Using Grounded Theory as a Supportive Technique for System Requirements Analysis

Mohanad Halaweh
College of Information Technology
University of Dubai
Dubai, UAE
mhalaweh@ud.ac.ae

***Abstract*—Requirements analysis is a key phase in information systems development. During this phase, system analysts use different techniques and methods to determine and structure the systems requirements. In this paper, the author rationalises the use of grounded theory as a technique for requirements analysis. It aims to establish theoretically that applying grounded theory procedures and techniques will strengthen and add value to the analysis phase.**

Keywords-grounded theory; IS devdelopment; requirements analysis; requirements engineering

I. INTRODUCTION

Requirements analysis (RA) is a key phase in information systems (IS) development. During this phase, system analysts use different techniques and methods to determine and structure the systems requirements. In this paper, the author rationalises the use of grounded theory (GT) as a technique for requirements analysis. It aims to establish theoretically that applying grounded theory procedures and techniques will strengthen and add value to the analysis phase.

The next section provides an overview of the grounded theory method, and the third section briefly explains the requirement analysis phase of IS development. The fourth section presents literature review. The fifth section rationalises the use of grounded theory for requirements analysis and explains how it adds value and strengthens the analysis stage, and the final section presents the conclusion.

II. OVERVIEW OF GROUNDED THEORY

Grounded theory has been intensively used in IS and software engineering research [1][2][3][4][5][6]. It is a "qualitative research method that uses a systematic set of procedures to develop an inductively derived grounded theory about a phenomenon" [7] (p.24). It was originally developed by Glaser and Strauss in 1967 [8].

Although different schools of thought concerning grounded theory have arisen from the subsequent disagreement between the originators themselves, the author does not discuss those, as they are beyond the research scope. Furthermore, the aim is to show how grounded theory can be applied in requirements analysis by utilising the concepts proposed by Strauss and Corbin's approach, and avoid the current debate concerning the theory itself. 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 [7].

- *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 initially 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 (i.e. data gathered previously), which provides the researcher with a starting point and a focus; later, the sampling becomes more in-depth. Strauss and Corbin [7] explain that the sampling should focus on sampling incidents and not 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*, where no new data and ideas emerge regarding the developed concepts and categories.
- *Coding* is the key process in grounded theory [7]. It begins in the early stages after the first interviews for data collection. This process comprises three coding steps:

1. **Open coding** is "the process of breaking down, examining, comparing, conceptualizing and categorizing data" [7] (p.61) by which concepts

 ISBN:978-1-61208-114-4

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 for a category should be selected logically and usually represents the data and is related to it. A reading of the literature gives the researcher an initial set of concepts that can be used. However, researchers should not be constrained by these concepts; rather, they should focus on the words and phrases used by the participants themselves. It is in this way that names are assigned to categories [7].

2. **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.*

3. **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.

- Through the coding process, two analytical techniques are used. The first is *constant comparative analysis*, which is a continuous process of identifying conceptual categories and their properties emerging from data by a consistent comparison of that data. The researcher needs to be *sensitive*, which means being able to identify what data is significant and to assign it a 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*, as are the expressions of the interviewees themselves, in particular, when they repeat the same phrases and concepts. The other technique is the *asking of questions*. Once the researcher names the concept (event, idea, action and incident), he or she asks questions such as what an object is and what it represents.
- *Conceptualisation and abstraction*: Grounded theory 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 [7].

III. REQUIREMENT ANALYSIS

Many methodologies are used for IS development. Two major methodologies have been used for system development: structured analysis and design and object-oriented analysis and design (OOAD). Generally, regardless of which methodology is used, the core phases for system development are analysis, design, implementation and testing. The purpose of requirements analysis is to understand the business problem and the customer (i.e. organisational) needs of the proposed system. The New York State Project Management Guidebook [9] pointed out:

> "The primary goal of [requirements analysis] is to create a detailed functional specification defining the full set of system capabilities to be implemented, along with accompanying data and process models illustrating the information to be managed and the processes to be supported by the new system".

Requirements are descriptions and specifications about the functions (what the system should do) and proprieties of the system. In fact, accuracy and completeness of the requirements affect the quality of the final developed system. A systematic process for requirements analysis is also known as requirements engineering (RE).

Requirements analysis involves two main activities that are achieved by the analyst: requirements determination/ elicitation and requirements structuring. Different techniques used for requirements determination include questionnaires, interviews, observation, documents and

 ISBN:978-1-61208-114-4

reports, and 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. Various techniques and approaches were proposed for requirement analysis such as goal-oriented/ goal-driven requirements analysis, scenario-based requirements analysis, inquiry based requirement analysis, and ontology based requirements analysis. In this paper, we propose a supportive approach for requirements analysis using GT.

