. Attempto Controlled English (ACE) [7] uses a controlled natural language to write precise specifications that, for example, enable their translation into a representation similar to first-order logic (called Discourse Representation Structures). Although, at first sight, ACE and CIRCE appear to have similar goals to our proposal, both of them lack on a deep integration with MDE tools and techniques. In literature, there are also approaches, such as the ones presented in [8] and [9], that aim to reduce the level of imprecision in requirements by using a limited number of sentence patterns to specify requirements for a particular domain. Denger et al. [10] have also identified natural language patterns used to specify functional requirements of embedded systems, from which they developed a metamodel for requirements statements. Juristo et al. [11] tried to formalize the analysis of natural language sentences to create precise conceptual models. Additionally, we researched the NL-OOPS [12] and the LIDA [13] tools, since they both process natural language requirements to construct the corresponding object-oriented model. The Object-Process Methodology, and its supporting tool OPCAT [14], deal with system static-structural and dynamic-behavioral aspects by adopting a single unified model that can be represented through a visual formalism, which in turn can be automatically translated into a subset of natural language. Finally, there are some emergent Aspect-Oriented Requirement Engineering (AORE) approaches [15], which recently started to employ NLP techniques to perform domain analysis.

On the other hand, regarding the MDE field, there are some initiatives that attempt to address model-to-model transformations. Some MDE approaches, such as OMG's Model-Driven Architecture (MDA) [16], advocate the use of model-to-model transformations in order to automatically obtain models at different levels of abstraction. However, the means through which these transformations are implemented are typically left unspecified, as there are still few adequate tools and languages. The details of implementation regarding these model-to-model transformations are beyond the scope of this doctoral proposal. Still concerning to methodologies and standards, SysML (Systems Modeling Language) [17] is an emerging general-purpose graphical modeling language for systems engineering. It is oriented towards complex systems and covers several activities and artifact types of the software development process. SysML is defined as a UML profile and covers a broad range of systems and components, namely hardware, software, information, processes, and actors. SysML language introduces: (1) a new diagram type called requirements diagram; (2) a graphical element for representing text-based requirements; and (3) relation types for linking them to other model elements, empowering the user with traceability mechanisms [18]. This new type of diagram captures requirements hierarchy and derivation, and also allows to check if they satisfy conditions and/or verify relationships. Therefore, it provides a bridge between the typical requirement management tools and the system's models. However, this approach still suffers from an important drawback: it treats requirements as black-box elements, neglecting its natural language semantics in favor of enhanced traceability. This clearly is in contrast with our approach, which seeks to capture each requirement meaning and verify requirement specifications accordingly.

Finally, regarding RE tools, most are commercial and integrated within larger proprietary suites, presenting a strong emphasis on requirements management, documentation, traceability, and impact analysis. There are renowned commercial tools such as IBM's Rational RequisitePro, IBM's Telelogic DOORS, and Borland's Caliber Analyst, which promote the usage of natural language to specify requirements. Some of these tools already provide parsers to analyze text for keywords, to automatically identify several data types, and to recognize requirements based on the document's structure while importing external sources. However, they lack semantic analysis of the requirements specifications and the model verification features. Their natural language approach deals with requirements without extracting their meaning through introspection (i.e., they are treated as simple pieces of text with manually added attributes and annotations), according to the previously mentioned black-box approach. The validation process is limited to the analysis of user-specified metadata or the analysis of relationships established among requirements and artifacts. Moreover, due to the usual tension between language expressiveness and formal strictness, which can compromise the tool's wide adoption by non-technical stakeholders, the commercial tools' developers do not address natural language usage drawbacks, namely ambiguity, inconsistency, incompleteness, and inadequacy. However, the storage of information with this kind of problems opens the door for inconsistent, incomplete, and generally low-quality requirements. We seek to address this by using a controlled natural language, which has the advantages of natural language (such as expressive power and familiarity), and also supports the rigorous specification of requirements and the derivation of models from them.

