We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter 8

Grounding Functional Requirements Classification in Organizational Semiotics

Richa Sharma

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/67608

Abstract

An information system has its requirements rooted in organizational policies and behaviour, the complexity of which is governed by the hierarchy and the dependencies of the activities within the organization. This complexity makes requirements analysis for an envisioned information system an intricately challenging task. The absence of well-defined body of knowledge clearly specifying which requirements must be looked for further deepens the challenge of requirements analysis. Though requirements are broadly classified as functional and non-functional, a special concern is required for functional requirements as the information system is expected to meet the behaviour of the organization. We explore the role of organizational semiotics in extracting and analysing functional requirements for an envisioned information system. We also report the results of supervised learning to automatically extract the functional requirements from the existing available documentation.

Keywords: organizational semiotics, requirements engineering, functional requirements, business rules

1. Introduction

Software Engineering has come a long way after its inception in 1960 with the famous NATO conferences [1, 2]. The discussions in these conferences are credited with bringing discipline to the activity of software development and laying down the foundations of this field by relating it to mathematics. There have been further developments and innovations in an attempt to realize the goals of systematic, disciplined and quantifiable approach to software development. The earliest proposed waterfall process model for software development evolved towards iterative process models and is now being replaced by the latest agile methodologies. In addition

INTECH
open science | open minds

© 2017 The Author(s). Licensee InTech. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

to process models, programming paradigms have evolved from procedural approach of structured programming [3] to object-oriented programming [4]. However, the goal of a systematic, disciplined and quantifiable approach is still far away. A key role in realizing this goal is played by the requirements, that is the main input to the (engineering) process of software development. Realizing the crucial role of requirements to the design and development of the software, requirements discovery and analysis activities came to be recognized as 'Requirements Engineering (RE)' with the publication of selected papers on RE in Ref. [5] and establishment of regular conferences on RE by IEEE Society. This helped in organizing and bringing discipline to various process models for RE activities and frameworks for analysing requirements. However, the proposed as well as practiced methodologies to ensure consistent, correct, complete and unambiguous requirements have not exhibited the three defining parameters of an engineering approach, namely repeatability, quantifiability and systematic thought process. An attempt to associate these parameters with RE activities calls for a fundamental question–Is the input to RE activities, that is requirements clearly and precisely defined? This is a difficult question, and the challenges are multi-fold in answering this question. Answering this question requires deliberating following points first:

1. What does clear and precise definition of inputs to RE activities signify?
2. What type of software system are we concerned with?
3. Is the solution or answer to one type of system applicable to another one?
4. What is the validity of the proposed solution or answer to the question on inputs to RE activities, that is the requirements?

The answer to the first point above lies in exploring the definition as well as the taxonomy of requirements. We shall present a brief overview of these points in Section 2. The second and third points are overlapping and very much depend on the requirements taxonomy considered. It has been argued in earlier studies that the solution approach to one type of system may not be applicable to another one whether the concern is related to requirements representation [6] or analysis [7, 8] as the systems may range from mission-critical, safety critical applications to enterprise applications and Web-based systems to mobile applications. The last point presents an opportunity to validate one of the proposed requirements taxonomies by either strongly correlating the taxonomy under study to an established framework or by conducting an empirical study at a wide scale.

In this chapter, we shall focus on the last point of validating the functional requirements taxonomy by considering one of the functional requirements classification proposed earlier [9–14]. Though requirements are broadly classified as functional and non-functional, the vital role played by requirements in the development of information systems motivated us to do an in-depth study of functional requirements. Moreover, an empirical study by Kamata et al. [15] on current RE supports our observation that functional requirements need an in-depth and extensive exploration to refine RE processes and methodologies. We shall follow the validation proposition of establishing correlation between an established framework and the functional requirements classification under study.

http://dx.doi.org/10.5772/67608

As points 2 and 3 above suggest, considering a wide spectrum of software systems is not feasible. We shall, therefore, take into account functional requirements in the context of information systems that are database-driven enterprise wide applications such as retail applications, financial applications and ERP systems. Such information systems need to embed organization structure, hierarchy, policies, processes and behaviour in the form of software requirements. Organizational semiotics present a feasible solution towards understanding requirements of an information system. We shall explore the role of organizational semiotics in extracting and analysing functional requirements for an information system considering requirements taxonomy proposed in Ref. [14]. The reason for selecting this classification scheme, in particular, is that the authors in their work [14] have presented classification of functional requirements with regard to information system only. Since this chapter focuses on the role of organizational semiotics towards better understanding (extracting and analysing) requirements of an information system, therefore, the work presented in Ref. [14] is the best suited choice. We shall explore the following research questions in this chapter:

RQ1. Do organizational semiotics provide heuristics to identify various functional requirements types?

RQ2. Do organizational semiotic analysis frameworks or methods to analyse an information system bear any direct/indirect relationship with analysing various functional requirements types?

RQ3. Is it possible to automate the process of identifying functional requirements from existing documentation using organizational semiotics?

These research questions will provide an opportunity to correlate the functional requirements classification scheme presented in Ref. [14] with the established organizational semiotics framework, thereby validating this categorization approach of functional requirements. The rest of the chapter is organized as: Section 2 presents a brief overview of requirements definition and taxonomy. Section 3 presents a brief summary of organizational semiotics followed by details for RQ1 and RQ2. Section 4 presents our study on the possibility of automating the process of extracting functional requirements from existing documentation, thereby addressing RQ3. Section 5 finally summarizes the chapter in the form of discussion and conclusion.

