Requirements Authoring and Verification for SMEs' Information Systems Engineering

Nawel Amokrane, Vincent Chapurlat, Anne-Lise Courbis, Thomas Lambolais, Mossine Rahhou

Requirements Authoring and Verification for SMEs' Information Systems Engineering

N. Amokrane[*,]. V. Chapurlat[*], A.L. Courbis[*], T. Lambolais[*], M. Rahhou[**]**

** Laboratoire de Génie Informatique et d'Ingénierie de Production - LGI2P - site de l'école des mines d'Alès, Parc Scientifique Georges Besse, F30035 Nîmes cedex 5, France (e-mail: surname.name@mines-ales.fr)*
*** RESULIS, 1 Rue de la Bergerie, 30100 Alès (e-mail: surname.name@resulis.com)*

Abstract: Requirements engineering is a major step in any project to enhance its chances to succeed. For many reasons, this step is particularly crucial when it is question of (re)engineering information systems to support the business activities of Small and Medium size Enterprises (SMEs). In this case, we state that requirements are highly related to the definition of the enterprise model. This article presents an approach for requirements authoring and verification that promotes the autonomy of domain experts in the case of SMEs.

Keywords: Information Systems Engineering, SME, Enterprise Modeling, Requirements Engineering, Verification.

1. INTRODUCTION

Information systems (IS) can be seen as the drive belt of information not only inside the enterprise, between decision and operational systems proving added value of the enterprise, but also within its environment that includes its customers, suppliers and other external partners. The IS should be an asset on which the enterprise can rely to maintain its survival in a changing economic environment. It is particularly crucial for Small and Medium size Enterprises (SMEs) that are often seeking, opportunistically and constrained by moving markets, to set up quickly and efficiently collaborations with other enterprises to gain market share. This is why they need to carry a special effort in the (re)engineering and development of their organization, processes, skills and above all of their Information System. Every enterprise can indeed adopt new practices relying on the reactivity and flexibility offered by IS through the use of new technologies (internet, cloud...), new communication and collaborative management tools. Many concerns have to be addressed at early stages of the IS development project to enhance its chances to succeed. According to a study done by The Standish Group (2009), 44% of software development projects have exceeded the project's schedule and cost and did not provide the expected functionalities mainly because of poor requirements engineering activities. It is now widely approved that Requirements Engineering (RE) constitutes the earliest, most crucial phase in the development of an Information System. However this activity requests some competencies, skills and time to be operated. We believe that SMEs should have the possibility to efficiently perform RE activities. This article presents a requirements authoring and verification tooled approach compliant with Enterprise Modeling (EM) reference models that offers means allowing SMEs' stakeholders (i) to model various aspects of their organization and (ii) to define autonomously their requirements regarding the (re)engineering of the IS. The proposed modeling languages and means are also used as a basis for analyses and verification in order to enhance the quality of stakeholder requirements. A set of rules and constraints are defined to avoid some inherent problems related to the requirements authoring phase such as inconsistency, duplication and imprecision. Such defects may potentially cause significant delays or severe quality issues in the project life-cycle.

2. GLOBAL MODELING APPROACH AND APPLICATION CASE

Business process models serve traditionally as the main source of requirements, according to Vom Brocke & Rosemann (2010). But still, this functional view of the enterprise is not sufficient to grasp all IS requirements. The actors in charge of the tasks have to be known along with their role and position in the enterprise or its environment. The information used as inputs and outputs of the tasks have to be traced and the resources needed for the achievement of the tasks are relevant information too. For this, Enterprise Modeling (EM) approaches allow to represent, understand and engineer the structure and behavior of the enterprise providing various modeling languages, frameworks and reference models. Therefor we consider that classical requirement models should be linked to enterprise models as the latter represent the concerns that have to be addressed when building an IS. It is important to notice that, by hypothesis here, the SME procedures and organisation are supposed known and stable. So, we do not seek to evaluate or optimize its operations and performance.