III. Research Objectives and Approach

Historically, the requirements specification process has consisted of creating a natural language description of what the target system should do or constraints about its behavior [19]. However, this form of specification is both ambiguous and, in many cases, unverifiable because of the lack of a standard machine-executable representation [19]. Apparently, the usage of formal methods could overcome these problems. But this isn't as trivial as it seems, because we still need to take care when interpreting the natural language requirements to create a formal specification. The engineers often misinterpret natural language specifications during the design phase. Moreover, there is an additional misinterpretation level regarding the typically complex syntax and correct use of formal languages [19]. Aside from formal methods, and despite the existence of visual approaches, textual specifications are still the most suitable, fast, and preferred manner (by non-technical stakeholders) to initiate the requirements specifications of the target system.

Considering these facts, the processing of natural language requirements *per se* is of paramount importance to the automatic extraction of concepts and relations of the target system from textual specifications, and to detect early inconsistencies within the specifications, for quality assurance of the IT project. Moreover, with these automatically-extracted models, one can infer or verify more formal properties of the target system, by using inference or formal methods' techniques, such as model-checking or theorem proving.

Thus, the main goals of this doctoral proposal are related with the seamless integration of the RE process with the MDE paradigm through a controlled natural language approach (illustrated in Figure 1), namely: (1) to extend the language ProjectIT-RSL [4] to support the extraction of richer models of the target system through the processing of textual requirements; (2) to define a low-level declarative language for specifying requirements models, called RSL-IL, which facilitates the transformation of these models to other notations, such as graphical or textual representations, or even to other MDE-related languages; (3) to define transformations between RSL-IL models to support the successive refinements of requirements models through

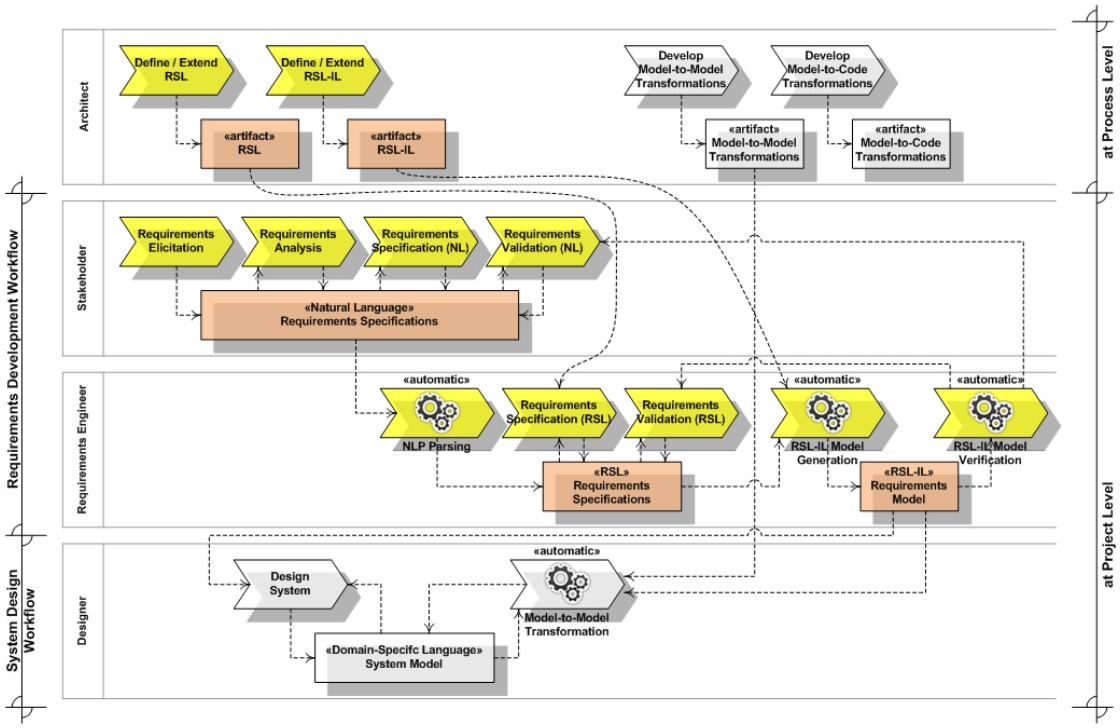

Figure 1. Requirements Development Workflow.