2. Requirements taxonomy

As introduced in Section 1 above, the Requirements Engineering (RE) practices in Information Technology (IT) industry are still far from engineering-oriented approach. RE practices need to adopt a more systematic, repeatable and quantifiable approach. In order to support this approach, we need to start with the basic questions—*'what is meant by requirements?' and 'what types of requirements need to be considered for information systems?'*. We are of the view that a fair understanding of the requirements (inputs to RE activities) will prove beneficial in devising RE methodologies with an 'engineering' perspective. 'Requirements' have been described differently by different authors. According to IEEE standard [16], 'requirement' is

defined as: 'a condition or capability needed by user to solve a problem or achieve an objective; and, a condition or capability that must be met by a system or system component to satisfy a contract, standard, specification, or formally imposed document'. Sommerville [11] defines requirements as a specification of expected system behaviour, or a specific constraint on the system or a user-level description. Despite varying versions, requirements describe the desired behaviour of the developed system and therefore in order to better understand the requirements of an information system, these have been broadly classified in terms of the expected behaviour.

Requirements are usually classified into two broad categories, namely—*Functional requirements* which specify the properties and the behaviour of the information system that must be developed, and the *Non-functional requirements* (NFRs) which describe the constraints on the system as well as the quality aspects of the system. However, requirements have been categorized at a further granular level too allowing elicitation and analysis of requirements to be carried out efficiently. Earlier, White and Edwards [9] proposed following hierarchical levels from requirements capturing point of view:

1. Operational environment—These requirements include external systems and operating needs.
2. System capabilities—These represent functions, behaviour and non-functional requirements.
3. System constraints—These include system architecture and the regulatory policies.
4. Development requirements.
5. Verification and validation requirements.
6. Specification of system growth and change including expected system changes and possible environmental changes.

The viewpoint put forward by White and Edwards has overlaps in system capabilities and system constraints in terms of non-functional requirements. Sommerville, however, has segregated functional and non-functional aspects of requirements. He suggests the following requirements categories [11]:

1. Functional requirements—These represent statements of service that the system should provide, how the system should react to inputs and also in particular situations. These requirements further represent user-level goals and the system goals.
2. Non-functional requirements—These represent constraints on services or functions offered by the system such as timing constraints and standards. NFRs further represent product level, organizational level and external interface constraints.
3. Domain requirements—These represent the features that reflect the domain and can be functional or non-functional.

Recently, Chung and Leite [10] and Slankas and Williams [13] have explored further granular levels of NFRs, and their extraction—both manual and automatic. Similar such

http://dx.doi.org/10.5772/67608

studies in the context of functional requirements have been carried out by Ghazarian [12], and Sharma and Biswas [14]. Ghazarian has studied nearly 15 Web-based enterprise system projects from the point of view of identifying atomic functional requirements. His study reveals 12 classes of functional requirements, namely: (1) data input, (2) data output, (3) data validation, (4) business logic, (5) data persistence, (6) communication, (7) event trigger, (8) user interface navigation, (9) user interface, (10) external call, (11) user interface logic and (12) external behaviour. Sharma and Biswas [14] have applied Glaserian Grounded Theory approach [17] on requirements specification documents from five information systems to identify seven categories of functional requirements, namely: (1) entity modelling requirements, (2) user interface requirements, (3) user privileges requirements, (4) user interaction requirements, (5) business workflow requirements, (6) business constraints requirements and (7) external communication requirements. Of these two available classification schemes—by Ghazarian [12], and Sharma and Biswas [14]—we have selected the latter one for our work because while studying these two schemes, we observed that the taxonomy of functional requirements as proposed by Ghazarian [12] is close to the solution domain (developed code) and not the problem domain (requirements specification) of information systems. RE is the only phase of software development that deals with both the problem space and the solution space of the envisioned software system [18] as this phase only bridges the gap between 'as-is' system and the 'to-be' system. Nevertheless, the starting point of any software project is the problem space, from where the requirements of an information system are drafted. Therefore, we selected the functional requirements taxonomy proposed by Sharma and Biswas [14] for our study.

We are interested in validating whether the functional requirements categories proposed by Sharma and Biswas [14] are meaningful and useful by grounding them in organizational semiotics framework. We shall do so by exploring first two points from our research questions—(1) RQ1: Do organizational semiotic suggest heuristics that can help in identifying the proposed functional requirements types? and (2) RQ2: Do organizational semiotic analysis methods to analyse an information system bear any direct or indirect relationship with analysing various functional requirements types? We shall explore these points in the following section. Before discussing these points, the following section presents a brief introduction to organizational semiotics.

3. Organizational semiotics

The crucial role played by requirements in the development of information systems has resulted in proposing various approaches to correctly identify and analyse the requirements for the information system. Granular classification of functional requirements is one such possible solution. We have presented this solution approach in detail in Section 2 above. Semiotic analysis framework is another possible solution that has been applied to understanding and analysing requirements of an information system by several authors like [8, 19–24]. In this section, we shall study the relationship between these two approaches, and how one of the former approaches (classification of functional requirements) is rooted in the latter approach.