Moreover, the requested stakeholders' autonomy needs providing simple modeling languages (abstract and concrete syntax), intuitive constructs (patterns and reference models), operational modeling and verification process (questionnaires, workflow…) and as possible automated tools supporting each of these activities. In this sense Amokrane et al. (2014) propose a new vision about modeling constructs inspired by the ISO19440 standard ISO/DIS (2004). Covering the classical functional, informational, organizational, and resources views to which were integrated: the external context of the enterprise and its interactions with external partners and the requirements of all internal and external stakeholders.

In order to illustrate the applicability of the authoring and verification approach, we will use a case-study: DMF is a French SME that works with brands and provides them with field actions: sales event and merchandizing in supermarkets and major retailers throughout the French territory. DMF wants to share information with its partners offering them the possibility to interact with its own information system. The clients have to be able for instance to place orders and follow the achievement of the service on line. For the sake of demonstration, we focus on an important added-value process consisting in the processing and the achievement of a service order. We only consider roles that intervene in this process.

3. REQUIREMENTS AUTHORING AND VERIFICATION

Requirements Engineering (RE) is the branch of Systems Engineering (SE) (cf. BKCASE Editorial Board (2014)) concerned with the elicitation, documentation and assessment of: (i) Stakeholders' Requirements that represent the wishes, expectations and constraints regarding the new system and reflecting the problem world; (ii) System Requirements that represent the commitment of the design team to meet as much as possible Stakeholders' Requirements when developing here a (set of) software application(s) to support the SME IS. At all events, RE aims at enhancing the quality of the initial stakeholders' requirements so that they can be transferred in confidence to the design team in a well-written, verified and validated requirements repository. So, a systematic RE process provided with modeling and verification means have to be applied. In our case: Stakeholders' Requirements are reflected by the different views (functional, resources, organisation and information) of the enterprise model where stakeholders express their current situation (e.g. current activities description), their expectations (e.g. being able to perform new activities with data and resources, for a provided reason and taking into account constraints and existing applications) and dissatisfactions (e.g. doing something differently that currently done). System Requirements result from the set of Stakeholder's requirements defied with collaboration with the design team to decide what has to be kept changed or created.

3.1 Requirements authoring

In the case of SMEs, gaps may appear between the Stakeholders' and System Requirements, due to the mitigated quality of requirements that are gathered informally by developers, as pointed in Cao & Ramesh (2008) and Nikula et al. (2000). This is mainly because most of SMEs cannot afford the assistance of works owner support and their stakeholders do not have the skills to use requirements elicitation tools. On account of that, stakeholders' requirements are usually written in Natural Language (NL) which is most of the time unconstrained (cf. Mavin et al. (2009)). This is inherently imprecise and informal thus hard to be automatically checked or validated by human experts without ambiguities. Other notations such as visual models have then been used to increase precision, like in: the goal oriented requirement language KAOS proposed by Darimont et al. (1998), scenario oriented language UCM (Use Case Maps) defined in ITU–T (2008) and URML (Unified Requirements Modeling language) proposed by Schneider et al. (2012) which is a high-level quality and functional system requirements language. These languages are however meant to be used by experts (works owner support analysts, requirement engineers, designers…). They use artefacts that require modeling competencies and so cannot be handled by SMEs' stakeholders without prior training. This goes against the autonomy of SMEs stakeholders. Controlled Natural Language (CNL) was introduced to encounter the informal nature of NL and as an easy notation to learn (cf. Williams et al. (2014)). CNLs, like in Hull et al. (2005), OMG (2008) and Mavin et al. (2009) offer engineered subsets of Natural Language also named boilerplates. Boilerplates represent simple sentence patterns, for which grammar and vocabulary have been restricted in a systematic way in order to reduce the ambiguity and complexity of full NLs (see Schwitter (2010)). Some are machine oriented and enable automatic processing like Attempto Controlled English (ACE) Fuchs et al. (2006) and RuleCNL Njonko et al. (2014). Mavin et al. (2009) for instance, propose generic templates to express event-driven, state-driven requirements or unwanted behaviors.

