Exploring Language in Software Process Elicitation:
A Grounded Theory Approach

Carlton A. Crabtree, Carolyn B. Seaman, and Anthony F. Norcio

Information Systems Department
University of Maryland Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250 USA.
{cac1, cseaman, norcio}@umbc.edu

Abstract

This paper presents the results of exploratory research that investigated how people describe software processes in natural language. We conducted a small field study with four participants working at an IT Help Desk. We elicited and documented a trouble ticketing process using a template under conditions similar to that of many process improvement initiatives. This study included two treatments. In the first treatment, the process engineer elicited information and documented the process. In the second treatment, the participants used the template to document the process on their own. The resulting data, including the process representations, observation field notes, and interview transcripts, were analyzed using a grounded theory approach. The results suggest that there are distinct ways in which process users describe process. We construct a theory that posits that descriptions of process are dependent upon perspectives shaped by the elicitation and process context. Future research will focus on the evaluation of this theory relative to other elicitation approaches and contexts.

1. Introduction

Elicitation is an important first step in creating documentation to describe software processes. Elicitation requires communication, extraction of knowledge and a shared understanding of goals [12]. Software process improvement initiatives often begin with process elicitation efforts that are aimed at producing documentation to demonstrate compliance with audit polices and best practice. In order to identify potential changes to existing workflow, organizations typically produce representations of various software development life cycle processes, in various forms, to establish the process baseline [8]. Once a process baseline is created, it provides relevant information to management about existing work practices so that the processes can be analyzed for inefficiencies (i.e. redundancy, gaps, integration problems) and refined to remove those inefficiencies. In process improvement, the process baseline undergoes changes to enforce new prescriptive rules of how the software processes should function [9]. For example, an organization might be required to make changes to the configuration management process and the software release protocol to accommodate mandates in the CMMI framework [6]. Unfortunately, hiring process specialists as consultants to build process representations to portray these activities can be very costly. As a result, many companies choose to model their own software processes. In most cases, the software processes are not already documented, so everyday software practitioners are tasked by executive management to take the lead in documenting the details of the individual tasks in each portion of the software development life cycle.

Eliciting software processes and creating representations can be difficult in these circumstances. Depending on the communication abilities of those building the representation, elicitation can be time consuming and inefficient. One person’s viewpoint of the software process may be completely different from another person’s perspective. Prior research also shows that humans demonstrate a sophisticated capability of generalization, creativity and indirect problem solving [16] [19]. However, a user’s articulation of the process model may differ dramatically from the actual performance of the process [7]. These issues add to the challenge of understanding and representing software processes. The outcome of process improvement is usually
expected to be better software quality, enhanced coordination, and extensive process documentation. However, these benefits are dependent upon the accuracy of the elicited information, the domain knowledge of the people involved in the effort and their ability to describe existing software processes [17].

1.1. Research Questions

The aim of this study is to understand how humans perceive real world software processes through analyzing the language they use to describe them, and to investigate the factors that affect process interpretation by those involved in creating a representation of the process model (both process users and process engineers). We feel that understanding these important aspects within the context of software process improvement may lay the foundation for future research. This study will focus on the creation of a grounded theory that addresses the following research questions:

- What thoughts are occurring in the process user’s mind while attempting to document a process?
- How do process users describe the process model elements in natural language?
- How do process users articulate existing processes?
- Finally, how do process users respond when having to represent changes to existing processes?

1.2. Scope

The scope of this study is limited to investigating elicitation of software processes. We do not review elicitation in cognitive psychology, knowledge based approaches or requirements elicitation, although these areas may eventually provide important theoretical foundations to our work. However, given the exploratory nature of this study and the specific research questions being investigated, we defer discussion of these related areas until this line of research is more developed, and it is more clear how directly applicable these research areas are. Since we are not starting with a set of hypotheses to guide the study, our goal is to openly investigate the factors affecting process elicitation, such as that performed as part of software process improvement, in a real world setting.