Organizational semiotics deal with the study of organizations using the concepts and methods of semiotics, where semiotics are the study of signs dealing with generation, transformation and communication of signs that people use for various purposes [25]. Organizational semiotics study is based on the fundamental observation that all organized behaviour is affected through communication and interpretation of signs by people, individually and in groups. Organizational semiotics analysis method, referred to as Methods for Eliciting, Analysing and Specifying Users' Requirements (MEASUR), proposed by Stamper [26] and further enriched by Liu [24, 27] has evolved into semiotic methods or framework for information systems. A radical, subjectivist stance has been accepted as the basic philosophy for developing this set of methods and tools for information systems development. The introduction of subjectivity is required when the context is of information system development as there are multiple stakeholders of an information system, each having varyingly different viewpoints on requirements of that information system. A brief overview of these methods for analysing information systems is presented in Section 3.1, followed by the discussions on first two RQs in further subsections.

3.1. Organizational semiotics for information systems

Organizational semiotics consider an organization as an information system in which information is created, processed and used. It tries to understand organizations in terms of its semiotics—signs, texts, documents, sign-based artefacts (contracts) and communication between stakeholders [28]. The goal of organizational semiotic study is to find new and insightful ways of analysing, describing and explaining organizations. Semiotic method for information systems, MEASUR provides a framework for planning, developing and maintaining information systems. It comprises of three key methods for analysing information system to be developed for an organization. These three key methods [24] include as follows: problem articulation method (PAM), semantic analysis method (SAM), and norm analysis method (NAM).

PAM can be applied at the initial stage of an information system development when the requirements, gathered for the system to be developed, are at a very abstract or high level with a lot of vagueness and ambiguity in the organizational context. PAM can help in better understanding the organizational structure and the scope of system to be developed. The techniques employed by PAM include as follows: (1) *unit system identification* to illustrate a particular course of action and agents involved in that action, (2) *stakeholder identification* to identify relevant groups or parties and their interest in an organization's products and services, (3) *collateral analysis* to structure problem situation into a central course of action and surrounding or collateral activities, (4) *system morphology* to clarify three basic functional areas (i.e. substantive, communication and control) of a socio-technical or a business information system; each of these components can, in turn, be treated as a unit for continued analysis, and (5) *valuation framing* to reveal the cultural behaviour of the stakeholders involved in the information system.

SAM emphasizes focusing on one articulated unit system or focal problem and suggests that analysts should encourage stakeholders or business users to describe their requirements within the scope of that focal problem. The required functions of the system are specified in

http://dx.doi.org/10.5772/67608

the form of an ontology model. This method is directed towards a focal action, and the agent responsible for carrying out that action. The relationship between these two is captured in the form of simple and well-formed formula (wffs) as:

- <agent-term> <action-term>

These wffs are then presented in the form of ontology models for visual representation that assists in visualizing the relationships between various agents and their actions in an information system.

SAM is followed by NAM which provides a way to specify the agents' patterns of behaviour in the business system. A norm specifies conditions in which an action may (or should/must or must not, etc.) be performed by some agent. These norms act as conditions and constraints; they govern agents' behaviour, normally in a prescriptive manner to decide when certain actions will be performed. Norms, in conjunction with the semantic model, clearly define the roles, functions, responsibilities and authorities of agents.

The organizational semiotic analysis methods, as discussed above, do offer heuristics in terms of lexical patterns for extracting requirements automatically from the available documentation instead of manually going through the existing available documentation and then finding the requirements. Manual intervention cannot be completely ruled out at the time of requirements gathering from documents or eliciting from clients. Nevertheless, some form of automated assistance would be of help to analysts or requirements engineers. We present such lexical heuristics from organizational semiotic approach in the following subsection.

3.2. Heuristics from organizational semiotics for identifying functional requirements

Organizational semiotic analysis approach applies to the complete process of information system development [24] including the requirements understanding and analysis as well. Liu has established the point that Requirements Engineering (RE) is a process of semiosis by identifying the concepts required for sense making of requirements specifications. Liu indicates that requirements specifications are the 'signs' corresponding to the actual requirements having origin in the business domain under study. These actual requirements formulate the 'objects' in semiosis process. The 'interpretant' is the agreed understanding of the sign, that is the requirements specification between analysts or requirements engineers and business users as other stakeholders. MEASUR methods consider organizations themselves as information systems and social norms as unit of specification. These methods are manually applied to an information system under study. We extend this idea and propose heuristics based on MEASUR methods to identify functional requirements from existing documentation. The existing documentation could be in the form of Request for Proposal (RFP) document or organizational structure and policies document, or may be some regulatory document. RFP is usually identified as the first reference document for software requirements, providing an insight into business rules and organizational activities. Referring to any of these documents, we can identify functional requirements by using the heuristics based on MEASUR methods, as described below:

1. Possible candidates for unit systems and focal problems/actions include verb phrases present in the form of participle, and the verb in base form ending in 'tion', 'scion' or 'cion', 'al'. Though not all such verb phrases would be unit systems actually, nevertheless, these serve as heuristic to automatically extract possible unit systems from the existing documentation. These candidates correspond to 'use-cases' in RE terminology.
2. Nouns or noun phrases are possible candidates of stakeholders, agents in the information system. These correspond to entities (classes in object-oriented paradigm) in RE terminology.
3. Statements having these keywords—'communication', 'message', 'queuing message', 'send message' qualify for external or user interface communication requirements.
4. Verbs and verb phrases qualify for actions performed by actors or agents. These phrases serve as heuristic to find user privilege requirements, business workflow requirements, and business constraints requirements.
5. Norm analysis patterns serve as the heuristic to identify business workflow requirements. These patterns are generally represented as [29]:

