JISTEM - Journal of Information Systems and Technology Management
Revista de Gestão da Tecnologia e Sistemas de Informação
Vol. 9, No. 1, Jan/Apr. 2012, *pp.23-38*
ISSN online: 1807-1775
DOI: 10.4301/S1807-17752012000100002

USING GROUNDED THEORY AS A METHOD FOR SYSTEM REQUIREMENTS ANALYSIS

Mohanad Halaweh
University of Dubai, Dubai, UAE

ABSTRACT

Requirements analysis (RA) is a key phase in information systems (IS) development. During this phase, system analysts use different techniques and methods to elicit and structure the system's requirements. The current paper rationalises the use of grounded theory (GT) as an alternative socio-technical approach to requirement analysis. It will establish theoretically that applying grounded theory procedures and techniques will support and add value to the analysis phase as it solves some problems of the existing traditional and socio-technical system design methods. Furthermore, to validate this proposal, a case study applying GT on a real project will demonstrate its applicability and success for requirement analysis. Implications of its application are also discussed.

*Keywords***:** Grounded theory; IS development; requirements analysis; requirements engineering.

1. INTRODUCTION

Traditional requirement analysis methods have failed because systems developers focus mainly on the technical functions and constraints of the systems. Therefore, it is widely recognised that the success of IS development involves appreciation of the social and organisational aspects besides the technical aspects (Clegg, 2000; Doherty & King, 2005; Madsen and Vidgen, 2009; Mumford, 1997; Reddy et al. 2003; Sabine and Vidgen, 2009), which leads to design systems that are more acceptable by end users. For this reason, a lot of research has advocated the use of socio-technical systems design methods (see, for example, Berg and Toussaint, 2003; Doherty and King, 2005; Eason, 2007; Sommerville and Dewsbury, 2007), which aim to give balance to social and technical issues when a new system is designed (Mumford, 2000). Baxter and Sommerville (2010) pointed out that failure to adopt socio-technical approaches to systems design can increase risks in which systems will not make contributions to the goals of the organisation. They mentioned that systems often meet the technical 'requirements', but are considered to be a 'failure' because they do not deliver the expected support for the real work in the organisation.

Several socio-technical system design approaches have been used such as soft systems methodology (Checkland, 1981), ethnographic workplace analysis (Martin and

---

Manuscript first received/*Recebido em* 29/09/2011 Manuscript accepted/*Aprovado em*: 12/03/2012

Address for correspondence / *Endereço para correspondência*

Mohanad Halaweh, Dr. Assistant Professor, College of Information Technology, University of Dubai, Maktoum Road, Deira, P.O.Box: 14143, Dubai – UAE Tel. +971 4 2072647 , Fax. +971 4 22 42813, E-mail: mhalaweh@ud.ac.ae

Sommerville, 2004), contextual design (Beyer & Holtzblatt, 1999), cognitive work Analysis (Vicente, 1999) and Alter's (2006) work system method. However, Baxter and Sommerville (2010) discussed several problems with the existing approaches to socio-technical systems design. For example, they noted that adopting socio-technical approaches involves collaboration between different disciplines, which is accepted. However, the problem is mainly related to failures in understanding and communication among development team members who are from different disciplines – one discipline does not fully understand what the other disciplines can do. Dekker et al. (2003), for example, have argued that practitioners of ethnography and contextual design fail to deliver products that can be used by other development team members. That is to say, some of the work carried out by ethnographers and those involved in contextual inquiry do not go far enough because essentially it is concluded after data collation, rather than analysing data so that it can be ready to be used by others (e.g. designers and programmers). Another communication problem was also found by Al-Rawas and Easterbrook (1996). They stressed that there is a communication problem between end-user and the development team due to technical notations used. They found that end-users face difficulty understanding the notations used to model their requirements by designers and developers. In particular, when 35 developers were asked whether their clients found the notations they used readable, only four of them said that they were understandable to end-users. This is because end-users are unfamiliar with formal specification languages used to model their requirements and thus they cannot validate their requirements. Vijayan and Raju (2011) and Geisser and Hildenbrand (2006) also pointed out that most systems failure is due to the poor communication between users and analysts. Another problem with existing socio-technical methods is related to the level of abstraction (Baxter and Sommerville, 2010); the tendency by some to decompose the system into two separate systems: social and technical. The depth of analysis for each of the sub-systems is then given different emphasis, with the focus often falling on the technical aspects (Eason, 2001) or on the social. Hollnagel (1998), for example, criticises the work on socio-technical systems for over-emphasising the context and the organisational factors. Instead, the focus should be on the interaction between the social and technical systems. A third problem highlighted by Baxter and Sommerville (2010) is related to the fieldwork. They state that although socio-technical methods, such as contextual design, emphasise users' involvement; they are fairly silent on issues such as how to identify the system stakeholders in the first place, which users to select and what level of experience in design they need. Because of all these problems, the current paper rationalises the use of grounded theory (GT) as an alternative socio-technical approach for requirement analysis. It will show how these problems can be alleviated and addressed when GT is applied to RA.

The next section provides an overview of the grounded theory method, and the third section provides literature on the requirement analysis phase of IS development and presents the related work. The fourth section shows how grounded theory techniques and procedures can be rationally used for requirements analysis. The fifth section provides a case study for its application in a real project. Section six provides discussion and implications, and the final section presents the conclusion.

