COMMUNICATION CHALLENGES IN REQUIREMENTS ELICITATION AND THE USE OF THE REPERTORY GRID TECHNIQUE

CHRISTOPHER J. DAVIS
University of South Florida Saint Petersburg
Saint Petersburg, Florida 33701

ROBERT M. FULLER
The University of Tennessee
Knoxville, Tennessee 37996

MONICA CHIARINI TREMBLAY
University of South Florida
Tampa, Florida 36920-6150

DONALD J. BERNDT
University of South Florida
Tampa, Florida 36920-6150

ABSTRACT

Requirements elicitation is a central and critical activity in the systems analysis and design process. This paper explores the nature of the challenges that confront analysts and their clients during requirements elicitation. A review of the literature highlights communication as a persistent locus of concern among systems analysts, users and procurers. The paper presents a classification of communication challenges that arise during the requirements elicitation process.

Empirical evidence from a brief case study is used to illustrate the scope and impact of these communication challenges and to present a complementary approach to requirements elicitation. The paper introduces the Repertory Grid technique as a means to ameliorate some of the communication issues that persist, particularly in projects where information systems support specialized work.

The paper is written in the form of a case tutorial, providing insight into the contribution of the Repertory Grid technique to requirements elicitation.

Keywords: systems analysis and design; requirements elicitation; communication challenges.

INTRODUCTION

Research published in both academic and practitioner journals highlights the ongoing challenges of systems analysis and design. Despite years of practice and a host of analysis and design methodologies, tools and techniques, developed systems frequently fail to achieve the functionality desired by their users. This paper explores the nature of these difficulties, paying particular attention to the requirements elicitation phase of systems analysis and design. Our research highlights the need for effective collaboration in both the analysis and design and user communities. Although this important issue has been identified in many previous studies (3, 5, 7, 13, 17, 25, 29, 31) the modes of use and development of newer information technologies such as application service provision (2), enterprise systems (14), internet portals (21) and on-line analytic processing (17), to name but a few, highlight the increasing pace of change faced by both analysts and user communities in the very dynamic environment of software development in the 21st century. Our review of the literature shows that some significant challenges persist. Communication emerges as the principal locus of the issues and concerns identified in previous research into requirements elicitation (5, 9, 39). The research question addressed by this paper is "How can user-analyst communication during requirements elicitation be made more effective?"

It is well known that requirements elicitation is one of the most important steps in systems analysis and design. The difficulties encountered in accurately capturing system requirements have been suggested to be a major factor in the failure of 90% of large software projects (37). The ability to accurately elicit and portray user needs earlier in systems development as well as the ability to elicit evolving needs is necessary to reduce the larger costs associated with error correction later in systems development (8).

The primary success factor of requirements elicitation is that requirements meet end user needs. This outcome is difficult to achieve because users often have trouble identifying and articulating their needs (22, 29) and because those needs often change as a result of system implementation (4). This difficulty is compounded for newer technologies such as data warehouses (17) because requirements continue to evolve over time as users become familiar with the systems and their needs for information change. For these technologies, system requirements are a moving target (19, 29). Over time, challenges arise from the simultaneous evolution of the technology and of the users' requirements (19). For these reasons, calls for effective user involvement in requirements elicitation continue (18). Effective requirements elicitation depends upon the ability of users and analysts to understand and appreciate one another's worlds. This represents a significant, but not insurmountable, challenge which we explore in this paper. The paper begins by classifying the communication, negotiation (6, 17) and learning (12, 19, 22) that arise during the requirements elicitation process. The basis for classification draws on previous research in this area (5, 7, 17, 32, 36) and personal construct theory (12, 15, 38). The classification is set up in Figure 1 and used to introduce the Repertory Grid (RepGrid) technique (10, 11, 12) as a means to facilitate communication and negotiation during requirements elicitation.

The potential benefits of this technique are demonstrated using empirical data from a pilot study of the implementation of a complex data warehouse with on-line analytic processing
(OLAP) functionality (17). The remainder of the paper is written in the form of a case-based tutorial in order to explain the RepGrid technique and to highlight its contributions to the requirements elicitation process.