1.3. Definitions

In this study, we generally refer to software processes as coarse-grained software development activities in an organization such as requirements analysis, design, testing, configuration management and so forth. Software process models have been characterized as: “a selective abstraction of real world manifestations” [15]. We use this definition as a foundation for analysis in this study. While prior studies have emphasized the importance of the process performer, this study uses the term process user to describe any professional in the software development organization following task oriented descriptions. The term process user is slightly more personalized for the context of this research and underscores the importance of the software professional as a customer for the processes that they use. The term process engineer includes anyone facilitating the effort of documenting descriptive processes. Additionally, we define elicitation as the set of activities performed by the process engineer and process user to collect and analyze process information in order to document the software process. The process engineer may also be a process user and is usually a subject matter expert internal to the organization who is responsible for documenting processes. Finally, this study specifically considers instances where the process user and engineer are directly discussing or documenting software development activities as software process elicitation.

2. Related Work

In the next few sections, we review three well-known models in software process elicitation that have direct influences on our research.

2.1. Elicit

A highly recognized and notable work in software process elicitation is that conducted by Madhavji et al. with the Elicit Method [18]. Elicit provides a robust framework for identifying a “meta model” for eliciting descriptive processes. Madhavji et al. describe their approach through classification of phenomena into three unique dimensions: View, Method, and Tool. The Elicit approach covers various organizational characteristics consistent with “identifying process modeling goals, planning for elicitation, eliciting process information from numerous sources, synthesis of this information into a formal process model, validation, and analysis of the elicited model.” [16, p. 113]. Madhavji et al. describe the process modeling
strategy for elicitation. First, the View dimension summarizes the basic requirements for a theoretical process model which include: process steps, artifacts, roles, resources and constraints. The Method dimension includes a seven step sub process including the specific details of the Elicit method: “(1) understanding the organizational environment, (2) defining objectives for eliciting a process model, (3) planning the elicitation strategy, (4) developing the process model, (5) validating the process model, (6) analyzing the process models, (7) post-analysis of the usage of the method and (8) packaging of the experience gained” [18, p. 113]. Finally, Madhavji et al. describe the Tool dimension. Statemate [13] and Elicit are the technology platforms applicable to the Tool dimension. As seen in Fig. 1 below, the Elicit command line application interface is used to extract the elements of the process model [14, p. 6].

![Figure 1. Elicit command line application interface](image)

Statemate [13] then extends these textual elements into graphical representations for capturing static and dynamic properties. The Elicit Approach has been successfully tested in a number of different industrial projects and provides a clear strategy for the process engineer. Another critical component of Elicit is the process elements depicted in Fig. 1. Madhavji et al. describes each of the process model elements such as Goal, Purpose, Procedure, Messages etc. Each process element is outlined on the left with corresponding input panes on the right for further detailed description. We see this format as potentially useful as a selective abstraction of the process model.

Unfortunately, Elicit is technical and platform dependent. This impacts its ability for practical application by everyday software professionals involved in process improvement. This is an important consideration for future development in software process elicitation strategies in that not all process users have the same level of technical knowledge that can be leveraged in elicitation. Madhavji et al. reveals that deriving formal documented artifacts through natural language analysis in process improvement is a difficult exercise. Moreover, Madhavji et al. suggests that inspection of process improvement documentation reveals many inconsistencies where process users do not include all the necessary process elements. Analysis of natural language for noun usage to identify candidate objects is also problematic in that it is “mechanical” and “complex” [14, p. 9]. We wanted to understand this complexity and identified this as a unique opportunity for investigation in our research. The observations from Madhavji et al. are critically important and point to the need for specialized research approaches that consider natural language analysis. In our methodology and approach we propose that grounded theory may be uniquely fitted to researching language.