2. OVERVIEW OF GROUNDED THEORY

Grounded theory has been intensively used in IS and software engineering research (Coleman and O'Connor, 2007; Georgieva1 and Allan, 2008; Hansen and Kautz, 2005; Linden and Cybulski, 2003; Sorrentino and Virili, 2005; Seidel and Recker, 2009). It is a "qualitative research method that uses a systematic set of procedures to develop an inductively derived grounded theory about a phenomenon" (Strauss and Corbin, 1990, p.24). It was originally developed by Glaser and Strauss in 1967. GT assumes that the researcher should set the literature and any predefined constructs aside when he enters the fieldwork and this procedure enables theory to emerge from the data gathered from that fieldwork.

Although different schools of thought concerning grounded theory have arisen from the subsequent disagreement between the originators themselves, the current paper does not discuss those, as they are beyond the research scope. The aim is to show how grounded theory can be applied to requirements analysis by utilising the concepts proposed by Strauss and Corbin's approach. This section presents the essential concepts, techniques and procedures of grounded theory that will be used in requirements analysis by following Strauss and Corbin's approach (Strauss and Corbin, 1990).

Theoretical Sampling: sampling in grounded theory is based on concepts shown to have theoretical relevance to the developing theory. It relates to the sampling of new data based on the analysis of that collected from the initial interviews, where the concepts that emerge constantly guide the researcher as to the nature of future data, their sources and the issues to be discussed in subsequent interviews in order to develop the categories. The initial questions for the fieldwork are based on concepts derived from literature, which provide the researcher with a starting point and a focus; later, the sampling becomes more in-depth. Strauss and Corbin explain that the sampling should focus on sampling incidents and not on persons – in other words, collecting data about what informants do in terms of action/interaction, condition and consequence of the action. The researcher continues this process until the theoretical base is saturated, at the point where no new data and ideas emerge regarding the developed concepts and categories.

Coding is the key process in grounded theory. It begins in the early stages after the first conducted interviews. Throughout the coding process, the researcher needs to be sensitive, which means being able to identify what data is significant and to assign it meaning. This sensitivity comes from experience, especially if the researcher is familiar with the subject under investigation. The literature review is another source of theoretical sensitivity, and also the expressions of the interviewees themselves, in particular, when they repeat the same phrases and concepts. The coding process comprises three steps:

Open coding is "the process of breaking down, examining, comparing, conceptualizing and categorizing data" (Strauss and Corbin, 1990, p.61). Concepts and their proprieties and dimensions are identified from data transcribed by the researchers. This can be achieved either line by line or by focusing on main ideas in sentences or paragraphs. Each code represents a word or sentence containing a meaningful idea, and a group of codes (two or more) forms a concept. A concept is an abstract representation of an event, object or action. In open coding, events, objects and actions are compared with others in terms of similarities and differences in order to give them, when similar,

the same name. The name or label that is assigned to a category should be selected logically and usually represents the data and is related to it.

Axial coding is the process of reassembling data broken down through open coding. Essentially, it is the process of relating categories to subcategories. Categories are higher in level and more abstract than concepts, and are generated by a constant comparison of the similarities and differences between such concepts. This is done by using what is called the 'paradigm model', which enables the researcher to think systematically about the data and relate them to each other. This model addresses the relationships between the categories by considering the following aspects: causal conditions, phenomenon, context, intervening conditions, action/interaction and consequences.

Selective coding is the process of integrating and refining the theory. The first step in integration is identifying the central or core category that represents the main theme of the research/phenomena. It must appear repeatedly in the data. The central category acts as a master that pulls the other categories together to form an explanatory 'whole picture' by using the paradigm model. In this step, the categories are refined at a high level of abstraction. The integration is not dissimilar to axial coding except that it is done at a higher, more abstract level of analysis, and the subcategories are linked to the core category.

Constant comparative analysis: This is a continuous process of identifying conceptual categories and their properties emerging from data by a consistent comparison of data to each other.

Conceptualisation and abstraction: GT aims to develop theories and concepts that can be generalised and applied to other situations. The generalisability of the grounded theory is partly achieved through a process of abstraction by moving from a detailed description to a higher level of abstraction; the more abstract the concepts, the greater the theory applicability.

3. LITERATURE REVIEW

System requirements are descriptions of the services provided by the system and its operational constraints. These represent the needs of customers for a system that is required to solve problems (Sommerville, 2006). The process of finding out, analysing, documenting and validating these services and constraints is called requirements engineering (RE) (Sommerville, 2006). Zave (1997) considers requirements engineering as a branch of software engineering, which is concerned with the real-world goals for, functions of, and constraints on software systems. Nuseibeh and Easterbrook (2000) pointed out that there is an important philosophical element in RE. More specifically, there are epistemological assumptions, since RE is concerned with interpreting and understanding stakeholder perceptions, concepts and goals; and ontological assumptions about the question of what can be agreed on as objectively true. They also stated that the context in which RE takes place is always a human activity system. Therefore, they referred to several social sciences such as cognitive psychology, anthropology, sociology and linguistics, which contribute to RA and provide practical techniques for eliciting and modeling requirements.

Sommerville (2006) highlighted that the term 'requirement' is not used by some in a consistent way; while some view requirement at a high-level – abstract statement of a service that the system should provide or a constraint on the system – others view it as a detailed formal definition of a system function. Hence, Sommerville distinguished between the terms 'user requirements' and 'system requirements'. The first refers to the high-level abstract requirements, which are statements written in natural language plus diagrams, describing what services the system is expected to provide and the constraints in which it must operate. In comparison, system requirements refer to the detailed description of what the system should do. Furthermore, requirements can be classified into functional and non-functional. The functional requirements for a system describe what the system should do. When expressed as user requirements, they are usually described in an abstract way. However, functional system requirements describe the system function in detail, its inputs and outputs, and exceptions. In contrast, non-functional requirements are not directly concerned with specific functions delivered by the system, rather they relate to emergent system properties such as reliability, response time and user interfaces design aspects Sommerville (2006).