COMMUNICATION CHALLENGES IN REQUIREMENTS ELICITATION

Requirements elicitation involves end users and analysts interacting to identify and ‘capture’ the data and processes that will make up the eventual system. User-analyst communication is an important part of requirements elicitation, but communication styles and techniques most readily associated with requirements elicitation – interviews and questionnaires, for instance – are rarely sufficient to elicit the whole range of requirements (5, 40). The use of such standard ‘instruments’ in any user-analyst exchange introduces the potential for errors of omission that arise as a consequence of the analyst’s difficulty in elicitating system requirements that are outside the instrument’s scope. Equally, such standard tools and techniques limit the user’s capacity to fully identify and articulate their needs (26, 36). Figure 1 depicts these difficulties in four quadrants - each a potential communication challenge that arises from differences in the domain experiences of the user and analyst. Figure 1 borrows from the Johari Window (21, 34) and Personal Construct Theory (12, 15, 29, 38) to classify these challenges. Variations in the way individuals construct, interpret and realize opportunities (e.g., system requirements), problems or other issues can impact the effectiveness of communication between user and analyst (40). The model highlights this variability and provides a method to compare established and potential alternative requirements elicitation techniques (14, 31, 33). The model draws attention to the contribution that techniques relying on the constructive alternativist epistemology of Personal Construct Theory (PCT) such as RepGrid (15, 32, 34) can make to the process of requirements elicitation.

FIGURE 1
A Typology of Communication Challenges in Requirements Elicitation

Analyt

Between the user and analyst, there is common area (a) where requirements are known by both parties – what we call conspicuous requirements. These requirements are known by both the user and analyst due to their shared prior experiences in the domain of interest. For these requirements, simple questioning techniques for requirements verification can be used. However, requirements outside this shared area represent a number of challenges to both the analyst and the user.

Firstly, there are potential requirements (b) known by the analyst that are not known by the user. These potential requirements come about due to the unique experience the analyst has in the system domain that is not shared by the user. Most traditional methods and techniques in requirements elicitation are designed to increase the size of the shared area (a) by applying the analyst’s experience and skills to identify patterns (3) or common system requirements, effectively enlarging area (a) downward. For these potential requirements, structured interviews and other questioning techniques are generally used (8). This application of the analyst’s experience in requirements elicitation seeks to exploit the opportunity to reuse some previously derived design artifact (22).

Secondly, there are potential requirements (c) that are known by the user, but not known by the analyst. These potential requirements come about due to the unique experience the user has in the system domain that is not shared by the analyst. Despite the trend toward domain expertise among systems analysts who work within user communities, these potential requirements still pose a problem in the various specialist fields of knowledge work (9, 15). Unless the user – typically an expert of some kind - can identify and articulate these requirements, they may go unidentified and unshared with the analyst, limiting the functionality of the system. To identify

Special Issue 2006 Journal of Computer Information Systems 79

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
these requirements and increase area (a) to the right, techniques such as directed questions, decision maps, and what-if scenarios attempt to get at the information that is known by the user (5, 12). These techniques generally assume that the analyst can ask the questions and prod for more information, and that the user can understand and answer the questions. It hoped that through the use of these techniques, the user's recognition of unknown (to the analyst) needs is triggered, providing additional requirements.

Finally, there are potential requirements (d) that are not known about by the analyst to ask, and not known by the user to request. These requirements are outside the immediate experience of both the user and analyst and represent opportunities arising from completely new concepts. Standardization and emphasis on design imperatives such as componentization and re-use mean that the requirements to exploit such opportunities are often neither realized nor 'captured' during typical requirements elicitation, but may be realized later, for example, when the system has been implemented and is in use. Such unrealized requirements and often referred to euphemistically as "lessons learned." To successfully elicit these requirements entails a process of mutual learning or co-discovery (22, 29) that can result in design innovation and learning for both the analyst and user.