MDE techniques; (4) to define transformations between RSL-IL and the languages used by the generative processes of ProjectIT initiative; and (5) to define and formalize the development process according to this innovative approach. To support this approach, and because ProjectIT promotes productivity through automation, we decided to develop a Wiki-based CSCW tool. The shift from a desktop to a web application platform seeks to overcome ProjectIT-Studio's limitations, namely those regarding cooperation and communication among virtual teams. These tools are complementary: the Wiki-based tool is oriented towards business stakeholders and deals with the tension between collaborative and competitive forces of RE's nature; whereas the ProjectIT-Studio's main purpose is to support technical stakeholders in productivity-intensive tasks. To achieve the previous goals we intend to follow an iterative and incremental approach consisting of the following deliverables and milestones. Initially, we are going to address the theoretical aspects of our proposal, namely to define the ProjectIT-RSL extensions and to design the RSL-IL language. The next step is to implement the RSL-IL interpreter for the automatic verification of models and generation of run-time artifacts, such as documentation and diagrams, or even prototypes of the system being developed. To evaluate our research ideas and to complement the current tool support we are developing a Wiki-based platform. Also, the benefits of this work will be demonstrated through practical case studies. Finally, we intend to promote and support the evaluation of this proposal's results by third-party entities, namely through academic and industrial case studies (with computer science students and software houses, respectively).

IV. Current Work and Preliminary Results

Until now we have been thoroughly researching related work regarding requirements specification languages. We have also been improving the ProjectIT-Studio/Requirements parsing mechanisms with NLP techniques. Additionally, we started the development of the Wiki-based supporting tool prototype. To incorporate innovative ideas into this prototype, we have been researching best practices and common features of CSCW platforms.

Regarding the case study, we have been establishing partnerships and conducting some experiments within the educational and e-learning areas with some Portuguese schools, namely we are performing the maintenance of the EscolaNaNet project [20], which aim to support and streamline schools' business processes. The provided scenario is challenging and entails a proper context to conduct an assessment of this work, since it has a considerable size and complexity. EscolaNaNet's scope covers several business entities, with rich relations between them, and it also requires the specification of some rather complex business processes.

V. Work Plan and Implications

This doctoral proposal can be seen according to several points-of-view. Initially, it will address concep-

tual/theoretical aspects related with the definition of suitable ProjectIT-RSL extensions and the design of a formal notation for the RSL-IL intermediate language. In parallel, we are going to address architectural/applicational aspects regarding the development of a Wiki-based tool, endowed with NLP techniques and with a RSL-IL interpreter. Finally, and according to a pragmatic/practical perspective, we are going to evaluate our achievements by conducting a pilot experiment.

To succeed, the controlled natural language approach provided by ProjectIT-RSL needs to cover not only static aspects of the target system (e.g., domain model) but also dynamic aspects related with functional and non-functional requirements. These facets are crucial to specify aspects regarding complex interaction patterns of actors with the target system. Hence, we are going to define ProjectIT-RSL extensions to capture more complex scenarios of the target system, namely behavioral patterns according to an operational perspective. Still, the non-functional requirements extensions, such as the ones addressed by GORE (Goal-Oriented Requirement Engineering) approaches [21], are going to be postponed. Although these issues are going to be kept in mind from the beginning, due to their importance regarding to architectural design, we are not going to address their impact at the functional level.

The extension of ProjectIT-RSL will cover two main topics: behavior and reusability. The former is oriented towards the capture of behavior, to address the specification of functional requirements in a procedural manner: it should be similar to pseudo-code. The purpose is to support the specification of more complex actions, namely beyond simple create, read, update and delete (CRUD) operations of business entities. This extension should be independent of the target programming languages used by the ProjectIT's generative techniques to produce source code. The other extension, regarding the reuse of ProjectIT-RSL specifications, which is aligned with the best-practices proclaimed by ProjectIT initiative, comprehends two kinds of templates. The first one focuses on reusing knowledge captured from previous projects (*applicational templates*), thus increasing the productivity by following a develop-by-reuse approach. Alternatively, the other follows a develop-for-reuse approach (the most common), whose main purpose is to define templates of requirements specifications already factorized (*architectural templates*), ensuring that these artifacts are compliant with the best-practices of requirements specification's writing techniques from the beginning.