Two major methodologies have been used for system development: structured analysis and design and object-oriented analysis and design (OOAD). Requirements analysis involves two main activities that are achieved by the analyst: requirements determination/ elicitation and requirements structuring (Hoffer et al., 2011). Different techniques used for requirements determination include questionnaires, interviews, observation, documents and reports, as well as other modern techniques such as joint application development (JAD) and prototyping. Analysts also use different models to structure and represent the requirements such as data flow diagram (DFD) and entity relationship diagram (ERD). In the case of OOAD, the analyst uses object/class diagrams, use case diagrams and other models (Hoffer et al., 2011).

A lot of research in IS development and the software engineering field have used the grounded theory method, as there is a widely held belief that it is a reliable method by which to elicit systems and user requirements (Coleman and O'Connor, 2007; Georgieva1 and Allan, 2008; Hansen and Kautz, 2005; Linden and Cybulski, 2003; Seidel and Recker, 2009; Sorrentino and Virili, 2005). Galal-Edeen (2005) indicated that a requirement engineer who produces a statement of system requirements is, in reality, engaged in generating 'grounded theories'. Grounded theory was originally developed and used in social sciences and was later adopted by other fields such as information systems and software engineering. One issue might arise from this inheritance to other fields (e.g. software engineering): can the grounded theory method be applied to requirements engineering by a systems analyst (SA) or a psychologist researcher (for example) to analyse requirements, supposing that he/she knows the business problem and questions? To answer this question, Carvalho et al. (2005) conducted empirical research in software engineering to generate a process model using the grounded theory method. The same gathered data were analysed by two researchers. The first researcher is a psychologist with a limited background in software engineering, but with knowledge of qualitative research methods and experience in the use of grounded theory. The second researcher is a software engineer, with a solid background in software engineering and experience in process modelling. The resulting model produced by the psychologist, however, significantly differed from that produced by an experienced process engineer using the same data.

One of the main differences in the models emphasises that modellers should not rely solely on qualitative methods to analyse process data, but rather on their experience of the research area and the technical aspects that appear in the gathered data. The psychologist was more likely to miss artifacts and activities. The notion here is that even when using qualitative research methods adopted from the social sciences, the SA should have theoretical sensitivity of the research/business problem in order to produce practical and relevant results. Chakraborty and Dehlinger (2009) state that there is a lack of systematic procedures within requirements engineering that enable the bridging between qualitative data and the final description of the system. In addition, they pointed that the focus has been on the representation of the system by UML models as an example. This leads to reduced traceability between source data (i.e., the requirements) and the final proposed models. Therefore, they proposed using grounded theory in requirements engineering to alleviate this deficiency. They provided a demonstration of how the grounded theory method can be used to interpret the requirements for an enterprise system by applying the grounded theory coding process to an illustrative example (university support system). Although the illustration was useful, Chakraborty and Dehlinger (2009) did not show how elements of grounded theory (such as theoretical sampling, theoretical sensitivity, data saturation and constant comparative analysis) can be operationalised and applied to requirement analysis, and what is the added value of its application in this context as an alternative approach to the existing requirements analysis methods. The current research takes further steps to reveal technically how the concepts of GT support the requirements analysis process. It also emphasises the point that applying GT will aid in solving some problems of the existing requirement analysis methods discussed in the introduction, such as poor communication between the end user and the development team. This is because the latter use formal notations and modelling, which make it difficult to validate and review the requirements end user. There is also, besides other problems discussed in the introduction, a lack of communication and understanding between the development team members as it is not clear how the outcomes produced by the requirement gathering team can be used by others like designers and programmers.

4. REQUIREMENT ANALYSIS USING GROUNDED THEORY

Figure 1 illustrates how grounded theory elements can be used as a technique for requirement analysis. As shown in this figure, the SA starts with a perception that there is a business problem or receipt of a request for proposal to modify the current system. The analyst starts without any pre-assumed functions or components of the required system. In fact, this is essential, as information systems probably fail because system analysts and developers assume that the requested system is similar to those already developed by them and for which they know the requirements. However, by using GT, the analysts can listen to users and remain open to accepting new and unique requirements. This is the characteristic of GT that guides an SA to start without any predefined requirements, as each system has a certain specialty. Then, the analyst interacts with the stakeholders to determine what they would like in the new system. Sampling in GT is purposive, which means that the stakeholders are selected based on their relevance and experience with the systems being developed. Recognising the right users assists the analyst in identifying the right systems requirements. This also conforms to the concept of a user-centred design, in that the analyst does not force his or her predefined requirements. Requirements are collected principally from interviews,

but possibly also from documents, observations and reports. Analysts gather the initial requirements from the first user and the gathered requirements guide him or her to discuss them with the second user, third user and so on. Perhaps after that, the analyst will return to the first user to solicit feedback regarding his or her systems' needs, as it is an iterative process.