The majority of requirements elicitation techniques fail to address the less conspicuous and often more tacit requirements, priorities, and issues that analysts do not know to ask about and that users do not or cannot readily identify and articulate. Traditional techniques are unable to fully diagnose how such contextual issues will affect system requirements, system development, and system evolution (3, 12). Furthermore, analysts need unbiased, systematic approaches during communication to assist users in identifying and articulating needs (22, 26, 30). To overcome the limitations and perceptual biases of traditional requirements elicitation approaches, the concept of user-centered analysis - the process of 'capturing' requirements from the user's point of view (25) - has frequently been promoted as a means to achieve a more comprehensive understanding of end user system needs.

Despite the promotion of user-centered analysis, contemporary requirements elicitation continues to present something of a paradox: information systems are victims of their own success. Information systems have become so ubiquitous that users expect sophisticated and reliable support for a wide range of business activities. The paradox arises from the penetration of information systems into an ever-widening range of specialist uses - such as medical informatics, forensic science and increasingly complex financial services. Advances in requirements elicitation that have improved both communication and productivity in the process of developing generic business applications (4, 8) can lead to reliance on language and concepts that arise from a perception of contemporary enterprise contexts as homogenous. However, while helpful in the development and maintenance of commonly used business applications, such as perception masks the complexity and variation of operations outside the generic production, sales and accounting functions. We argue that dependence on standardized languages and techniques can adversely affect the communication process, particularly in the elicitation of requirements for these newer information technologies.

In the remainder of this paper we show how we adapted the Repertory Grid (RepGrid) technique to support requirements elicitation (15). RepGrid provides a structure for gathering data (9, 11, 12, 38) where the specific content of the analysis instrument is generated entirely by the user. RepGrid provides only structure and form to user - analyst communication but without introducing any standard questions, patterns, pre-conceptualization or other biases. Requirements can be initially defined in the users' own language, perhaps using terms specific to the context of use. User-analyst discussion develops as the technique progresses, giving rise to early shared understanding of context-specific concepts, issues and concerns (16). We have found that this produces a more robust conceptual system design (9) and reduces the likelihood that requirements will be misunderstood, obscured or inappropriately prioritized - common complaints associated with the use of conventional interview schedules and questionnaires (7, 36).

RepGrid provides a unique set of structures to enable the user to share their experience with the analysts, allowing potential requirements that the user may have difficulty identifying and articulating (area (c) in Figure 1) to be uncovered. RepGrid has potential for mutual discovery (area (d) in Figure 1) - enabling requirements that were conspicuous to neither analyst nor user to be addressed. In the next section we describe how the RepGrid technique can be used to enhance requirements elicitation. We then explore the benefits using a case where the technique contributed to the understanding of user perceptions and requirements for a health care decision support system.

THE REPERTORY GRID TECHNIQUE

Use of RepGrid for requirements elicitation involves three phases, a creation phase, an assessment phase, and a clarification phase.

Phase I: Creation. The RepGrid process begins with a blank Repertory Grid (Figure 2) and semi-structured interviews with each individual. The goal for the first phase is to identify names for the task elements, which are used to label the columns of the RepGrid, and to identify names for the constructs, which are used to label the rows of the grid. Although the analyst can provide the task elements - if they were familiar with the work context, for instance - user-centricity is optimized by eliciting elements directly from the users, minimizing the risk of misperception in later phases of analysis and design.

Once this list of task elements has been compiled, the names given are copied onto index cards.1 The cards are then presented back to the individual, three at a time. Participants are asked to identify the "odd one out" from the 'trial' of three cards, by explaining how the paired task elements are alike and how the third is different.

These explanations are recorded: they identify the poles of the construct used to differentiate the triad of task elements offered. Figure 3 shows a simple example. "Check mail" and "verify customer details" were similar "gate keeping" tasks, while "confirm order" was seen as different, concerned with "quality control." This process continues until all card comparison possibilities are exhausted. At the conclusion of this first phase, the task element and construct names are transferred to the grid, with the task element names as column headings and construct names - such as 'gate keeping' and 'quality control' - as row headings, populating the blank grid.