2.2. Spearmint

Another important area of software process elicitation research is that of Becker-Kornstaedt et al. and the Software Process Elicitation, Analysis, Review, and Measurement in an INTegrated Modeling Environment (Spearmint) [4]. The Spearmint approach focuses on technical support for the process engineer. As with the integrated process engineering environments supported by process modeling languages (pmls) in the late 1980’s, Spearmint provides capabilities for classifying and representing process models in an application environment that is web based. The power of the Spearmint approach rests in the Electronic Process Guide generator (EPG) tool to create process specifications and to provide direct support for the process engineer. The EPG disseminates domain knowledge throughout the organization. In this way, Spearmint directly impacts efforts for documentation of processes in process improvement.
Additionally, the Spearmint approach provides a powerful tool for the process engineer to increase domain understanding about how software processes are performed and integrated. However, knowledge can still be lost in interviews and meetings that attempt to reinterpret a process user’s understanding. Elicitation and subsequent documentation by the process engineer only interprets a process user’s knowledge. As outlined in the introduction and scope, process users are often directly responsible for documenting processes on their own in process improvement. Our research approach is guided by trying to understand how support can be provided to process users in these circumstances.

2.3. Prospect

Becker-Kornstaedt et al. formulate a decision model to address the problems faced by inexperienced process engineers conducting elicitation efforts [5]. The Prospect model (Process-outline based software process elicitation) is based upon the Goal Question Metric (GQM) paradigm proposed by Basili in 1984 [3]. The GQM paradigm is initially used to establish the goals for the process engineer. Through the phases of orientation and detailed elicitation, the process engineer works in iterations to complete the process model. “Prospect techniques are defined according to a template. This template includes the preconditions for the application of a technique, the information source types (e.g., roles or document types) the technique can be applied to, and the information a technique can elicit in terms of process model elements [5, p. 7].” We were particularly interested in the idea of using a template to elicit process model data and saw this as a very interesting opportunity to leverage these insights in our study design.

It is also relevant to note that the Prospect technique uses natural language and questions to elicit the process model data. Through a case study the Prospect method was evaluated in real world conditions. Becker-Kornstaedt et al.’s work with Prospect was a major influence on our decision to use templates. However, because our interest in understanding the conditions applicable to process user documentation of process, we extend her interpretation to allow process users to directly use a template when articulating their thoughts about how the process should work.

3. Methodology and Approach

Viewing a software process model as a “selective abstraction” provides flexibility in deciding how to represent the process model. We wanted to identify a tool that could provide a practical real world representation to elicit the software process but that would not be overly technical for the process user. In the end, we decided on a use case template because it strongly resembles other available formalisms for process models. Also we feel our construction of the use case template is quite similar in format to the approaches taken by Madhavji et al. and Becker-Kornstaedt et al. Finally, our comparison of a use case template to formal definitions for software process models shows a strong similarity in that the use case template contains many of those elements that are considered to formulate a process model, such as artifacts (inputs/outputs), activities (tasks) and resources (entities) [13]. Table 1 shows the process elements we used in the template, and the corresponding natural language instructions for the process engineer or user.

<table>
<thead>
<tr>
<th>Process Element</th>
<th>Natural Language Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>What is your primary role?</td>
</tr>
<tr>
<td>Process Name</td>
<td>What is the name of this process?</td>
</tr>
<tr>
<td>Description</td>
<td>Please describe this process.</td>
</tr>
<tr>
<td>Trigger</td>
<td>What action begins this process?</td>
</tr>
<tr>
<td>Entry Condition</td>
<td>Before this process begins what information do you already have?</td>
</tr>
<tr>
<td>Inputs</td>
<td>What information do you use in this process?</td>
</tr>
<tr>
<td>Input Entities</td>
<td>Who or what provides you with the information to perform this process?</td>
</tr>
<tr>
<td>Process Steps</td>
<td>Please list each step to complete</td>
</tr>
</tbody>
</table>
this process.