In fieldwork, the interplay between data collection and analysis is processed simultaneously by identifying the requirements emerging from the first interviews, so that they become more specified as time progresses, since the SA validates them with the next users. At the same time, theoretical sensitivity and sampling and constant comparison between requirements (functions, processes, objects and attributes are compared with others in terms of similarities and differences in order to group similar ones together, assign a name to them and eliminate repeated ones) are taken into account, finally resulting in the data becoming saturated. The concept of data saturation also applies to RA in that the analyst continues the process of gathering data and comparing this with other data until they become saturated and no further unique ideas can be elicited. Requirements can be initially collected from a small number of stakeholders (for example five). Then, if data collected from participants number six and seven are repeated and do not include any new ideas, then the requirements gathering stop here and it can be said that the requirements are saturated. Repeating the same data during data collection guides the analyst that this requirement (function, constraint) is a priority for the system.

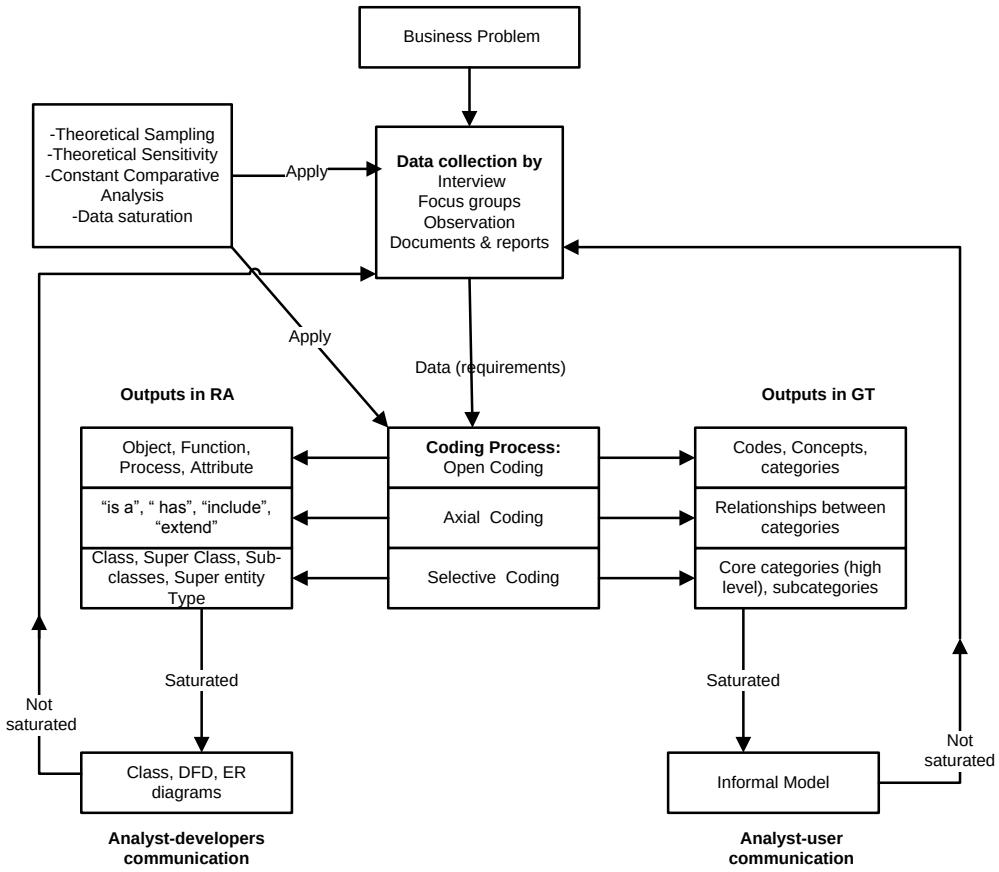

Figure 1. Using grounded theory concepts in requirement analysis

A systematic process of coding begins once the requirements have been gathered. The analysts continue to apply a constant comparative by comparing concepts that have common attributes and combining them to generate a category. This category can be a class in OOAD or a super entity type in ERD. As much as the analyst conceptualises at a higher level, he or she can generate superclasses, for example, in OOAD. The outcomes from each coding step are shown in Figure 1: codes and concepts, categories and relationships between them and categories and associated subcategories. These, ultimately, form the informal model. The corresponding outcomes in RA are shown in Figure 1. In open coding, the outcomes could be a list of functions, processes, entities, objects, attributes and classes. The outcome from axial coding is the association between classes (e.g., "is a") or association relationship between entities. The outcome of the selective coding is a refinement of the classes and entities found in open coding to a higher level, which includes super entities, superclasses and related subclasses, and the generalisation/specialisation relationships between them. The resulting categories and relationships (equivalent outcomes in RA such as classes and super classes) may not end up being fully saturated. Consequently, a second round of data collection and analysis is initiated, which leads to the developments of a new version of the model.

In qualitative research, in particular, grounded theory, the researcher is part of the research problem and is not independent. Hence, in this case, the analyst is part of the process and participates if something is missing from the user. Consequently, the role of the analyst is to complete the system requirements, as users may not always provide all of the requirements or may not focus on non-functional requirements such as performance and security, thus requiring interference from the analyst. However, this interference should come at the final stages after the users reveal all of their needs.

The resulting model from grounded theory is informal; this means that no standard notation or rules exist for drawing this model, as is the case in the ERD and DFD model (see an example of informal models in IS research-applied GT: Carvalho et al. 2005; Coleman and R. O'Connor, 2007; Georgieva and Allan, 2008). Informal models are used throughout all communication between the SA and the end user, which are based on simple language and representation understood by the end user. On the other hand, the equivalent model in RA such as UML models (e.g., class diagram) is easily created from the informal models and used in communication between the analyst and developers. Table 1 shows the possible outcomes from the grounded theory and the equivalent elements in OOAD (e.g., class/object, use case diagrams), ERD and DFD.