Phase II: Assessment. In the second phase, the user rates each task element against each construct in turn: we used a 9 point scale (see Figure 4). Beginning with the topmost row (construct), each element (column) is rated using the numbers 1-9 to represent the applicability of the labels that identify the poles of the construct being considered. Typically, users proceed left to right across each row. The use of an odd numbered scale and the allowance of blank grid cell values enable users to omit
or provide neutral ratings in situations where the poles of the construct are irrelevant or apply equally to a particular element. Having completed the top row of the grid, the user moves down to the second row and applies numeric values to the new scale represented by the labels of the poles of construct 2. This process is repeated until, as far as possible, all the cells in the grid have been rated.

After rating, two-dimensional cluster analysis (9, 11, 15, 28) is used to re-order the rows and columns and highlight the statistical association of the ratings provided by the user. The cluster analysis identifies tasks that have been rated similarly by the user. The analysis groups the most similarly rated elements and constructs: the degree of association is shown through the addition of a simple hierarchical structure to the grid (see Figure 4). The scales on the right hand side of the figure are simply an indication, in percentage terms, of the relationship between the rows or columns spanned by the branches of the hierarchical “tree.”

**FIGURE 2**
The Structure of the Repertory Grid

<table>
<thead>
<tr>
<th>Pair</th>
<th>Task Element 1</th>
<th>Task Element 2</th>
<th>Task Element 3</th>
<th>Task Element 4</th>
<th>Task Element 5</th>
<th>Task Element n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construct 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct 1</td>
</tr>
<tr>
<td>Construct 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct 2</td>
</tr>
<tr>
<td>Construct 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct 3</td>
</tr>
<tr>
<td>Construct 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct 4</td>
</tr>
<tr>
<td>Construct 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct 5</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Construct n</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Construct n</td>
</tr>
</tbody>
</table>

**FIGURE 3**
Triad of Task Elements

**Phase III: Clarification.** Our use of RepGrid in requirements elicitation does not rely on any further statistical analysis since this would tend to overwhelm the processes of analysis and interpretation by the user. Instead, a ‘talkback’ procedure (38) is used to guide further conversation about the re-ordered grid. During this phase, the analyst talks the user back through the reordered grid using the tree structure to prioritize and guide the discussion. Beginning with the most
significant (highest percentage) associations – the ‘tits’ of the tree branches - the analyst asks the user to explain what the associations signify. The RepGrid provides a medium for further discussion and co-discovery of the user’s task context (15, 18, 38, 38). For example, in Figure 4, the bottom-most three elements (check currency of utilisation of database; compile hospital and nursing home utilisation of data; prepare quarterly utilisation report) have the highest percentage score (about 93%). When asked what such a close association of those tasks meant, the user explained that the tasks concerned involved contact with others outside the organization to increase understanding of the task at hand.

**FIGURE 4**

A Requirements Repertory Grid following Cluster Analysis

<table>
<thead>
<tr>
<th>follows a protocol</th>
<th>allows interpretation</th>
<th>data triage</th>
</tr>
</thead>
<tbody>
<tr>
<td>a more stable task</td>
<td>a less stable task</td>
<td>investigate anomalies between quarterly reports</td>
</tr>
<tr>
<td>a flexible task</td>
<td>an inflexible task</td>
<td>choose data presentation format/medium</td>
</tr>
<tr>
<td>does not allow</td>
<td>allows choice</td>
<td>data tuning (change research parameters)</td>
</tr>
<tr>
<td>choice</td>
<td>an investigative task</td>
<td>interview service providers</td>
</tr>
<tr>
<td>has a fixed format</td>
<td>subject to a wide range of influences</td>
<td>prepare inventory of 10 essential services</td>
</tr>
<tr>
<td>a mundane task</td>
<td>more subjective</td>
<td>represent Health Council at meetings</td>
</tr>
<tr>
<td>less subjective</td>
<td>involves complex communication</td>
<td>attend MAPP meetings</td>
</tr>
<tr>
<td>involves little communication</td>
<td>requires analysis</td>
<td>develop response (report) for enquirer</td>
</tr>
<tr>
<td>requires little thinking</td>
<td>more stimulating</td>
<td>identify anomalies between reports</td>
</tr>
<tr>
<td>less stimulating</td>
<td></td>
<td>conduct observational studies</td>
</tr>
</tbody>
</table>