- If <condition> then <consequence>

Behavioural norms may have more specific form depending on the complexity of behaviour as:

- Whenever <condition> if <statement> then <agent> is <deontic operator> to do <action>

Though Liu and Dix [29] have proposed above-mentioned two norm patterns, but the expression for norms can take several other forms. We have observed following patterns describing norms in an organization through manual study of requirements documents:

- In case <condition> then <consequence>
- <Consequence> provided <condition>
- When <condition> then <consequence>
- Once <condition> then <consequence>
- Only <condition> <consequence>
- In order to <consequence to hold> then <condition>
- <Condition> in order to <consequence to hold>
- <Condition> must (hold) <consequence with infinitive clause>

The organizational semiotic approach does not offer any heuristic to identify graphical user interface (GUI)-related requirements. We have used the above-mentioned heuristics to identify five categories of functional requirements (excluding GUI-related requirements) as proposed by authors in Ref. [14] for employee self-service (ESS) module of HR management project developed at our industry partner's end. Since the project had started following agile approach, therefore the development team could not collaborate for the entire

http://dx.doi.org/10.5772/67608

project with us, and therefore, we confined our experimental study to this one module only. Following the heuristics described in the points above, we carried out lexical search and started tagging the user management module's proposal document for the presence of verb and noun phrases. The proposal document for this module was a small document running into pages only. We found 14 unit systems following first heuristic and presented these to the development team for validation. They observed that we found four false unit systems and that we could not identify three unit systems. These three unit systems did not follow the lexical pattern of first heuristic. Of the four falsely reported unit systems, two were actually attributes of an information content, and one was related to the style of writing the document. The author of that document had a peculiar style of writing every use-case by mentioning—'Provision to …', and the presence of word 'provision' led to ignoring other unit cases. Following second heuristic, we found 31 candidates for agents—of these, only three candidates are in the role of 'actors' (entity modelling requirements)—this observation was in agreement with the development team working on ESS module. However, the team pointed out that the heuristic is not sufficient to detect abstract concepts.

Such challenges are always there with lexical heuristic approaches, but we believe that more and more experimentation will enable us in refining the heuristics and the solution approach to automate the process of extracting functional requirements from the existing documents like proposal document in our case. For the sake of clarity and brevity, we are not presenting the observations for other heuristics. To summarize, our overall observations using the above-mentioned heuristics were approximately 60% close to the requirements identified by the development team. This percentage is sufficient to infer that heuristics can serve as guiding tool for functional requirements extraction and that functional requirements classes and organizational semiotic heuristics are closely related.

3.3. Organizational semiotics analysis framework v/s functional requirements

An in-depth study of organizational semiotic analysis framework, MEASUR indicates a strong and direct correlation between the framework's analysis methodologies and, the types of functional requirements [14] (except for GUI-related requirements) considered in this chapter. We summarize the relationship for each functional requirement as below:

1. Entity modelling requirements—These requirements represent the domain model of the organization. The domain-relevant concepts are modelled as entities while implementing the information system for an organization. PAM of stakeholder identification helps in identifying agents and stakeholders of the system. These, in turn, correspond to entity modelling statements from the reference documents for an information system. This method defines roles in six different categories, thereby making it easier to identify the stakeholders. These six categories are as follows: actor, client, provider, facilitator, governing body, and bystander. For example, consider the following requirements statement:

   RS1: The system shall only allow a user with an authorized official (AO) role to create a new submission.

The stakeholder identification method in PAM analysis helps in identifying *user* as agent, and *Authorized Official (AO)* as role name. Following entity modelling requirements premises, RS1 has four possible concepts—system, user, authorized official (AO), and submission. Analysing RS1 manually indicates that though there are four concepts but modelled entity is 'user', whose role is that of 'Authorized Official (AO)' and 'submission' is an affordance for 'user'. We observe stakeholder identification results in an enriched information while entity modelling requirements yield in a superset of information from stakeholder identification. It can be observed that there is no conflict between the resulting entities/concepts from two methods; one has enriched details while other has more number of concepts. Thus, it verifies that there exists direct relationship between stakeholder identification of PAM and the entity modelling requirements.

2. User interface requirements—These requirements represent the presentation layer of the information system, that is the graphical user interface used by the agents to interact with the information system. All those statements that describe the layout of information on interface or flow of information from one level to another interface level belong to the category of user interface requirements. These requirements remain undiscovered by MEASUR methods, and therefore, these requirements do not bear any relationship to organizational semiotic analysis approach. Nevertheless, it can be observed that this approach can gain from granular classification of functional requirements to enrich its identified set of requirements. A sample of user interface requirements is illustrated below:

   RS2: Any entity/text on the user interface that is a link should be in blue font and underlined.

   There is no direct analysis method in organizational semiotics framework, MEASUR, to identify user interface requirements like RS2. But, the requirements identified using this framework can be further enhanced by adding GUI-related requirements for which this category of functional requirements define identification criterion.