Concerning the intermediate language RSL-IL, we are going to research similar approaches from other fields of expertise, namely we will conduct a thorough state of the art survey to identify their strengths and weaknesses. With this insight we will follow a formal definition process to design, a new low-level and declarative language to express the domain model and constraints on the target system's behavior. This language must be formally specified and should be monotonic to support an evolutionary refinement of the requirements models. The evaluation of the axiomatic properties and expressiveness of the RSL-IL is of paramount importance. The former provides the foundations for inference and model consistency verification; whereas the latter establishes the tradeoffs required for its usability in practical scenarios, namely models' verification and model-to-model transformations. Although RSL-IL is intended to be sufficiently generic, in order to quickly tackle the problem at hand (independently of the target system's type), and to quickly get concrete results, this language will be initially designed for a particular application domain. It will focus on CMS (Content Management System) or LMS (Learning Management System) applicational areas, since they are related with the scenarios we choose for conducting the assessment of our research ideas through case studies.

Another important item of this work plan concerns the design and implementation of the respective supporting tool prototype. The envisioned Wiki-based CSCW tool must provide a repository for the abstract models produced, enhanced with a traceability and reuse mechanisms to ensure consistency and refactoring of models during the refinement process. Moreover, it will incorporate specific features to assist the users throughout the two main RE workflows [22]: requirements development, and requirements management. The core asset of this application will be the tool's feedback provided by the parsing tools and the artifacts automatically generated through MDE techniques. The NLP parsing algorithms are responsible for capturing the preliminary sketches from the requirements specifications written in natural language. The output is a set of requirements models translated to RSL-IL, whose purpose is to support their formal specification. The RSL-IL interpreter, also provided by the supporting tool, will be in charge of verifying the models' consistency. After these models have been generated, the user can directly trigger the MDE generative process of ProjectIT.

Finally, the assessment of this work is going to be performed in two iterations. The first one will provide a coarse-grained validation of both the language and the supporting tool within an academic scenario. Afterwards, with the incorporation of the amendments provided by feedback from the first experiment, we will conduct a second assessment within an industrial and more challenging context, to verify the pragmatic feasibility of this approach.

VI. Conclusion

This paper presents a doctoral proposal for addressing the open issue of integrating the complementary visions of RE and MDE. The main motivation for this proposal is to adopt a pragmatic attitude regarding the high overall project failure rate, due to early mistakes such as inadequate, inconsistent, incomplete, or ambiguous requirements.

An important innovation of this proposal is the focus on requirements models as first-class citizens, namely we argue the benefits of automatically verifying requirements models, which were extracted from textual specifications written in natural language, through NLP techniques. This approach supports clear and precise communication through machine-executable (but still human-understandable) specifications to surpass one of the greatest challenges of RE: to minimize the gap between the stakeholder's vision and the model captured by the requirements engineer. The main goals of this proposal are: (1) to design and evaluate a textual and extensible requirements specification language, based on a controlled natural language approach; (2) to enhance the ProjectIT approach regarding RE-related open issues; and (3) to design an intermediate language, called RSL-IL, to support the seamless integration of RE with MDE within the context of the ProjectIT initiative.

The attainment of this doctoral proposal's goals will certainly contribute to the advance of the state of the art regarding requirements specification, semantic verification of models, and requirements-to-model transformations. The main expected results of this research are: (1) an enhanced version of ProjectIT-RSL language, enriched with extensibility and reusability mechanisms; (2) an intermediate language for requirements model representation; (3) a Wiki-based tool that provides an interactive and collaborative environment to support the assessment of our research ideas; and (4) a case study, focusing both academic and industrial contexts. The ultimate goal is to delegate on final users and business people the responsibility of directly specifying, or at least validating, what they need or want, because the overall quality or value of information is judged by themselves.

REFERENCES

[1] K. Emam and A. Koru, "A Replicated Survey of IT Software Project Failures," *IEEE Software*, vol. 25, no. 5, pp. 84–90, September 2008.

[2] A. Silva, "O Programa de Investigação ProjectIT (whitepaper)," October 2004.