This phase of the process enables a number of additional unifying concepts to be articulated and recorded, providing the basis for a user-driven model of the work context and deepening the analyst’s understanding of what the users require of the system - and why it is important. Repeating this process with other users enables the analyst to learn about the issues and concerns that are most important from their perspective. Such ‘mutual discovery’ (quadrant d in Figure 1) is supported by the commonality and sociality corollaries of PCT (9, 15) and the capacity of the RepGrid software to compare grids. As the RepGrid process is repeated within a community of users, the significance and prioritization of requirements is made evident to the analyst. Collating the RepGrid analyses and using the results to guide further enquiry enables the analyst and users to move toward collective agreement. This adaptation of RepGrid represents a considerable extension of previous applications in systems analysis and design (15).

To better illustrate the benefits gained from the use of RepGrid in requirements elicitation, we provide a brief account of our use of RepGrid during systems development. The following section describes the development site and context, the system of interest, and how RepGrid enabled its users to articulate requirements that had previously remained unrealized.

**REPERTORY GRID AS A TOOL FOR REQUIREMENTS ELICITATION**

The case site is a private, nonprofit, community health-planning agency established by the Florida Legislature in 1982. The agency is involved in a wide array of planning and evaluation projects as well as administrative and fiscal management activities for the four county districts it serves. The agency addresses healthcare allocation and resource distribution issues relating to the unserved and under-served in health care. Health planning agencies collect data from a variety of sources and employ various tools for data analysis and presentation purposes. Many of their tasks entail combining data and tailoring it to each particular request from their clients.

Our development team was implementing a decision support system comprising customized data cubes (17) that drew their data from the Comprehensive Assessment for Tracking Community Health (CATCH) database warehouse. The CATCH data warehouse integrates fine-grained event data such as vital statistics (birth and death records), hospital discharge data, freestanding clinic data, along with several more detailed disease registries. Our primary interest was to design a decision support system and related data cubes to enable the users to use OLAP functionality, and be able to adapt the data and analysis to their particular focal task.
In order to assess whether our determination of users' requirements were moving us toward this goal, we needed a method that would increase our understanding of their data and analysis requirements. Our priority was to clearly identify users' needs in the context of their daily tasks, based on their perceptions. The analysis and learning challenge that faced us – summarized in Figure 1 – was to avoid inappropriate attribution of needs. This occurs when the developers' perceptions of users' wants are used as surrogate, albeit rational, specifications that supplant the users' real needs (40). The complexity and dynamics of both the task environment and the data cube significantly increased the risk that requirements analysis would be affected by developer-defined needs (arising from our own prior experiences with data warehouses and data analysis tools).

The RepGrid technique was ideally suited to avoiding such perceptual bias in this situation. Requirements learned from the application of the RepGrid helped guide the design of the data cubes and user interface.

After initially creating the grid and assessing the task elements and constructs identified by the user, we began the talkback process to clarify the associations identified from the cluster analysis. During clarification of the topmost four elements shown in Figure 4 (data triage; investigate anomalies between quarterly reports; choose data presentation format/medium; data tuning – change research parameters), the user who developed this grid explained that "...these tasks involved more creativity than analysis." The grid shows that the topmost four constructs (follows a protocol – allows interpretation; a more stable task – a less stable task; a flexible task – an inflexible task; does not allow choice – allows choice) were the primary differentiators of this element cluster. The constructs compared task stability and flexibility and the extent to which choice and interpretation was required. Data triage and data tuning were specific task elements (see Figure 4) that exemplified the creative data manipulation necessary for the user to interpret clients' needs. The user observed that "...they (clients) don't really know what data they need. This happens 50% of the time." The ability to 'triage' and 'tune' the data was an essential requirement, but something that had not come to light during previous elicitation interviews.