CNLs using requirement boilerplates show significant improvement of requirements quality (cf. Mavin et al. (2009)). Nonetheless, they are often used for general purposes and are system and solution oriented (e.g. The <system name> shall <system response>). They are most of the time used for non-functional requirements. We advocate that in the case of the development of an Information System, boilerplates can be further specialized. So we defined a set of boilerplates to harvest information related to the enterprise particular model. They are composed of fixed syntax elements and fill in the gaps attributes (between '<', '>') The boilerplates are contextualized so that the attributes represent language constructs. For example "<Enterprise> works in (<Domain>)+" captures the constructs *enterprise* and *domain*. In some cases the language construct that we want to capture is represented by a set of words. A *task* for instance is captured by "<verb> <object>". We want our approach of requirements authoring to be close to what is done in real interviews while offering autonomy to SMEs' stakeholders. Therefore, rather than listing their needs and constraints regarding the new system in an informal requirements documents using full NL, stakeholders progressively build the different views of the enterprise model answering a series of questions using the proposed set of boilerplates. To begin

with, the questions are addressed to decision makers (managers or heads of department). They first concern the context of the enterprise i.e. its business and partners. They also concern the structural organization and the resources and the distribution of the responsibilities. Next, the questions are intended for enterprise members to represent their roles and roles on which they have good knowledge. They explain these roles in term of tasks that are achieved and the interactions with other enterprise members or external partners. Reasoning about daily practices and the way tasks are carried is more intuitive for SME stakeholders. Indeed they usually do not have a full, common image of business processes in which they are involved especially when these processes are cross-cutting different departments. In the same way, business stakeholders who may have this knowledge, i.e. decision makers, do not always know the detailed tasks performed by other stakeholders according to their roles. The requirements authoring process consists of the steps described in the following. Each step consists of a questionnaire and for each question the boilerplate of the answer is given where this notation: ? = optional, * = any number, + = at least once, is used to show the cardinalities of parts of the boilerplate and the | symbol shows options in fixed syntax elements.

Step 1: Framing by defining enterprise context, business domains and partners interactions

Business domains and environment of the enterprise are captured. The scope of the enterprise part to be modeled is defined and limited by domains; a *domain* is a functional area representing a business of the enterprise. For each domain, relevant inputs and outputs that are exchanged with *external partners* are defined. The exchanges are later detailed in the functional view in step 4. For the example, we focus on one domain "direct marketing" in which DMF provides its main services: sales events and merchandizing.

In which business domains do you work?
<Enterprise> works in (<Domain>)+
DMF works in direct marketing.

Who are the external partners with whom the enterprise has exchanges?
(<Enterprise> (clients | suppliers | partners) are (<ExternalPartner>)+)+
DMF clients are Brands.
DMF suppliers are logisticians.
DMF partners are: independent contractors, supermarkets.

What do the partners provide you?
(<ExternalPartner> provide <ExternalEnterpriseInputs>)*
Logisticians provide event material.
Brands provide service order.

What products or services does the enterprise provide?
(<Enterprise> provide services (<Service>)+ to (<ExternalPartner>)+)+
DMF provides services sales event, merchandizing to brands.

Step 2: Organization and resources views definition

This step is concerned with the definition of the structure and resources of the enterprise. It provides a first look of the distribution of the responsibilities among the human resources according to their *workstation* (position) in the different *organization cells* (departments…). The main *resources* (software, equipment, machinery, robots…) necessary for the achievement of tasks in each workstation are also listed.

Is the enterprise organized in different structures? What are they?
If an organization structure does not have a name, you can give its main function.
<Enterprise> is structured into <quantity> departments (<OrganisationCell>)+
DMF is structured into 2 departments: Administrative and Financial Department, Operational General Management.