[3] A. Silva, J. Saraiva, D. Ferreira, R. Silva, and C. Videira, "Integration of RE and MDE Paradigms: The ProjectIT Approach and Tools," *IET Software Journal*, vol. 1, no. 6, pp. 217–314, December 2007.

[4] C. Videira, D. Ferreira, and A. Silva, "A Linguistic Patterns Approach for Requirements Specification," in *Pro. of the 32nd EUROMICRO Conf. on Soft. Eng. and Advanced Applications*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 302–309, ISBN: 0-7695-2594-6.

[5] D. Ferreira and A. Silva, "A requirements specification case study with ProjectIT-studio/requirements," in *SAC '08: Proc. of the 2008 ACM symposium on Applied computing*. New York, NY, USA: ACM, March 2008, pp. 656–657.

[6] V. Ambriola and V. Gervasi, "On the Systematic Analysis of Natural Language Requirements with CIRCE," *Automated Software Eng.*, vol. 13, no. 1, pp. 107–167, January 2006.

[7] N. Fuchs, U. Schwertel, and R. Schwitter, "Attempto Controlled English - Not Just Another Logic Specification Language," in *Logic-Based Program Synthesis and Transformation*, P. Flener, Ed., Eighth International Workshop LOPSTR'98. Manchester, UK: Springer, June 1999, pp. 1–20.

[8] C. Achour, "Guiding Scenario Authoring," in *Proc. 8th European-Japanese Conference on Information Modelling and Knowledge Bases*. IOS Press, May 1998, pp. 152–171.

[9] C. Rolland and C. Proix, *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science. Springer, April 2006, vol. 593/1992, ch. A natural language approach for Requirements Engineering, pp. 257–277.

[10] C. Denger, D. Berry, and E. Kamsties, "Higher quality requirements specifications through natural language patterns," in *Software: Science, Technology and Engineering, 2003. SwSTE '03. Proceedings. IEEE International Conference on*, November 2003, pp. 80–90.

[11] N. Juristo, J. Morant, and A. Moreno, "A formal approach for generating OO specifications from natural language," *Journal of Sys. and Soft.*, vol. 48, no. 2, pp. 139–153, October 1999.

[12] L. Mich and R. Garigliano, "The NL-OOPS Project: OO Modeling using the NLPS LOLITA," in *Proc. 4th Int. Conf. Applications of Natural Language to Information Systems*. Springer, November 1999, pp. 215–218.

[13] S. Overmyer, B. Lavoie, and O. Rambow, "Conceptual modeling through linguistic analysis using LIDA," in *Proc. 23rd Int. Conf. Software Eng. (ICSE'01)*, May 2001, pp. 401–410.

[14] OPCAT, "What is OPM?" Retrieved Monday 8th June, 2009 from http://www.opcat.com/products_opm.htm.

[15] A. Rashid, "Aspect-Oriented Requirements Engineering: An Introduction," *IEEE International Conference on Requirements Engineering*, vol. 0, pp. 306–309, September 2008.

[16] OMG, "OMG Model Driven Architecture," Retrieved Thursday 23rd April, 2009 from http://www.omg.org/mda/.

[17] ——, "OMG Systems Modeling Language," Retrieved Thursday 23rd April, 2009 from http://www.omgsysml.org/.

[18] L. Balmelli, D. Brown, M. Cantor, and M. Mott, "Model-driven systems development," *IBM Systems Journal*, 2006.

[19] H. Foster, A. Krolnik, and D. Lacey, *Assertion-based Design*. Springer, 2004, ch. 8 - Specifying Correct Behavior.

[20] A. Silva, L. Estevens, G. Bôrrega, and R. Azevedo, "eEscola: Sistema de Gestão de Ensino à Escala Nacional," *Ingenium*, pp. 64–72, November 2005.

[21] A. van Lamsweerde, "Goal-Driven Requirements Engineering: the KAOS Approach," Retrieved Thursday 23rd April, 2009 from http://www.info.ucl.ac.be/~avl/ReqEng.html.

[22] K. Wiegers, *Software Requirements*, 2nd ed. Microsoft Press, March 2003, ISBN: 978-0735618794.