Table 1. Outcome from grounded theory and the equivalent elements in OOAD, ERD and DFD

Grounded Theory	OOAD	ERD	DFD
Codes (event, action, object,) concept	Object, use case, method	Entity/Entity type	Process, data store, data flow
Group of concepts (category)	Class	Entity type	
Group categories upper/general category	Super Class	Super entity type	
Relationships between categories and sub-categories (Consequences, causal conditions, action/interaction, intervening conditions)	"is a" "has" "include" "extend"	Verbs represents the association between entities	
Context (properties)	Attribute	Attribute	
Conceptualisation	Specialisation/ Generalisation	Specialisation/ Generalisation	DFD decomposing into sub-process
Data	Requirements	Requirements	Requirements
Theory/informal model	Object/class model	Data model	Process model

5. CASE STUDY

In this section a case study is provided to prove that the GT theoretical concepts can be applied to RA.

The case study is a real project aimed at developing a platform for serving a community of practice (CoP) for researchers in UAE. Communities of practice are groups of people who share a concern or knowledge or a passion for something they do and learn how to do it better as they interact regularly (Wenger, 1998). Many CoP platforms have been developed for people who have common interest in several domains such as healthcare, education, business and others. However, there is lack of CoP platforms for researchers. This project aimed to achieve that. The reason for developing a specialised CoP platform for researchers is that obviously each domain has its particular activities. For example, a group of researchers have interests and activities which are different from a group of professional physicians.

A total of seven semi-structured interviews were conducted with professors from two UAE universities, of mixed gender, active in business and IT research fields and aged between 30 and 50 years old. Based on the literature review on CoP domain, a set of questions were formulated which are open in nature. However, the current research did not rely solely on these questions; other issues and ideas which emerged during the interview were also considered. This approach was to enable the participants to reveal their needs freely without any direction. Collected data were analysed, which resulted in the informal model depicted in Figure 2 (see Halaweh et al., 2011, for more details). The oval shape represents the category and the highlighted oval shape represents the core category. Boxes represent sub-categories, a line between categories, and sub-

categories represents relationships which can be "association" or "consist of" or "has" relationship. The figure shows seven main categories. Additionally, it was found that collaboration is the core category as revealed by the data analysis. It represents the key need of the CoP researchers within UAE. This category includes three sub-categories: content management, communication and debating. Content management has also three sub-categories.

In grounded theory every piece of data gathered is coded. Each code represents a significant meaning. The code represents an object or action or process or concept, and similar and related codes can be refined and combined to form a category. For example, when one of the stakeholders was asked how he would like to participate in the activities of a community of practice (CoPs), he stated:

Sharing research paper with others, like if one of the Faculty found a good paper he can post it and comment on it, and there could be a part to summarise the paper to save our time on reading, so when you access to the paper you will find the summary there. There is a tool called delicious to share links, you can favorite your best links then you can share them, also instead of sharing the links locally you can share them within the Internet and you can create a category of links.

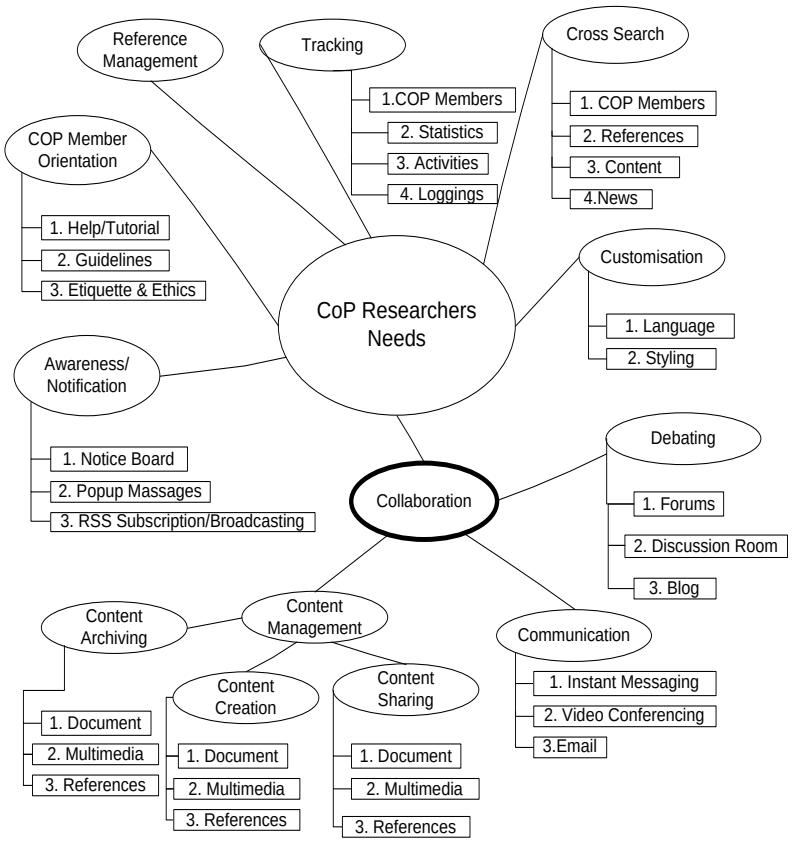

Figure 2. Informal Model of features (categories) of CoP for researchers

From the above short quote, significant keywords were highlighted through underlining, which represents a certain code (object, action). It may be elicited from this excerpt that sharing (i.e. action) is the main function that the participant needs to have in