<table>
<thead>
<tr>
<th>Outputs</th>
<th>Once this process is completed what is produced?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Entities</td>
<td>Who or what uses the output that is created from this process?</td>
</tr>
<tr>
<td>Exit Condition</td>
<td>When this process is complete, what information have you produced?</td>
</tr>
<tr>
<td>Exceptions</td>
<td>Are there any exceptions to this process?</td>
</tr>
<tr>
<td>Errors</td>
<td>Is there the potential for error in this process? If so, could you please describe these conditions?</td>
</tr>
</tbody>
</table>

3.1. Data Collection

Data collection was conducted with four software professionals at a small IT Help Desk. All sessions were audio recorded. Prior to data collection, a meeting was held with executive management and with the participants to discuss what type of process would be a candidate for the study. It was decided that the process of opening a problem ticket to track a customer issue for resolution would be appropriate because it was explicit enough and not overly abstract. Two experimental treatments were designed to realistically simulate the challenges that can occur in real world elicitation. Two participants were exposed to Treatment A and two participants were exposed to Treatment B. In the first treatment (Treatment A), the process engineer (played by the first author) elicited information about how the process user performed the process, using an interview format, and using the use case template to document the process. We saw this treatment as representing the typical scenario where a consultant or process engineer collects the process model information from the process user. In the second treatment (Treatment B), the process user used the template (in Microsoft Word) to document the process on their own while the process engineer asked questions about what was going through their mind. Treatment B was intended to represent conditions where the process user was directly responsible for defining the process. In both treatments, questions were asked in the same format for consistency. The interview questions were based upon the use case template natural language instructions as depicted in Table 1. We felt it was important to create questions that removed the process user from the technical interpretation of the process model element but that also still captured the technical meaning for that element. For example, instead of asking the participant to tell us what the “triggers” for the process were, we instead asked “what set of actions begin the process?” or “how does this process start?”

It is important to note that while the questions and the interview format was consistent and structured, this study also employed an interactive observation session style where the researcher (taking on the role as process engineer) and research participant, (taking on the role of process user) were free to exchange viewpoints and communicate openly about the challenges and perceived difficulties in documenting the existing process for opening a problem ticket.

The participants were not allowed to reference any change control tools or other process documentation during either treatment. Participants were instead directed to use their existing memory to articulate what they thought the process entailed. This was done to simulate the lack of documentation and realistic cognitive demands placed upon people in elicitation efforts in software process improvement. Additionally, our participants were encouraged to “think out loud” as they were writing in Treatment B. In both sessions, probing questions were used, such as “What is going through your mind right now?” or “Do you understand the context of the question or are you having difficulty describing the process?” or “Why did you choose to describe the process that way?”

Finally, in an attempt to simulate a basic process improvement change, we designed a process variation that required that the existing process for opening problem tickets be modified to include manager approvals. The change was introduced, in both treatments, after the initial version of the process had been elicited. Participants were verbally told about the process change and asked how they thought the model would be impacted as a result without reference to the template. Afterwards, they were allowed to look at the template (in Treatment A) to see how their understanding changed or use the template to document where they thought it would change (in Treatment B). We were specifically interested in how people reacted to this stimulus because often software professionals are confronted with similar conditions in process improvement and are expected to understand how and where the overall software development activities will be impacted.

3.2. Data Analysis

The overall goal in the data analysis was to approach the problem of complexity and ambiguity in the data with an open mind and to look for trends and patterns in the qualitative data. We employed a grounded theory approach to analyze the textual data. Grounded theory analysis has been successfully applied in studies of process improvement such as that illustrated by Coleman and O’Connor’s study in the Irish software industry [8]. We feel this type of analysis is relevant to our research as well.

In grounded theory the researcher approaches the research phenomena of interest with the intent of producing a theory or narrative to describe the data.
The term “theory” is used here to mean an overarching description of what has been identified in the data. It is not meant to imply that any propositions or hypotheses were tested and validated, nor does it mean that the theory will hold in all contexts. The grounded theory simply represents the findings of the study in a form that allows for further empirical evaluation. We adopt Strauss and Corbin’s interpretation of grounded theory, which provides greater flexibility in data interpretation and also allows pre-formulation of research questions [21]. Going into the analysis phase, there were no preconceived notions of how the textual data would be sub divided and organized, although we did understand at a general level that we would be using our research questions to guide that analysis.