3. User privileges requirements—These requirements describe various roles played by the business users in an organization and the privileges associated with those roles. PAMs of stakeholder identification and SAM analysis method together correlate in terms of identifying privileges associated with different roles in an organization.

   Considering RS1 again, for example,—this statement on one hand contains entity modelling requirements, and at the same time, it describes role of 'Authorized Official (AO)' who has the privilege to create a new submission. SAM analysis (considering the focal problem of submission) adds value to the information obtained using stakeholder analysis technique of PAM—it associates affordance 'submission' to the role of 'Authorized Official (AO)' possessed by agent and 'user'. Thus, we observe a strong correspondence between the outputs of organizational semiotics framework, MEASUR for identifying user roles and functions with the user privileges requirements.

4. User interaction requirements—These requirements describe how an end-user of an information system will interact with the system through user interface. Though MEASUR methods do not mention much about human-computer interaction but system morphology PAMs could possibly relate to user interaction requirements. This method requires exploring the system with the goals of identifying—substantive behaviour of the agents, message passing from one person to another inside and outside the organization, controls

http://dx.doi.org/10.5772/67608

flow to ensure smooth communication and substantive actions. System morphology (communication and control) method can help in identifying these requirements as in the 'to-be' system, nature of interaction and communication between people in an organization might get replaced in the form of interaction/communication with the system (valuation framing). An example of user interaction requirements statement:

RS3: The system shall allow the user to edit a submission by clicking on the Facility column. The system shall allow the Facility column to be clicked only when the submission is still underway.

The above statement, RS3 describes how a user interacts with the system to edit an affordance, *submission*. System morphology technique of PAM, thus, can help in extracting user interaction requirement. RS3 is designated as an example of this type of requirement as it describes 'how-to-use' part of user interface. That's how system morphology method and user interaction requirements bear a correspondence with each other.

5. Business workflow requirements—These requirements describe business rules, policies and procedures. In turn, these business rules and policies provide justification to the agents' behaviour within the information system. SAM and NAM of semiotics yield in identifying the actions, the agents responsible for those actions, the conditions under which the action would be carried out, and the actions in consequence, thereby giving a complete view of a business workflow requirement. Control technique of social morphology PAM also results in identifying what can be referred to as business workflow requirements. Considering a requirements statement from the famous London Ambulance Service case study [30, 31]:

   RS4: When an operator receives a phone call concerning a medical emergency, he should dispatch a nearby available ambulance.

   RS4 follows one of the norm analysis patterns presented above, so following NAMs, this statement can be marked as business workflow requirements. In this statement, the agent—'operator' is in the role of actor and is responsible for the action of dispatching an available ambulance. Thus, we can infer the observations from SAM and NAM methodologies for analysing information systems agree with the identification of business workflow requirements.

6. Business constraints requirements—These requirements correspond to the constraints on the information system apart from business workflow logic. Such additional constraints may arise because of organizational policy, external regulatory bodies or market regulations in which the organization is operating or possibly, technical constraints. SAM and NAM help in finding business workflow as well as constraint requirements. Business constraints requirements can be distinguished from business workflow requirements by checking the agent of action under consideration. If the agent is in the role of governing body or facilitator, then the corresponding requirement is an instance of business constraints requirement.

   RS5: The user must have Javascript enabled for the message prompts to occur.

   In RS5, the agent—'user'—is in the role of facilitator for enabling Javascript. In this manner, checking the role of agent can help in designating business constraints requirements, thereby indicating correspondence between organizational semiotic analysis methods and business constraint requirements.

7. External communication requirements—These requirements describe interaction of the information system with other systems or agents outside its scope. PAM of system morphology with a focus on communication with external agents (i.e. agents who are related to the system under study but are actually out of its scope) can help in extracting external communication requirements. Following statement is an example of external communication requirement where the database of the system is modified by an external trigger:

   RS6: Updates to the ALMIS database in the system are commonly performed via remote data transfer. Remote data transfer is commonly accomplished using FTP over the Internet.

   Communication and control techniques of system morphology method of PAM indicate the presence of communication with an external agent or bystander, 'remote data transfer'. RS6 presents an example where observations from PAM methodology and external communication requirements are in agreement with each other, indicating a direct relationship between the two.

In this section, we have addressed our first two research questions—RQ1 and RQ2. We have found that organizational semiotics do offer heuristics to identify different types of functional requirements except for user interface-related requirements. This exceptional case can be attributed to the very formalism of organizational semiotics that has its roots in organization's structure and behaviour of its people, that is the scope of organizational semiotics is confined to the problem domain of information systems and not to the solution domain (the layout of the system to be developed). Addressing RQ2, we have similar observation that there exists direct correlation between organizational semiotic analysis frameworks to analyse an information system and the functional requirements types of an information system with an exception for user interface-related requirements. This leads to infer that the functional requirements types (except for user interface-related requirements) as proposed in [14] are grounded in organizational semiotics bearing a strong correspondence with their analysis methodologies—PAM, SAM and NAM. Secondly, the heuristics from organizational semiotics are helpful in automatically extracting various functional requirement types from available documentation, but it has to be followed by manual intervention. The next section considers the possibility of automated extraction of functional requirements from existing documentation (requirements corpus).