CoP. Furthermore, he specified two types of sharing including links and research papers (i.e. objects). Throughout this process and continuous comparative analysis between codes obtained from other interviews, it appeared that there were similarities in some codes, which were later grouped together under one category. Data revealed by earlier participants were investigated by the next participants so they were constantly compared with other data to validate the requirements. For example, other participants referred to sharing multimedia objects as one of the activities that they would like to see in the CoP, so the current researcher formed a category and named it 'Content', which has three different types: document (mentioned as research paper in the above excerpt), multimedia and links (URL references). The researcher continued this process until all possible codes and categories were identified, and this resulted in the open coding stage being completed. Figure 2 shows all these categories in an informal model.

As mentioned before, the outputs of GT can be translated into RA outputs. Figure 3 shows how part of the informal model is converted into a class diagram. For example, the categories – collaboration, content, communication and debating (from Figure 2) – are labelled as classes in Figure 3. As an example, the class "Content" has attributes and functions. One of the functions is "share ()" and this was revealed by the data shown in the above excerpt (e.g. sharing research paper, share links).

The next stage in GT is axial coding, where relationships between categories are identified. For example, there is a relationship between collaboration and content, that is, any collaboration involves content. This relationship can be represented as association in the class diagram as shown in figure 3. Another example of relationships is between the class "Content" and the sub-classes document, multimedia and links. As shown in Figure 3, there is inheritance/generalisation relationship between the super class "Content" and the sub-classes: document, multimedia and links. The sub-classes inherit the functions (for example, the function share ()) and attributes of the super classes. Finally, in the selective coding, the researcher elicited the core category that was mentioned frequently by the stakeholders, whether this was implicitly or explicitly, which also represents the main function/service of the CoP. It was found that the core category is collaboration and this was represented as super class in the class diagram. The full class diagram obtained from the informal model can be used later by programmers.

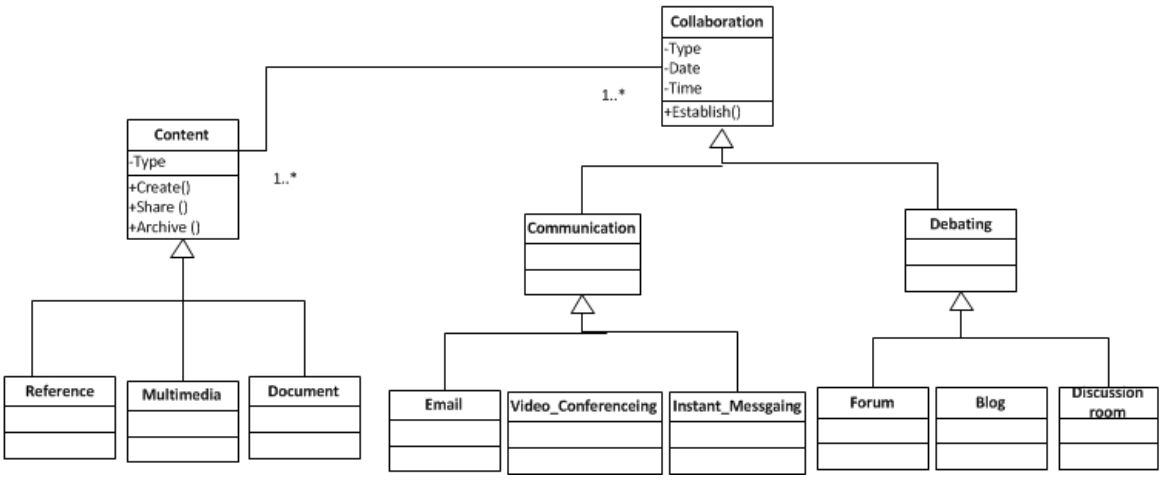

Figure 3. Class diagram translated from the GT informal model

6. DISCUSSION AND IMPLICATIONS

Based on the application of GT in the project, it was found that there are strengths of using GT as a technique for requirements analysis.

Firstly, GT will assist the analyst in identifying the non-technical aspects associated with developing the system. The reason is that the nature of GT is used to understand the organisational and social phenomena. This may not be considered by analysts who do not apply GT as they focus rather only on the technical systems requirements. Analysts can advise the decision-makers and management about any potential problems associated with introducing the system. This may also help to specify appropriate system features and functions, installation and training policy. In addition, this can guide the development team to design a system that can overcome some of the social and organisational problems. The use of GT in this project revealed some social and cultural issues, which are relevant particularly to the UAE context such as language, weather and gender isolation in education. Some of the participants needed to have an option to change the platform language to Arabic. This is applied to the platform menus and interfaces, the created contents and the search options. However, because this issue can be added as a feature to the platform, it was represented in the informal model (under the Customisation category). However, there are other issues which are not provided by the systems as a service or feature. For example, the weather of UAE is very hot especially in the summer when the temperature sometimes reaches 50^{o}C. The weather and geographical position of UAE make the mobility of people difficult during the summer time. This motivates participants to have a platform that enables them to collaborate online without the need to meet face to face. In addition, some participants argued that there is a justifiable need for CoP of researchers in UAE for gender mix constraint. The UAE governmental educational system segregates students based on gender at universities and this also requires isolating professors. Females cannot easily meet males face to face for religious and cultural reasons. Therefore, this limits the research collaboration and discussion. As highlighted earlier, the use of existing traditional socio-technical design methods sometimes falters because of the tendency to focus the in-depth analysis on each of the sub-systems separately, namely the technical aspects of the system or on the social systems. However, in grounded theory all data gathered are coded and treated equally when analysed without separation into technical or social aspects; the pieces of data are constantly compared with each other. Therefore, the resulting informal model is considered a socio-technical model; neither solely focusing on the context, nor purely technical. It is worth mentioning that one of the resulting categories from the gathered data related to the social aspects is not represented in the informal model as it is not a service provided by the system that needs to be validated by the user or used by another development team. The category includes contextual and cultural issues which are relevant particularly to the UAE context as mentioned before such as weather and gender isolation in education.