((<organisationCell>)+ reports to <organisationCell>)*
Payroll department, accounting department, IT department reports to Administrative and Financial Department.
Sales event departments, Merchandizing department, Regional management, Sales force, BackOffice reports to Operational General Management.

What are the workstations/positions that constitute the organization structures?
If you do not have a name for the position, think about its main function, or replace it with the name of the person that occupies the position.

(<OrganisationCell> is composed of (<quantity> <Workstation)+)+
Merchandizing department is composed of head of department, 7 account managers.
Regional management is composed of 3 Area managers.
BackOffice is composed of 5 backOffice operators

Step 3: Function and Information views definition

According to each business domain, single roles are defined where a *role* is the function that an enterprise member plays while intervening in the enterprise activities. Indeed, in SMEs the enterprise members may plays different distinguish roles. External partners are also considered: they can request or interact with the enterprise's IS. Therefore, their roles are defined to precise what the enterprise offers and allows them to do. Here, the focus is on what and how the work is done while playing a single role, without having to picture the whole execution of the business processes. The role is defined in terms of tasks. Each *task* is delineated by a set of *actions* that can be triggered by *notifications* (receiving some kind of enterprise object or being notified by a state change). Informational *Input* and *output* are defined and more precision is given for the used resources. The *business rules* that have to be respected during the achievement of the task are also specified, linking the rule to its application. In the example, the roles that where considered are: client (played by the external partner "Brands"), account manager, area manager and independent contractor. Because of lack of space only some tasks are presented.

Role: Client
List all the tasks that you achieve:
(<verb> <object> (<preposition> <noun>)*)+
Oder service, Follow achievement of service

Task: Order service
When do you have to achieve this task? When I need it

Can you detail the steps of the task?
{(<verb> <object> (<preposition> <noun>)*)+}
{Enter order information, Register order}

What information/file/product/service is produced during this task?
<EnterpriseObject> ((<information>*))?
Service order (begin date, end date, hours, supermarket, number of animators)

What is done with it?
(<EnterpriseObject> (is sent to|…) (<workstation>
|<companyMemeber> | <ExternalPartner>))*
Service order is sent to DMF

What resources do you use (software, equipment, machinery…)?
(Using <resource>+)*
Using electronic mail

Role: Account manager
List all the tasks that you achieve:
(<verb> <object> (<preposition> <noun>)*)+
Validate order, configure operation, order event material.

Task: Validate order
When do you have to achieve this task? *
((After receiving|…) <EnterpriseObject> from (<workstation>
|<companyMemeber> | <ExternalPartner>))*
After receiving Service order from Brands

What information/file/product is used or transformed during this task?
(<EnterpriseObject> ((<information>*))?)*
Service order (begin date, end date, hours, supermarket, number of animators)

Can you detail the steps of the task?
{(<verb> <object> (<preposition> <noun>)*)+}
{check entered service orders,
(validate order | cancel order)}

What information/file/product/service is produced during this task?
(<EnterpriseObject> ((<information>*))?)*
Service file, Invoice elements

What is done with it?
(<EnterpriseObject> (is sent to|…) (<workstation>
|<companyMemeber> | <ExternalPartner>))*
Service order is sent to Area manager

Are there any business rules that have to be followed?
If the order is ok then validate order else cancel order

3.2 Requirements verification

Verifying requirements consists of ensuring that each requirement is identifiable, useful, concise, complete, unambiguous, simple, communicable, single, justifiable and traceable. Also, with verification experts aim to have a clear, non-redundant, complete, comprehensive, robust, homogeneous and consistent set of requirements. Validating requirements determines their relevance according to the real world and whether a satisfying solution can be built and tested. In this article, we only address requirements verification and provide mechanisms to avoid and detect some of the inherent problems related to the requirements authoring phase, namely: complexity, ambiguity, inconsistency, duplication, omission, and imprecision.