The requirement for flexibility, providing the required facility to 'triage' and 'tune' the data was, in this context, at least as important as the quality of the data itself. Use of RepGrid enabled us to identify this requirement. By eliciting this 'want,' we were able to ensure that the system supported the user's desire to exploit the capacity of the OLAP tools. Had we not used this technique, there is a strong possibility that the users' range of retrieval, presentation and, therefore, decision making options would have been limited by the developers' perceptions of the users needs, based on prior interviews and discussion. The significance of the discovery of this particular requirement was made clear in a post-RepGrid interview when the developers offered 'to improve' the system by providing "...a web interface that is simple to use but which does not have much flexibility." The user's response was "No! We can do that! I think we need the tailored thing. I'd rather have the flexibility...don't even bother with the other (simple web interface)."

By fulfilling the users' requirement for flexibility, the RepGrid interviews enabled us to develop the system to more fully support the users' decision making. The example above shows that the enhanced capacity to compare demand for health services and their provision informs decision making at a number of levels. In addition to the ability to assess the match between supply and demand, the capacity to 'triage' and 'tune' the data enabled the users to enlighten themselves and their clients about the significant but poorly documented level of unfulfilled demand and, in turn, the health policy implications of the data stored in the CATCH data warehouse.

RepGrid can contribute to the requirements elicitation process both directly – as illustrated above – and indirectly. The significance of the unifying concepts underlying the hierarchy of elements and constructs (see Figure 4) is not always immediately realized by either analyst or user. During the talkback phase, a client explained that a particular element cluster represented "...larger, amorphous, less clearly defined (requirements)...with variable impacts (rather than) a specific project." Prompted by the need to more clearly define this requirement, discussions with the user elicited specific concerns over new tasks to examine physician shortages or flight, exposing a requirement area that had not been explicitly recognized by the designers or users. The RepGrid and later discussions served to refine the users' needs and helped to uncover the fact that including data on actual physician services rather than just license data would be far more helpful in identifying physician count trends.

This is an example of a design innovation that had not been anticipated by the users or developers. The RepGrid prompted a discussion that revealed a 'requirement' that was previously 'unknown' to either party, as shown in Figure 1, quadrant (d). The developers responded by making sure that fine-grained data was used to build the OLAP components, providing the user with the ability to explore hundreds of different diagnostic and surgical categories along with a unique physician count. This allowed the users to create new analytic views that focused on the number of physicians providing various services, as well as trends in average caseloads. The discussions guided by the RepGrid helped us understand not only what the users wanted to do, but also why they wanted to do it, and what features of the system would be necessary to provide them the ability to do it.

BENEFITS GAINED FROM THE USE OF THE RG TECHNIQUE

Prior research (3, 5, 7, 13, 17, 19, 31) has repeatedly identified the need to increase the size of the shared 'pane' (a) in Figure 1 by shifting its boundary to the right. Work that has evaluated and compared various information systems development methodologies, tools and techniques has highlighted the challenge that this presents (1, 33). Our experience with RepGrid enabled us to make progress in this direction, allowing us to elicit requirements that might otherwise have been missed or misunderstood because their definition was unclear. By providing the users' with a medium to reflect on their work, we were able to increase the range of requirements identified and articulated - those previously in the Needs (c) or Unknown (d) areas (Figure 1).

The classification of communication issues in requirements elicitation set out in Figure 1, although rather rudimentary, provides a basis to compare user-analyst communication techniques, an important first step in answering our research question. Our research into and use of techniques such as interviews, observation, scenario analysis and contextual inquiry in a number of contexts and settings prompted us to explore the capacity of RepGrid to contribute to the communication challenges that persist in requirements elicitation. Rather than attempting to 'capture' or determine requirements according to a given set of goals or other predetermined frame of reference, we sought to more fully understand and appreciate the values that informed the users'
perceptions and judgments of their work and the technologies they used to support it. Such an approach is particularly important where both technology and its use are changing quickly, as in the case reported. Our research to date highlights the potential dangers of over-generalizing requirements by applying generic explanations and criteria drawn solely from the analysts’ previous experience. Even though the jargon or group shorthand used everyday by users or analysts might be familiar to everybody, it is unsafe to assume that it carries exactly the same meaning for everybody (36). The successful design of newer, dynamic, malleable information technologies depends in large part on an understanding of users’ requirements that is itself adaptive and accommodating. However, there are some significant cautions that must be kept in mind.