Secondly, the resulting informal model from GT can be used as a communication tool between the stakeholders and the analyst (or development team) to validate the requirements which solve the problem found earlier related to difficulty understanding the formal models and notations by the end user. The real world represented by the informal model is closer to the end users, and they like visualisation as it gives a picture about the system components and boundaries. At the same time, it is not a formal

analysis model (e.g. class diagram, DFD and ERD), which may require some effort from the end user to understand its notation and rules.

Thirdly, another improvement that GT made to RA is that the transition from the elicited requirements to formal models is clear according to grounded theory as shown in the previous section. Analysts will find it easy to convert the resulting model of GT into formal models such as a class diagram as shown in Figure 3 and thus produce useful output for designers and programmers. The GT model works as an intermediary medium to facilitate moving from a large amount of detailed data to standard analysis models. This solves the problem that was mentioned before (Baxter and Sommerville, 2010) in that practitioners of ethnography and contextual design fail to deliver products that can be used by others because they essentially stop after collecting data.

Next, following the GT procedures will assist in gathering complete requirements, and building a system based on user requests, which, ultimately satisfies the users' needs. GT supports the concept of a user-centred design, as the requirements are user-based driven, and no predefined requirements are forced. Furthermore, the core category(s) assists the analyst in specifying the functional requirements. The core category represents data that is repeated many times, which refers to the main system needs. It also represents an agreement on indispensable functions, those without which the system would be incomplete. In the current case study, the core category was "collaboration", which is the central point of CoP purpose, namely collaboration between a group of researchers who have the same interests. In addition, applying the conceptualisation technique by moving from the descriptive details to more abstract concepts assists in defining the super and sub classes in class diagrams, for example. GT also guides the analyst based on theoretical sampling, in order to identify initial participants based on their relevance and experience with regard to the systems being developed. The data, moreover, collected from those participants guide the search for other requirements. In other words, the identified preliminary requirements specify whom to interview next and what requirements to look for. Participants are not selected randomly, rather there is more focus on the actual users who will benefit from and interact with the system. Also, data are continuously gathered until saturated, meaning no new ideas can emerge, or data are repeated frequently. Data saturation will assist the analyst in deciding when to stop gathering requirements or direct him to identify new sources of data if there is repetition in the data. All these issues were not clearly specified by the current socio-technical methods as mentioned before.

Although there are several advantages of using grounded theory for requirement analysis, there are some shortages. More specifically, based on the current project, using GT did not help to find much technical – low level – detailed functions of the CoP platform. The resulting categories form high level abstraction functions of CoP platform, which can be labelled as user requirements not system requirements as pointed out by Sommerville (2006).

7. CONCLUSION

This research provided a proposal for using GT as an alternative technique for requirements analysis. It demonstrated through a case study the application of grounded theory procedures and techniques in an exemplary project. Although the current paper has presented logical justification for its use, and provided real example for its application, there is the possibility of failing to determine the technically detailed

functions. Nonetheless, GT can be used as supportive technique to improve the communication between the analyst and the user, to improve communication between the development team members as it is easy to transform the informal models of GT into standard models, thus improving the communication between analyst and developers. It can also be used as a socio-technical technique for requirements analysis, in order to understand better users' needs and to address the nontechnical issues related to information systems development.

One of the limitations of this research is that the current application of GT to RA did not produce low level technical functions of the system. Other future examples of application GT in different types of systems might succeed in producing low technical detailed functions.

This paper demonstrated through example how the outputs of GT can be translated into a class diagram. Future research might also provide evidence for the translation of GT outputs into other types of models such as ERD and DFD.

REFERENCES

Allan, G., (2003). A critique of using grounded theory as a research method, *Electronic Journal of Business Research Methods*, 2(1), 1-10.

Al-Rawas, A. and Easterbrook, S., (1996). Communication Problems In Requirements Engineering: A Field Study, *Proceedings of the First Westminster Conference on Professional Awareness in Software Engineering*, Royal Society, London, 1-2 February 1996.

Alter, S., (2006). The Work System Method, Connecting People, Processes, and IT for Business Results, 1st ed., Work System Press, Larkspur, CA.

Baxter, G., Sommerville, I., (2010). Socio-technical systems: From design methods to systems engineering, *Interacting with Computers*, 23, 4-17.

Berg, M., and Toussaint, P., (2003). The mantra of modelling and the forgotten powers of paper: a sociotechnical view on the development of process-oriented ICT in health care, *International Journal of Medical Informatics*, 69(2-3), 223-234.

Beyer, H., & Holtzblatt, K. (1999). Contextual design. *Interactions, 6*(1), 32-42.

Carvalho, L. Scott, L. and Jeffery R., (2005). An exploratory study into the use of qualitative research methods in descriptive process modeling, *Information and Software Technology*, 47, 113–127.