The use of CNL sentence templates addresses requirement formulation problems like ambiguity, duplication or wordiness. It is one way of doing verification through guided modeling as proposed in Chapurlat (2007). Mavin et al. (2009) performed an empirical study that shows that CNLs have a number of advantages over the use of unconstrained NL as they significantly reduce NL requirements problems. Some experts, such as Annervaz et al. (2013), advocate the use of domain models for the verification of textual requirements. These methods usually address full NL requirements and have available domain models like ontologies. This enhances the semantic precision of requirements. For now, as our approach is not specific to a certain domain, it would be hard to use such techniques. Other experts translate NL requirements to more formal models like in Aceituna et al. (2014), and perform verification on these models. Yet the core feature of their work is the transformation itself. Transformations to languages supporting formal logics aim to verify the satisfaction of requirements upon system models, like in Yan et al. (2014), but they mostly concern non-functional requirements over technical systems, like embedded systems. These transformations to formal logics are difficult when the requirements are of high level and about IS concerns but are a promising trail for requirements validation. Consistency, completeness, non-ambiguity are considered in classical RE: they should also be addressed when it comes to IS requirements.

Stakeholders' Requirements are written by SME members who are not experts in requirement definition. They tend to give imprecise, incomplete and conflicting requirements and this leads to ambiguity and inconsistency. To check the requirements contents and to detect these problems, we perform verification over textual IS requirements models, by first providing guided requirements authoring with boilerplates. We defined a formal definition of their syntax (the language's grammar), which supports the users in the process of entering syntactically correct inputs. We use Natural Language Processing (NLP) techniques as well for part of speech tagging to check whether the grammatical formulation of the requirements corresponds to the boilerplates. Nonetheless the well-formedness of each requirement boilerplate does not guaranty the overall consistency and the absence of omission, imprecision and duplication. To check the existence of such defaults we defined a set of verification rules (technically defined into model queries and constraints) where the re-

quirements given by one stakeholder or different stakeholders are crossed within the same view of the enterprise model or inter-views. Furthermore expert reviews are used. Decision makers of the SME arbitrate over conflicting situations, where SME's stakeholders may describe differently the same tasks. Throughout these mechanisms we address six requirements problems:

Complexity: it is considered at two levels: (i) inherent complexity of enterprise systems due to their sociotechnical, structural and behavioral characteristics. This is managed by following EM principles like considering the enterprise views separately and by guiding the users to gradually provide more details. For example, the roles that stakeholders play are captured, then the tasks regarding each role are listed independently of any behavior, finally each task is detailed to show behavioral interactions among stakeholders. (ii) Compound and interrelated statements. This is handled by the use of CNL boilerplates: several sub-clauses are, for example, not allowed and interrelated statements are manageable and no longer an issue as every attribute refers to a construct in the language abstract syntax where relationships among construct are already defined.

Ambiguity: information that have different interpretations. Ambiguity is reduced thanks to the alignment between the attributes in boilerplates definition and the constructs of the enterprise model (e.g. "In <OrganisationCell1>, <EnterpriseMember> occupies the position <workstation>"). Also every step in the requirement authoring process is used to harvest precise concepts of the enterprise model. We also use expert reviews and restitutions to help check for any ambiguities.

Inconsistency: contradictory information, mutual exclusion between information. There are some universal inconsistencies related to the nature of the information. We have defined some constraints to avoid them e.g. the used resources must not exceed the resource quantity, the duration of an enterprise processor (business process, task...) must be greater than or equal to its sub-processors, an enterprise member cannot be subordinate to himself. Other inconsistencies related to the definition of an information system e.g. an enterprise member should have access to information of all enterprise objects used in the tasks that he performs. Also inconsistencies may exist between different enterprise views e.g. there exists at least one notification concerning external enterprise inputs and outputs related to each external partner. Some of these inconsistencies may seem obvious but they should be highlighted because they can simply be mistakes or show other defaults like omission, duplication or imprecision. We also alleviate inconsistencies with the use of arbitration among different stakeholders' statements and provide restitutions to give stakeholders the possibility to check whether the provided information (collected from the questionnaires) is really what they meant to say.