Firstly, the RepGrid requires practice in administration. Several software packages (9, 11, 27, 28) advocate and accommodate direct input by the user. Our previous experiences in the healthcare, transport, logistics and financial services sectors showed this to be overwhelming for the user and ineffective for requirements elicitation. The RepGrid process was faster and more productive when the analysts undertook input. This approach limits the (not insubstantial) learning burden to the analyst, enabling the analyst to coach the user through the three phases described above. Secondly, the technique is time consuming, requiring about 1½ hours for each interview. However, these issues are compensated by the richness that the technique adds to user-analyst communication.

COGNITIVE MAPPING TECHNIQUES

The RepGrid is one of a number of cognitive mapping techniques that are, collectively, strong candidates for eliciting requirements in the ‘unknown’ (d) and ‘needs’ (c) quadrants of Figure 1 – situations where the communication challenge requires a degree of co-discovery and conceptualization (29, 34). In addition to RepGrid, there are three other well-known cognitive mapping techniques – causal mapping, semantic mapping and concept mapping.

A cognitive map is a representation of the ‘mental model’ or internal schema used by an individual in problem solving. Our successful exploitation of RepGrid and other previous research (13, 16, 29, 30, 32) has demonstrated the capacity of such maps to help guide the elicitation process.

Mapping techniques vary in their theoretical bases, ease of use, ease of interpretation and the format of their output. Stau and Tan (34) review the comparative strengths and weaknesses of causal mapping, semantic mapping and concept mapping. They also highlight the significance of these techniques in overcoming the technical communication challenges that arise in information systems analysis and design (32). In the remainder of this section, we briefly consider the strengths and weaknesses of RepGrid when compared to these other cognitive mapping techniques.

The most obvious differentiator between RepGrid and the other three mapping techniques is the form of the output. The two-dimensional structure of the grid and the dendrograms that arise from cluster analysis (see Figure 4 above) is quite different from the more familiar cluster and tree-like ‘maps’ that arise from use of causal, concept and semantic mapping. Although we have highlighted the strength that arises from the capacity of the RepGrid’s two-dimensional structure to elicit users’ experience – such as tasks that they undertake in the course of their work and the constructs that they use to differentiate them – the idiosyncrasy of the RepGrid ‘map’ makes it less immediately intuitive to novice users. In addition to the time required to undertake the interviews and complete the data analysis, considerable time is required to learn to use the technique effectively.

All four techniques share the capacity to focus attention, trigger memory, reveal and fill ‘gaps’ and direct attention to areas that need further exploration (34). For these reasons, all four are powerful supplements to traditional techniques such as structured interviews. The following table revisits the communications challenges summarized in our model (Figure 1) to provide a brief comparison of the strengths and weaknesses of the four techniques.

Clearly, the RepGrid involves a higher degree of cognitive effort than the other three mapping techniques: that much is evident from comparison of the RepGrid output with the more familiar hierarchical maps generated by causal, semantic and concept mapping (32). However, our experience of use of the technique – briefly illustrated and discussed in our case tutorial – shows how it can be used to reduce the occurrence of bias and misperception in requirements elicitation.

IMPLICATIONS AND CONCLUSION

Our research suggests that the challenges of requirements elicitation arise, at least in part, from the mis-matching mental models of user and analyst. Our typology of communication challenges in Figure 1 suggests that reliance on systems analysis tools and techniques that place greater emphasis on standardization and re-use might reduce rather than increase the repertoire of requirements discussed between the analyst and their client. The brief case tutorial showed how the RepGrid technique was used to rebalance users’ contribution to the requirements elicitation dialogue, complementing other analyst-driven tools and techniques. The content-free structure of the grid substantially reduced the likelihood of inappropriate attribution of needs – a common shortcoming of structured interviews and questionnaires – and allowed us to take a complementary, inductive approach to requirements analysis. Unlike structured techniques such as interviews and questionnaires, or unstructured techniques such as contextual enquiry, RepGrid provides a medium for communication between user and analyst that, although structured in form, is content-free and therefore sensitive to context-specific requirements that might be missed using more traditional elicitation techniques.