First, consistent with qualitative methods in empirical studies, we started by documenting the researcher’s observations and impressions immediately after each session, and writing field memos based upon these observations [20]. We then compared field memos from different sessions identifying similarities or points of interest between participants. Second, all interviews were transcribed verbatim into separate electronic files with line numbering enabled to track individual instances. Afterwards, all transcripts were read and repeatedly compared to the field memos. The researcher constantly checked to see if the recorded impressions from the field memos were consistent with the interview transcripts. Finally, a set of post-formed codes was created based upon the initial research questions, and text in the interview transcripts was categorized. These codes are further described in the next sections.

3.2.1. Open Codes. The first step in our data analysis is open coding [10]. In open coding, data are analyzed for high-level concepts and categories that describe the text. In this case we were able to code most if not all of the data with a set of open codes derived from our research questions. The following codes were created based upon the research questions:

1. **Process Thoughts (PT)** - What thoughts are occurring in the users mind when reasoning about process?
2. **Process Description (PD)** - How does the process user describe the process elements in natural language?
3. **Process Articulation (PA)** - How easy or difficult is it for the process user to articulate the existing process?
4. **Process Change (PC)** - Finally, how do users respond when having to make changes to the existing descriptive process?

3.2.2. Sub Codes and Properties. Additional sets of sub-codes and properties were needed to capture additional detail that was not first anticipated when analyzing the interview transcripts. It is important to note that the creation of sub codes and properties is a creative process and not based upon the opinion of the researcher to force interpretation but upon the analysis of the data [10]. In other words, the researcher did not start with the sub codes and properties attempting to fit the data to them but instead derived them from an analysis using the initial open codes (representing the research questions) and from thorough reconciliation of textual relationships and understanding of interview data. In analysis we found that in order to identify sub codes and properties we had to ask additional questions based upon the original open codes to put the data in context. The outline below presents our derivation of the sub codes and properties:

1. Based upon participant responses to questions, at what level of detail is the process being described? How can we capture level of detail based upon the data?
   a. **Open Code**: Process Description (PD)
   b. **Sub Code**: Level of Detail (LoD)
   c. **Property**: Example (E): The participant is reasoning about the process through example and describing the individual instances or instantiations of the descriptive processes.
   d. **Property**: General (G): The participant is reasoning about the process generally and is not citing any specific examples. The descriptions provided by the participant represent more of an overview.

2. Thoughts about the process and memory are being used throughout the interview. In what context are they generally being applied?
   a. **Open Code**: Process Thoughts (PT)
   b. **Sub Code**: Memory (M)
   c. **Property**: Process Context (PC): The participant is using memory to describe and recall the descriptive processes.
   d. **Property**: Elicitation Context (EC): The participant is using their memory to recall the interactive observation session.

---

1 Grounded theory methodology is not reviewed here. The reader is directed to [12].
2 Further explanations and theoretical debates on the issue of emergence versus forcing can be found in Glaser’s 1992 text [11]
3. Based upon analysis of the responses in both treatments, how can we interpret the ability of the process user describe the process?
   a. **Open Code**: Process Articulation (PA)
   b. **Sub Code**: Ability to Describe (AD)
   c. **Property**: Difficult (D): The participant is confused and/or is having difficulty in discussing the process
   d. **Property**: Easy (E): The participant seems comfortable and is able to describe the process with relative ease

4. How did the participant react to the process change?
   a. **Open Code**: Process Change (PC)
   b. There were no identifiable sub codes
   c. There were no identifiable properties

3.2.3. Axial Coding. The next step in our data analysis was to code the data axially. In axial coding the researcher looks at the text and attempts to identify similarities and relationships among the data. We analyzed the open codes initially created and identified areas of overlap between all open codes, sub codes and properties. An example of this type of analysis is represented below:

   **Process engineer**: “Please tell me each step you take to complete the process?”

   **Process user**: “I prioritize the tickets. Then I start on each ticket. I call the person and setup a time. If it is something very easy, I just kinda walk out there...I also look to see if there is an error message or something that would not make it an everyday problem...Those are like the general first steps...There really isn’t a lot of thought process that goes into it” [PD,LoD(G)] [PT,M(PC)].