4. Automated extraction

In this section, we explore our third research question, RQ3—Is it possible to automatically identify the different categories of functional requirements in the available requirements documents. A major challenge in addressing this point is that of atomicity of a requirements statement. One single statement can have instances of different types of functional requirements. For example: RS1 is an instance of both entity modelling requirements and the user privilege requirements. The fact that the requirements statement can have multiple forms of expressions in natural language worsens the challenge. We have observed in Section 3.2 that lexical heuristics, though provide a solution to extracting functional

http://dx.doi.org/10.5772/67608

requirements, are not well-accepted by practitioners as they feel the approach is as good as manual analysis techniques. If the approach can be automated or semi-automated, then the solution would have higher chances of acceptance by practitioners. Machine learning classification algorithms offer a seemingly feasible solution, and we explore the viability of this solution in this section.

Machine learning (ML) is about the construction and study of systems that can learn from the data. A broad classification of machine learning algorithms identifies two types of learning: supervised and unsupervised. Supervised learning makes use of the guiding function to map inputs to desired outputs (also referred to as labels, because these are often provided by human experts labelling the training set). Unsupervised learning, on the other hand, models a set of inputs by grouping or clustering common instances/patterns.

In this study, we have used supervised ML technique considering labelled or annotated documents as input to our study. We have explored Naïve Bayes, Bayes net, K-Nearest Neighbourhood and Random Forest algorithms to identify statements signifying different functional requirement types. Naïve Bayes is a probabilistic classifier that applies Bayes' theorem with strong (naive) independence assumptions. The underlying assumption in Naïve Bayes algorithm is that the presence or absence of a particular feature bears no relationship to the presence or absence of any other feature, given the class variable. Despite this assumption, Naive Bayes classifier proves to be quite effective in a supervised learning setting. Bayesian network, in contrast, makes use of conditional dependencies. KNN classifier classifies objects by a majority vote of its neighbours. Random forests are an ensemble learning method for classification (and regression). A multitude of decision trees is constructed at training time in this algorithm, and the final output class that is the mode of the classes given as output by individual trees.

The common metrics used to check the result of ML algorithms are as follows: precision, recall, accuracy and F-measure. Of these, we have used precision, recall and F1-measure to compare the results of these learning algorithms to find which algorithm suits better for automated extraction of requirements. Precision defines in terms of the fraction of retrieved instances that are correct, whereas recall refers to the fraction of correct instances that are retrieved. Abbreviating requirements statements as 'RS', we define precision and recall for our study as:

Precision = True Positive RS Type/(True Positive RS Type + False Positive RS Type)

Recall = True Positive RS Type/(True Positive RS Type + False Negative RS Type)

Here, 'True Positive RS Type' indicates correct predictions for the category of functional requirements statement. 'False Positive RS Type' statements are incorrectly labelled as belonging to that class of functional requirements. 'False Negative RS Type' statements are the predictions which were not labelled as belonging to an appropriate functional requirements type but should have been.

F-measure considers both the precision and the recall of the test, representing weighted average of these two metrics (precision and recall). F-measure reaches its best value at 1, and the worst score is 0. It is defined as:

F-measure = 2x (Precision × Recall)/(Precision + Recall)

4.1. Requirements corpus

We prepared our data set (requirements corpus) using text version of the requirements documents by copying the documents to text file. We had access to nearly eight requirements documents of varying sizes in terms of counts of statements. We dropped the non-functional requirements section while preparing data set as our evaluation study is focused towards functional requirements. Though atomicity of functional requirements is desirable, but it is not always possible to discretely express one type of functional requirements with natural language expressions. Therefore, we allowed one statement to belong to more than one category of the functional requirements. The lexical heuristics are present in the requirements statement, therefore, in our case, we have composed the feature vector to be presented as input to ML algorithms as requirements statement followed by 'yes' and 'no' indicators for the presence and absence, respectively, of a type of functional requirement. Two sample statements from feature vector are illustrated below for making the point clearer:

- 'The document contains following sections', no, no, no, no, no, no, no
- 'An administrator should be able to perform all the search queries as a normal user.', yes, no, yes, yes, no, no, no

Here, the first sample statement does not correspond to any type of functional requirement. Consequently, it has all 'no' labels. The second statement indicates the presence of entity-modelling requirement, user privilege requirement, and the user interaction requirement. Therefore, this statement has corresponding 'yes' labels to it, and to signify the absence of rest of the functional requirement types, there are 'no' labels corresponding to them.

The task of annotating the requirements statements with 'yes' and 'no' labels corresponding to presence/absence of different types of functional requirements in the statement under study was performed by five human subjects to ensure fairness and unbiasedness of our study. The subjects chosen for the study are research scholars and master students, who have done courses on Software Engineering and Business Modelling. Two of the selected subjects had industry experience too. After dropping the non-functional requirements, the details on the size of the documents studied are presented in **Table 1**.

Manual annotation by different subjects can possibly have lot of variations depending on an individual's thought process. Therefore, manual annotation could be a potential threat to the validity of our results. In order to mitigate this threat, the author of this chapter organized meetings with the subjects and shared the background of the annotation task to be done. The details of the proposed classification were discussed thoroughly as subjects might get confused in closely related categories such as user interface requirements and user interaction requirements. We also performed validity check for annotation by selecting a random sample of 100 statements in one of the initial meetings and labelling this set. We, then, performed peer review of those annotations. The result of peer review revealed that there are not drastically differing views of the rule labelling. Once satisfied with the observations from peer review, we proceeded with our experiments on the annotated requirements corpus.

http://dx.doi.org/10.5772/67608

Document	Size (number of statements)
Doc 1	208
Doc 2	798
Doc 3	1135
Doc 4	1251
Doc 5	351
Doc 6	1305
Doc 7	900
Doc 8	230
Total	6187

Table 1. Requirements corpus details.