Checkland, P., (1981). Systems thinking, systems practice. Chichester, UK: Wiley.

Chakraborty S. and Dehlinger J., (2009). Applying the Grounded Theory Method to Derive Enterprise System Requirements, *10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*, 27-29 May 2009, Daegu, Korea, 333-338.

Clegg, C., (2000). Sociotechnical principles for system design, *Applied Ergonomics*, 31, 463-477.

Coleman G. and O'Connor, R., (2007). Using grounded theory to understand software process improvement: A study of Irish software product companies, *Information and Software Technology*, 49, 654–667.

Dekker, S. W. A., Nyce, J. M., & Hoffman, R. R. (2003). From contextual inquiry to designable futures: What do we need to get there?, *IEEE Intelligent Systems,* 18(2), 74-77.

Doherty, N.F., and King, M., (2005). From Technical to Socio-Technical Change: Tackling the Human and Organizational Aspects of Systems Development Projects, *European Journal of Information Systems*, 14, 2005, 1-5.

Eason, K., (2007). Local sociotechnical system development in the NHS national programme for information technology, *Journal of Information Technology*, 22(3), 257-264.

Galal-Edeen, G. H., (2005). Information Systems Requirements Engineering: An Interpretive Approach, *The Egyptian Informatics Journal*, 6 (2), 154-174.

Glaser B. and Strauss, A., (1967). The discovery of Grounded Theory. Chicago: Aldine.

Georgieva1, S. and Allan, G., (2008). Best Practices in Project Management Through a Grounded Theory Lens, *Electronic Journal of Business Research Methods*, 6(1), 43-52.

Geisser, M.; Hildenbrand, T., (2006). A Method for Collaborative Requirements Elicitation and Decision-Supported Requirements Analysis. In: Ochoa, S.F. und Roman, G.-C. (eds): *IFIP Int. Federation for Information Processing, Advanced Software Engineering: Expanding the Frontiers of Software Technology*, 108-122.

Halaweh, M., ALmourad, B. and Miniaoui, S., (2011). Identifying the needs of Community of Practices (CoPs) for Researchers within UAE, *European, Mediterranean and Middle Eastern Conference on Information Systems (EMCIS)*, 30-31 May 2011, Athens, Greece.

Hansen Bo H. and Kautz, K., (2005). Grounded Theory Applied –Studying Information Systems Development Methodologies in Practice, *Proceedings of the 38th Hawaii International Conference on System Sciences*, 3-6 January, 2005, Big Island, HI, US.

Hoffer, j., Joey G., Valacich, J., (2011). Modern Systems Analysis and Design, 6th edition, Prentice Hall.

Hollnagel, E. (1998). The cognitive reliability and error analysis method, Oxford, UK: Elsevier.

Linden T. and Cybulski, J., (2003). Application of Grounded Theory to Exploring Multimedia Design Practices, *7th Pacific Asia Conference on Information Systems*, 10-13 July, 2003, Adelaide, South Australia.

Madsen, S., and Vidgen, R., (2009). A pragmatic approach to IS development and socio-technical evaluation, *ECIS 2009 Proceedings.*

Martin, D., and Sommerville, I. (2004). Patterns of cooperative interaction: linking ethnomethodology and design. *ACM Transactions on Computer-Human Interaction (TOCHI),* 11(1), 59-89.

Mumford, E., (1987). Sociotechnical Systems Design; Evolving theory and practice. Bjerknes G. & Ehn P. & Kyng M. (eds.) *Computers and Democracy*: *A Scandinavian Challenge*, Avebury, pp. 59-76.

Mumford, E., (2000). A Socio-Technical Approach to Systems Design, *Requirements*

Engineering, 5, 125-133.

Nuseibeh, B. and Easterbrook, S., (2002). Requirements engineering: a roadmap, *In Proceeding of the IEEE International Conference on Software Engineering* (ICSE), 35–46, 2000.

Reddy, M., Pratt, W., Dourish, P. and Shabot, M., (2003). Socio-technical requirements analysis for clinical systems, *Methods Inf Med*, 42, 437-444.

Seidel S. and Recker, J. (2009). Using Grounded Theory for Studying Business Process Management Phenomena, *17th European Conference on Information Systems*, 2009.

Sommerville, I., (2006). Software Engineering, Addison Wesley, 8th edition.

Sommerville., I., and Dewsbury, G., (2007). Dependable domestic system design: a socio-technical approach, *Interacting with Computers*, 19, 438-456.

Sorrentino M. and Virili, F., (2005). "Web Services System Development: a Grounded Theory Study", *18th Bled eConference eIntegration in Action*, 6-8 June, 2005, Bled, Slovenia.

Strauss A. and Corbin, J., (1990). Basics of Qualitative Research: Grounded Theory Procedures and Techniques, SAGE Publication, London.

Vicente, K. (1999). Cognitive work analysis, Mahwah, NJ: LEA.

Vijayan J. and Raju G., (2011). A New approach to Requirements Elicitation Using Paper Prototype, *International Journal of Advanced Science and Technology*, 28.

Waterson, P. E., Gray, M. T., and Clegg, C. W., (2002). A sociotechnical method for designing work systems, *Human Factors*, *44*(3), 376-391.

Wenger E., (1998). Communities of Practice: Learning Meaning and Identity, Cambridge University Press.

Zave, P., (1997). Classification of Research Efforts in Requirements Engineering, *ACM Computing Surveys*, 29(4): 315-321.