   In the example above the participant is describing the process at a level of detail (LoD) that is general (G). The process thoughts (PT) and memory (M) seem to be related to the process context (PC). This type of analysis was performed for Treatment A where the researcher (acting as process engineer) documented the process and for Treatment B where the participant (acting as process user) attempted to document the process on their own. The researcher experienced difficulties in this part of the process in that it seemed that in order to establish connections between the codes a core category had to be established first. At times the data had to be completely abandoned for several days so that a fresh perspective could be obtained.

3.2.4. Selective Coding. Eventually, we found it easier to establish these relationships by attempting to identify the overarching core category based upon the open codes. This led to the part of our data analysis that generally resembled that of selective coding in grounded theory, where the researcher is attempting to identify the core theme that will guide a narrative theory to explain the data [9]. Once this category was identified, all codes, sub codes and properties would fall underneath it. We visualized trying to pick up the core category as if to hold it steady in the air such that all other elements would dangle underneath and fit neatly together. The axial relationships were visualized as that of connections being formed between open codes, sub codes and properties (Fig. 3). If we could articulate all elements underneath the core category then we had identified the narrative basis for our theory. Since all of our codes were based upon the data and research questions we felt comfortable in simply asking if all of the codes, properties and their meanings could be explained underneath the core category in the form of a narrative.

![Figure 3. Grounded Theory Representation](image)

It is very difficult to portray grounded theory analysis to the reader through text, therefore, in order to visualize the results from this study in a meaningful way we created a picture and narrative to explain the data (Fig. 3). We have uncovered a picture in the formulation of our grounded theory that demonstrates that there are significant complex relationships present in the data. By identifying process description as a core category, a representation of the theory is revealed. The visualization was derived from the selective coding process. The original open codes (representing research questions) are connected by double headed arrows, the open code to sub code and property relationships by single headed arrows, and the axial relationships by dotted lines. The axial relationships are further described in the grounded theory narrative and in the discussion section.
4. Discussion

In this section, we first provide the grounded theory narrative to accompany the representation in Figure 3. This created narrative is within the bounds of the data collected in this study. We then proceed by discussing the detailed observations and supporting data that support the formation of the model shown in Figure 3.

4.1. Narrative

How a process user describes a process in natural language (PD) is affected by that user’s ability to describe the process in words and their ability to think (PT) about and recall process details. In their process articulation (PA), users speak about the process at different levels of detail. They are sometimes able to express the process generally but at other times drop down into specific examples to aid their memory of details. Use of a template (as a selective abstraction) during the elicitation context by the process user appears to help them articulate the process context and recall details from memory. Finally, the perception of process change (PC) may be impacted by the presence or absence of a selective abstraction of the process model.

4.2. Supporting Data

In the following subsections, we present the data and analysis leading to various portions of the model presented in Figure 3. Excerpts from the data also include the grounded theory coding labels.

4.2.1. Process Description, Process Articulation and Level of Detail. At the highest level in Figure 3, we observe that process description, articulation and level of detail are related. The researcher (as process engineer) in Treatment A was intent on gathering a description of the process at a general level of detail, one that described how the overall process was performed. However, the researcher found that extracting this information was quite difficult because often the process user waxed and waned between various states of description. This was a major cause of back and forth and confusion between the researcher and participant in Treatment A. The following excerpt demonstrates the challenge:

Process Engineer: “What information do you use to perform the process steps?”

Process User: “P1: His name, correct spelling of his name. Full time/staff/part – time, sometimes the length of time he needs the account.”

Process Engineer: “I feel like we might be missing some pieces and I’m just curious…When a ticket is opened, it’s only because a person doesn’t have an account.”

Process User: “No. I just used one example. I wanted to simplify this.”

Process Engineer: “I think the issue here is… I feel like I’m getting an isolated viewpoint. The examples may not give us the full picture”

Process User: “I zeroed in on one person. Maybe I should not use the specific example of the account. Maybe it would be a person having an issue with a printer.”