4.2. Evaluation study and observations

We performed our experiments by applying Naïve Byes, Bayes Net, K-Nearest Neighbourhood and Random Forest algorithms to our annotated corpus. Our classification results are based on n-fold cross-validation study as recommended by Han et al. [32]. We have computed precision, recall and F-measure for each of the classifier. In n-fold cross-validation, data are distributed randomly into n-folds where each fold is approximately of equal size and equal response classification. We have used Wekahttp://www.cs.waikato.ac.nz/ml/weka/ tool for carrying out our experimental study. We loaded the files in Weka and converted the annotated statements to word vectors using Weka filter. In the first phase of our study, we used the documents in their original form, that is without any changes to the word form, or applying any filter. The results of the first step of our study are presented in **Table 2**.

The next phase of our evaluation study included filters—we dropped stop-words at the time of data set preparation. Stop-words refer to a list of words that should be filtered out during classification due to either commonality of words or domain-specific generality of words. We have considered determiners only (a, an, the) as stop-words in this work, and we have not observed much of an improvement after applying stop-words as filters. Instead, KNN and Bayes net performance dropped as reported in **Table 3**.

The experimental study that we carried out for automated extraction of functional requirements is just a starting first step towards effectively utilizing ML classification algorithms for classifying functional requirements, and needs to be further worked up further refinements. These results are not very good because high recall has resulted in lower precision and a high precision yielded in lower recall. Bayes net algorithm only has yielded in both good results and good recall. Nevertheless, the results are encouraging in the sense that heuristics also allowed us to be 60% closer to actual requirements (that too with a small document), and ML approach too has nearly 60–70% of correctness in terms of precision and recall.

Classifier	Precision	Recall	F-measure
Naïve Bayes	0.396	**0.717**	0.483
Bayes net	**0.607**	**0.646**	**0.608**
Random forest	**0.747**	0.401	0.481
KNN	0.608	0.585	0.584

Table 2. Functional requirements extraction—10-folds cross-validation study.

Classifier	Precision	Recall	F-measure
Naïve Bayes	0.396	**0.717**	0.483
Bayes net	0.476	0.54	0.483
KNN	0.586	0.492	0.512

Table 3. Functional requirements extraction with determiners as stop-words.

5. Discussion and conclusion

In this chapter, we have presented the role of organizational semiotics in identifying functional requirements for an information system. We hypothesized that organizational semiotics do provide heuristics to identify various functional requirements categories, and there exists a direct relationship between semiotic analysis framework and analysis based on identifying functional requirements classes. Our study reveals that semiotic analysis framework and functional requirements categorization approach (to better understand the requirements) bear strong correspondence with each other and, at times complement each other provided the categorization of functional requirements is meant for information systems. The software systems have a wide spectrum as we have elaborated in the chapter, and one solution or one classification scheme for a type of system may not be applicable to another type. Secondly, our study reinforces functional requirements categorization, based on grounded theory, in context of information system [14] by grounding the classification scheme in an established theory of organizational semiotics.

We believe that our study around organizational semiotics and functional requirements will prove useful in bringing an organized and systematic approach to requirements engineering for information systems. Organizational semiotic analysis approach has slowly paved the way to information systems engineering though with certain gaps in context of information systems, where these gaps can be bridged by deliberating carefully as to what requirements we want to consider while developing an information system. Additionally, requirements analysis methods considering functional requirements categories first may gain from the knowledge of heuristics rooted in organizational semiotics.

With increasing complexity of software systems being developed, it would be worthwhile to develop an automated approach to assist practitioners. We have explored two separate

approaches towards the purpose—one semi-automated using lexical heuristics and word-tagging, and the second of ML classification. The observations from both the approaches are almost similar (approximately 60%). It is difficult to judge which one is a better solution as this would require more extensive study and experimentation. We intend to carry out this study as part of our future work. Additionally, we intend to improve upon the heuristics from organizational semiotics analysis framework.

Author details

Richa Sharma

Address all correspondence to: sricha@gmail.com

BML Munjal University, Gurgaon, India

References

[1] Naur P, Randell B. Software Engineering Report on a conference sponsored by NATO SCIENCE COMMITTEE. In: Proceedings of the NATO Software Engineering Conferences; 7-11 Oct. 1968; Garmisch, Germany. Brussels: Scientific Affairs Division, NATO; 1968.

[2] Buxton J N, Randell B. Software Engineering Techniques, Report on a conference sponsored by NATO SCIENCE COMMITTEE. In: Proceedings of the NATO Software Engineering Conferences; 27-31 Oct. 1969; Rome, Italy. Brussels, Belgium: Scientific Affairs Division, NATO 1969.

[3] Royce W W. Managing the development of large software systems. In: Technical Papers of Western Electronic Show and Convention; 1970; Los Angeles, USA. p. 328-338.

[4] Booch G. Object-oriented Analysis and Design with Applications. 2nd ed. Santa Clara, CA: Addison-Wesley; 1998.