The brief example above shows that it is possible to expand the shared area of conscious system requirements to the right (Figure 1), enabling unrealized and more tacit requirements to be identified. Our example shows how the RepGrid technique can be used to reduce users’ ‘blindness’ (21) by enabling them to articulate their requirements in context. The examples reported in our brief case tutorial show how we were able to expand the shared area (a in Figure 1) to also include requirements that were previously unknown by both the user and analyst. In this way, the unbiased interaction facilitated by the RepGrid technique allows the user and analyst to mutually discover system characteristics that enable them to move toward a shared understanding of tasks and requirements that might otherwise remain ill- or un-defined.

In summary, the RepGrid technique can complement other tools and techniques used for requirements elicitation by:
1) Allowing analysts to elicit requirements that usually are not articulated by users
2) Allowing analysts to elicit requirements in scenarios where they do not have “business knowledge”
3) Providing analysts and designers with a clearer picture
of the context in which tasks are performed by users
4) Providing analysts and designers with an understanding of the relevant elements of the context that impacts tasks
5) Helping prioritize user requirements by identifying the values that differentiate the task elements
6) Providing a tool to understand the evolutionary nature of user needs
7) Enabling training needs to be identified from within the task by user and analyst, rather than retrospectively following design and implementation.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Causal Mapping</strong></td>
<td>- Strong theoretical foundation (Personal Construct Theory)</td>
</tr>
<tr>
<td></td>
<td>- Most commonly used mapping technique</td>
</tr>
<tr>
<td></td>
<td>- Relatively low 'cognitive effort' required</td>
</tr>
<tr>
<td></td>
<td>- Reveals cause-effect relationships</td>
</tr>
<tr>
<td><strong>Semantic Mapping</strong></td>
<td>- Does not capture non-causal relationships</td>
</tr>
<tr>
<td></td>
<td>- More likely to elicit 'wants' (Quadrant b) than 'needs' (Quadrant c)</td>
</tr>
<tr>
<td></td>
<td>- Depends on the identification of a central single, unifying concept</td>
</tr>
<tr>
<td></td>
<td>- Potential for bias or misdirection if initial concept selection is inappropriate</td>
</tr>
<tr>
<td><strong>Concept Mapping</strong></td>
<td>- More likely to elicit 'wants' (Quadrant b) than 'needs' (Quadrant c)</td>
</tr>
<tr>
<td></td>
<td>- Strong theoretical foundation (Learning Psychology)</td>
</tr>
<tr>
<td></td>
<td>- Familiar, hierarchical structure</td>
</tr>
<tr>
<td></td>
<td>- Can accommodate causal, temporal or other classes of relationship</td>
</tr>
<tr>
<td></td>
<td>- Powerful ideal generation technique</td>
</tr>
<tr>
<td></td>
<td>- Quick and easy to develop</td>
</tr>
<tr>
<td><strong>RepGrid</strong></td>
<td>- Relatively high 'cognitive effort' required</td>
</tr>
<tr>
<td></td>
<td>- Slow and complex</td>
</tr>
<tr>
<td></td>
<td>- Unfamiliar structure</td>
</tr>
</tbody>
</table>

RepGrid cannot, and should not, replace traditional requirements elicitation techniques. Rather, like the other three cognitive mapping techniques discussed, it should be seen as a powerful complementary addition to the analyst's development tool kit. The particular strength of the RepGrid technique is the provision of an opportunity for highly user-centered, demonstrably unbiased requirements elicitation. These qualities could add value in many systems development projects. RepGrid is, as we have shown, of particular value in settings where the specialty or unique nature of the work makes the use of traditional systems analysis techniques more prone to analyst bias.

ENDNOTES

1While any mechanism can be used to capture and present the task elements back to the user, we use index cards as they are easier to manipulate and provide a more visual experience for the user when comparing and contrasting the cards.

REFERENCES

10. Fagan, M. “The Influence of Creative Style and Climate on Software Development Team Creativity: An Exploratory Study.”