V. RELATED WORK

Many research in IS development and the software engineering field has 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 [1][2][3][4][5][6]. Galal-Edeen [10] 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 emerges from this inheritance to other fields: Can the grounded theory method be applied in requirements engineering by a systems analyst (SA) or a psychologist researcher (for example) to analyze the requirements, supposing that he/she knows the business problem and questions?

To answer this question, Carvalho et al. [11] 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 emphasizes that modelers should not rely solely on qualitative methods to analyze 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 problem in order to produce practical and relevant results. Chakraborty and Dehlinger [12] 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, 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 (open, axial and selective coding) on an illustrative example (university support system). Although the illustration was useful, the authors did not highlight 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 or supportive technique to the current requirements analysis methods and techniques. The current research takes further steps to technically reveal how the concepts of GT support the requirements analysis process by providing a methodology.

VI. 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 or maintain the current system. The analyst starts without any pre-assumed functions or components of the required system. In fact, this is essential, as many information systems fail because system analysts and developers assume that the requested system is similar to ones that were already developed by them and that 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 users to find out what they would like in the new system. The users are selected for their relevance to the business problem by applying theoretical sampling. Sampling is theoretical based, which is helpful for identifying involved users who will interact and use the system. Identifying the right users assists the analyst in identifying the right systems requirements. This also supports the concept of a user-centered design, in that the analyst does not force his or her predefined functions/features/requirements. Requirements are collected principally from interviews, but possibly also from documents, observations and reports. The concept of prototyping for requirements gathering conforms to the concept of theoretical sampling. Analysts gather the initial requirements from the first user and the gathered requirements guide him or her to discuss them with the second user, and 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

 ISBN:978-1-61208-114-4

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. That is the point at which no new requirements emerge. Repeating the same data during data collection advises the analyst that this requirement (function/attribute, process) is a priority for the system. In addition, this information indicates to the analyst that the data collected from the users is saturated.

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 conceptualizes at a higher level, he or she can generate superclasses. 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 relationship between entities. The outcome from 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 generalization and specialization relationships between them. The resulting categories and relationships (equivalent outcomes in RA such as classes and super entities) 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.