Arbitration

As the view of each stakeholder is considered, decision makers have to intervene to resolve the conflicts discovered after verification or that may appear after comparing different stakeholder's descriptions of the same roles. For instance stakeholders can describe in different ways the achievement of the same tasks. In that case, decision makers choose the appropriate way. Then the different descriptions of a role are gathered in one synthesis. In addition to increasing the level of consistency of the enterprise model, arbitration allows the practices among the enterprise to be unified.

Restitution

To offer global views of the enterprise and favor expert reviews, visual restitutions are provided from the gathered textual IS requirements. The different interactions with external partners are depicted in a context diagram and the organizational structure of the enterprise and the allocated resources are presented in organigrams. Business processes are derived in a bottom-up way from the local role-based descriptions of the tasks achieved by enterprise members or external stakeholders. This is assured by means of a mechanism that uses notifications to trace the workflow throughout the enterprise structures. Such restitutions provide awareness among enterprise members of the impact of the tasks they achieve and the use of their productions in the overall business processes of the enterprise.

Duplication: information that is repeated or restated in other words. Repeated information is identified (when it is unwanted) using NLP techniques to compare the words' roots, a task for example should be mentioned only once in the same function. Restated information can be hard to detect, as we cannot know that the different formulations refer to the same thing. Resources, partners, workstations and other constructs must have unique names. Checking consistency among enterprise views or among stakeholders' statements is a way to prevent duplication. A consistency check may for example reveal that some listed external partner does not intervene in any role or event and this could be caused by the reformulation of the partner's name.

Omission: omitted information by objective or subjective oversight. It is not uncommon for stakeholders to forget mentioning some aspects of their enterprise's structure and functionality. We seek to avoid omission by considering all views of an enterprise: organization, resources, information and function views. Asking different stakeholders about the same information and at different views helps detect omission by comparing statements. Some omissions may cause inconsistencies in the achievement of enterprise tasks such as having an internal document whose creation is never mentioned. In the case study, an omission is detected in the task *validate order* by *account manager*, where *service file* and *invoice elements* are documents that are produced in this task but the user forgot to mention their creation in the detailed description of the task. Such omission detection makes users detail even more the processing of the task. We have noticed that is one of the hardest parts to elicit. Yet it is very important as it is the main source of the functional Requirements. Of course we will unfortunately not be able to detect information that was not mentioned anywhere and whose omission doesn't cause any inconsistencies. This can be prevented by expert reviews or by the use of domain models.

Imprecision: lack of structure and detail. We gain in precision by using a reference model to represent different enterprise views and by guiding the requirements authoring with CNL boilerplates. Still, the users may have a granularity of description different than what is expected. We alleviate this issue by giving users examples of what they are expected to express. Imprecision can also be caused by omission or duplication.

Requirement problems can be interrelated: inconsistency can for instance be the result of omission or duplication. Some of the problems can be real defaults in the enterprise's structure or function. Once detected, we provide stakeholders warnings of their existence and ways to correct them and it is up to them to decide what to do. Highlighting these errors while building the IS may encourage SMEs' stakeholders to provide more information and rethink their organization. It also harmonizes the practices among the enterprise.

4. CONCLUSION

Authoring and verifying requirements needs the involvement of enterprise members, as they are the most likely to define the enterprise structure, behaviours and expectations. A requirements authoring and verification approach suitable to that context is proposed. It is compliant with Enterprise Modeling reference frameworks and favoring the autonomy of SME stakeholders. Functional System Requirements are mainly extracted from the definition of the tasks achieved within the different roles. External partners' requirements were reflected by representing their roles in the IS. Verification among all the enterprise views, arbitration and restitutions are used to enhance the quality of the Stakeholder Requirements and to provide with confidence System Requirements to the development team. The benefit of our work is to be a trade-off between the autonomy of the SMEs stakeholders that requires a simple formalism and the level of formalization needed to operate and automatize the requested support for these engineering activities. The trail of MDE model transformations is now under investigation to accelerate the development of mockups for requirements validation. This way the updated requirements will ideally be re-injected to the transformation/development process.