[5] Thayer R H, Dorfman M, editors. System and Software Requirements Engineering. Los Alamitos, CA: IEEE Computer Society Press; 1990.

[6] Hayes H, editor. Specification Case Studies. UK: Prentice Hall International; 1992.

[7] Lamsweerde A van. Requirements engineering in the year 00: a research perspective. In: International Conference on Software Engineering; 2000; Limerick, Ireland. p. 5-19.

[8] Sharma R, Biswas K K. Using norm analysis patterns for automated requirements validation. In: IEEE Second International Workshop on Requirements Patterns (RePa), co-located with RE'12; 24 Sept.; Chicago, USA. IEEE; 2012. doi:10.1109/RePa.2012.6359965

[9] White A, Edwards M. A requirements taxonomy for specifying complex systems. In: 1st IEEE International Conference on Engineering of Complex Computer Systems; 1995. p. 373-376.

[10] Chung L, Leite J.C.S. do Prado. On non-functional requirements in software engineering. In: Borgida A, Chaudhari V K, Giorgini, P, Yu E, editors. Conceptual Modeling: Foundations and Applications. New York: Springer; 2009. p. 363-379.

[11] Sommerville I. Software Engineering. 9th ed. India: Pearson; 2011.

[12] Ghazarian A. Characterization of functional software requirements space: the law of requirements taxonomic growth. In: 20th IEEE International Requirements Engineering Conference; 24-28 Sept.; Chicago, USA. IEEE; 2012. p. 241-250.

[13] Slankas J, Williams L. Automated extraction of non-functional requirements in available documentation. In: International Workshop on Natural Language Analysis in Software Engineering (NaturaLise), co-located with ICSE-2013; 18-26 May; San Francisco, CA. IEEE; 2013. p. 9-16.

[14] Sharma R, Biswas K K. Functional requirements categorization grounded theory approach. In: International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE); 29-30 April 2015; Barcelona, Spain. IEEE; 2015.

[15] Kamata M I, Yoshida A Y, Yoshida H, Aoki N. Figure out the current software requirements engineering—what practitioners expect to requirements engineering? In: 14th Asia-Pacific Software Engineering Conference (APSEC 2007); 5-7 December; Nagoya, Japan. 2007. p. 89-96.

[16] IEEE Computer Society. IEEE Standard Glossary of Software Engineering Terminology. IEEE Standard 610.12-1990, 1990.

[17] Glaser B G. Theoretical Sensitivity: Advances in the Methodology of Grounded Theory. Mill Valley, CA: Sociology Press; 1978.

[18] Jureta I. Problem and solution concepts in requirements engineering. In: The Design of Requirements Modelling Languages. Switzerland: Springer International Publishing; 2015. p. 21-28. doi:10.1007/978-3-319-18821-8

[19] Stamper R, Liu K, Hafkamp M, Ades Y. Understanding the roles of signs and norms in organisations—a semiotic approach to information systems design. Journal of Behaviour & Information Technology. 2000; **19**(1):15-27.

[20] Salter A, Liu K. Using semantic analysis and norm analysis to model organisations. In: 4th International Conference on Enterprise Information Systems (ICEIS-2002); 3-6 April; Spain. 2002.

[21] Shishkov B, Xie Z, Liu K, Dietz J L G. Norm analysis to derive use cases from business. In: 5th Workshop on Organizational Semiotics (OS 2002); 14-15 June; Delft, The Netherlands. 2002. p. 187-195.

[22] Baranauskas M C C, Schimiguel J, Simoni C A C, Medeiros C B. Guiding the process of requirements elicitation with a semiotic approach. In: 11th International Conference on Human-Computer Interaction; Las Vegas, USA. 2005. p. 100-110.

[23] Regev G, Wegmann A. Where do goals come from: the underlying principles of goal-oriented requirements engineering. In: 13th IEEE International Requirements Engineering Conference (RE'05); Paris, France. 2005. p. 353-362. doi:10.1109/RE.2005.80

[24] Liu K, Li W. Organisational Semiotics for Business Informatics. London and New York: Routledge Taylor and Francis Group; 2015. p. 428.

[25] Peirce C S. Collected Papers of Charles Sanders Peirce. In: Hartshorne C, Weiss P, Burks A W. Cambridge, editors. Mass.: Harvard; 1931-58

[26] Stamper R K. Information in Business and Administrative Systems. New York: John Wiley and Sons; 1973.

[27] Liu K. Semiotics in Information Systems Engineering. Cambridge: Cambridge University Press; 2000.

[28] Gazendam H W M. Organizational semiotics: a state of the art report. Semiotix. 2004; **1**(1).

[29] Liu K, Dix A. Norm Governed Agents in CSCW. In: First International Workshop on Computational Semiotics; 1997.

[30] Musick E and Welch H. The 1992 London Ambulance Service Computer Aided Dispatch System Failure [Internet]. Available from: http://erichmusick.com/writings/technology/1992-london-ambulance-cad-failure.html [Accessed: 21 January 2017]

[31] Finkelstein A and Dowell J. A comedy of errors: the London ambulance service case study. In: 8th International Workshop on Software Specification and Design (IWSSD); 1996.

[32] Han J, Kamber M, Pei J. Data Mining: Concepts and Techniques. 3rd ed. Morgan Kaufmann; Massachusetts, USA. 2011.

IntechOpen

IntechOpen

IntechOpen