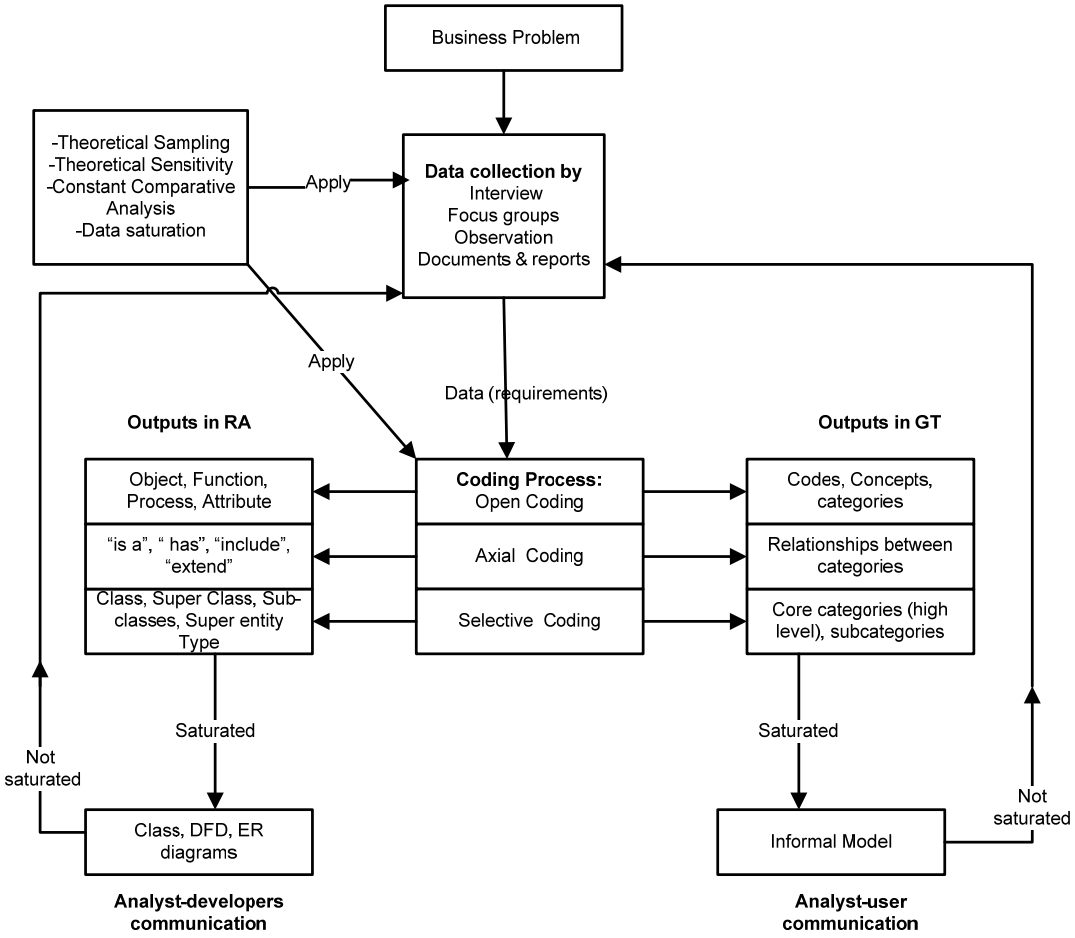

Figure 1. Using grounded theory concepts in requirement analysis

 ISBN:978-1-61208-114-4

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: [4]; [5] [13]). 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 outcomes from the grounded theory and the equivalent elements in OOAD (e.g., class/object, use case diagrams), ERD, and DFD.

TABLE I. 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	Supper 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

Strengths of Using Grounded Theory as a Technique for Requirements Analysis:

- Analysts will find it easy to convert the resulting model of GT into a data model (ERD), process model (DFD) and other models. The GT model works as an intermediary medium to facilitate moving from a large amount of detailed data to standard analysis models.
- Models resulting from GT can be used as a communication tool between the analyst (development team) and end users. The real world represented by the informal model is closer to the end user, and they like visualisation. At the same time, it is not a formal analysis model (DFD, ERD, and class diagram), which may require some effort from the end user to understand its notation and rules.
- 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-centered design, as the requirements are user-based driven, and no predefined requirements are forced.
- GT will guide the analyst based on theoretical sampling to identify the relevant users who will interact and use the system, and who will explain the system requirements.
- Applying the conceptualisation technique by moving from the descriptive details into more abstract concepts assists in defining the system data and functions. This is also helpful in the case of using OOAD to specify the super and sub classes in class diagrams, and the objects in object diagrams that represent (instances) detailed data and a high abstract concept (category) that represents the class. The linkage between the categories and its subcategories specify the inherent relationship between the parent and child classes (inheriting classes).
- 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.
- 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 about the indispensable functions, without which the system would be incomplete.

 ISBN:978-1-61208-114-4

- All traditional techniques of data gathering are combined into one method, as GT employs a group of techniques: observation, interviews, focus groups and documentation of current systems. In addition, any gathered text is considered input to the theory/model.
- GT will assist the analyst in identifying the non-technical aspects associated with developing the system, such as user resistance change, and political and power issues emerging as a result of introducing the system within the organisation. The reason is that the nature of GT is used to understand the organisational and social phenomenon. This may not be considered by analysts who do not apply GT as their focus, rather focusing 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 an appropriate system installation and training policy. In addition, this can guide the development team to design a system that can overcome some organisational problem.

VII. CONCLUSION AND FUTURE WORKS

This research provides a concept for using GT as a supportive technique for requirements analysis. Although the author has presented logical justification for this method, this conceptual proposal must be validated by a real example. This would be interesting work for future research to apply the methodology illustrated in Figure1. Practically, this paper proposes applying GT as an effective approach for requirements analysis by showing the strengths of its application.

REFERENCES

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

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

[3] Bo H. Hansen and K. Kautz, "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.

[4] G. Coleman and R. O'Connor, "Using grounded theory to understand software process improvement: A study of Irish software product companies", Information and Software Technology, vol. 49, 2007, pp. 654–667.

[5] S. Georgieva1, and G. Allan, "Best Practices in Project Management Through a Grounded Theory Lens", Electronic Journal of Business Research Methods, vol. 6, no. 1, 2008, pp. 43-52.

[6] S. Seidel and J. Recker, "Using Grounded Theory for Studying Business Process Management Phenomena", 17th European Conference on Information Systems, 2009.

[7] A. Strauss and J. Corbin, Basics of Qualitative Research: Grounded Theory Procedures and Techniques". SAGE Publication, London, 1990.

[8] B. Glaser and A. Strauss, The discovery of Grounded Theory". Chicago: Aldine, 1967.

[9] The New York State Project Management Guidebook, Release 2, 2003. The New York State Office for Technology. Retrieved 27 July, 2010, from www.cio.ny.gov/pmmp/guidebook2/SystemReq.pdf

[10] G. H. Galal-Edeen, "Information Systems Requirements Engineering: An Interpretive Approach", The Egyptian Informatics Journal, vol. 6, no. 2, 2005, pp. 154-174.

[11] L. Carvalho, L. Scott, and R. Jeffery "An exploratory study into the use of qualitative research methods in descriptive process modeling". Information and Software Technology, vol. 47, 2005, pp. 113–127.

[12] S. Chakraborty and J. Dehlinger "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, pp. 333-338.

[13] G. Allan, "A critique of using grounded theory as a research method". Electronic Journal of Business Research Methods, vol. 2, no.1, pp. 1-10, 2003.

 ISBN:978-1-61208-114-4