REFERENCES

Aceituna, D., Walia, G., Do, H., & Lee, S. W. (2014). Model-based requirements verification method: Conclusions from two controlled experiments. Information and Software Technology, 56(3), 321–334.

Amokrane, N., Chapurlat V., Courbis A.L., Lambolais T., Rahhou M (2014). Modeling frameworks, methods and languages for computerizing Small and Medium-sized Enterprises: review and proposal. Interoperability for Enterprise Systems and Applications (I-ESA), 77–87.

Annervaz, K. M., Kaulgud, V., Sengupta, S., & Savagaonkar, M. (2013, November). Natural language requirements quality analysis based on business domain models. In Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on (pp. 676–681). IEEE.

BKCASE Editorial Board. (2014). The Guide to the Systems Engineering Body of Knowledge (SEBoK). v. 1.3. R.D. Adcock (EIC). Hoboken, NJ: The Trustees of the Stevens Institute of Technology. Available: http://www.sebokwiki.org/.

Cao L., Ramesh B. (2008). Agile requirements engineering practices: An empirical study. IEEE Software, 60–67.

Chapurlat V. (2007) Vérification et validation de modèles de systèmes complexes : application à la Modélisation d'Entrepise. HDR Université de Montpellier 2.

Darimont R., Delor E., Massonet P., Van Lamsweerde A. (1998). GRAIL/KAOS: An Environment for Goal-Driven Requirements Engineering. ICSE'9R—20th Intl. Conf: on Software Engineerin.

Fuchs, N. E., Kaljurand, K., & Schneider, G. (2006). Attempto Controlled English Meets the Challenges of Knowledge Representation, Reasoning, Interoperability and User Interfaces. In FLAIRS Conference (Vol. 12, pp. 664-669).

ISO/DIS (2004). Enterprise integration — Constructs for enterprise modelling. ISO/DIS 19440, ISO/TC 184/SC 5.

ITU–T (2008), Recommendation Z.151: User Requirements Notation (URN) — Language Definition, Geneva, Switzerland.

Mavin A., Wilkinson P., Harwood A., Novak M. (2009). Easy Approach to Requirements Syntax (EARS). IEEE International Symposium on Requirements Engineering (RE).

Nikula U., Sajaniemi J., Kälviäinen H. (2000). Management view on current requirements engineering practices in small and medium enterprises. The Fifth Australian Workshop on Requirements Engineering. Queensland University of Technology, Brisbane, Australia, p. 81–89.

Njonko, P. B. F., Cardey, S., Greenfield, P., & El Abed, W. (2014). RuleCNL: A Controlled Natural Language for Business Rule Specifications. In Controlled Natural Language (pp. 66–77). Springer International Publishing.

OMG, (2008). Semantics of Business Vocabulary and Business Rules, Version 1.0.

Schneider, F., Naughton, H., & Berenbach, B. (2012). A modeling language to support early lifecycle requirements modeling for SE. Procedia Computer Science, 8, 201–206.

Schwitter, R. (2010, August). Controlled natural languages for knowledge representation. In Proceedings of the 23rd International Conference on Computational Linguistics: Posters (pp. 1113–1121). Association for Computational Linguistics.

The Standish Group (2009). THE CHAOS REPORT. The Standish Group International, Inc.

Vom Brocke, J., & Rosemann, M. (2010). Handbook on business process management.

Williams, S., Power, R., & Third, A. (2014). How easy is it to learn a controlled natural language for building a knowledge base?. In Controlled Natural Language (pp. 20–32). Springer International Publishing.

Yan, R., Cheng, C. H., Zhang, G., & Chai, Y. (2014). Formal Consistency Checking over Specifications in Natural Languages. arXiv preprint arXiv:1405.5003.