A potential source of this confusion is the fact that in Treatment A there was no shared selective abstraction of the process model in the form of a template for reference between the process engineer and process user. A selective abstraction of the process model creates a shared understanding for comprehension between process user and process engineer. Without a shared representation, it was difficult in our study to find common ground for discussion of the process model during the elicitation context. Considering that the intent for the elicitation of software process models is often to create manuals and guides that specify how the process is performed in a very generic sort of way, these findings may reveal some interesting insights into the current challenges faced by process engineers in the field.

4.2.2. Process Thoughts, Ability to Describe, General and Examples. Many of the questions asked during interview required the participant to use their memory to recall information. Based upon this, we also believe that there are lower level sub relationships between ability to describe and the subcodes and properties of the process thoughts code (memory of the elicitation context and process context). The observations collected suggest that the process users preferred to describe process through example. Consider the following excerpt from a probing question in Treatment A. The participant is reflecting on the relative ease to describe process through the use of example during the elicitation context.

Process Engineer: “How difficult or easy is it for you to describe the process generally without referencing a specific example of how the process works? What if I just asked you to tell me each step?”

Process User: “For me I like to think of a specific example because it helps me walk through the steps.”

Process User: “I can picture most of this information in my head as you are asking me it. I like specific examples it helps me think through it”

In Treatment B we observed that participants were able to use examples to describe the process. However, based upon the data collected, we feel the participants were able to describe the processes at a much higher level of detail using the template. Consider the following two written excerpts from Treatment B from

978-1-4244-4841-8/09/$25.00 ©2009 IEEE
two different process users who used the natural language instruction to document the process steps on their own using the template without intervention:

**Natural Language Instruction:** “Please list each step that you take to complete this process”

**Process User:** “Receive a call from a customer. Determine that there is a problem. Verify contact information. Determine which department would handle said problem and assign to that department. Input the problem in words by the customer into the Long and short description of the ticket. Verify that what was input was correct. Then submit the ticket which generates the ticket number” [PD,LoD(G)] [PA, AD(E)] [PT,M(PC)].

**Natural Language Instruction:** “What actions take place to begin this process?”

**Process User:** “A call is received by the helpdesk associate from the customer of the customer is received by physically walking in to the helpdesk and at that point we determine if in fact there is a problem and if it is something that we could actually handle at the helpdesk. If it is determined that there is a problem then at that point a ticket is produced.” [PD,LoD(E)] [PA,AD(E)] [PT,M(PC)].

It is observed in the two excerpts above that the process users are working at a very low level of detail while providing written descriptions (both general and detailed examples) of the process model. The participant seems fully capable of describing process using their memory recall of the process context. We feel we were able to obtain richer descriptions regarding the process context when the process user was allowed to directly use the template to document the details of the process model citing examples and using the template as a reference. Feedback provided from the following two process users suggests the same finding:

**Process User:** “I think when you actually have to sit down and type it out it causes you to think about the process even more. And you think about the process…in talking you tend to ramble and things don’t come as quickly. And when you write it out, you think about the process step by step and you think about things you forgot about when talking. It brings things down incredibly.” [PA,AD(E)] [PT,M(EC)].

**Process User:** “I think conversation is a lot different then when you read. Because when you talk, you can look at my face. I can tell your not understanding. I speak to my audience. When I’m writing, I try to make it a little more professional. I am trying to be clear and precise.” [PA,, AD(E)] [PT,M(EC)].

The relationship between process thoughts and ability to describe is also governed by lower level sub codes and axial relationships. In the example above, we see that the process user is recalling (via memory their ability in Treatment B to articulate the process with relative ease) using the use case template in the elicitation context. Overall, the participants found that writing helped to reinforce memory recall, thoughts and articulation.

4.2.3. Process Change, Elicitation and Process Context. As previously discussed in Section 3, we wanted to obtain feedback from our participants with regards to process change because this is a common occurrence in process improvement initiatives. Users are asked to make changes to existing descriptive processes to accommodate new prescriptive guidelines of how the process should be performed. The data collected suggests that participants viewed the change within the context of the performance of the process, not necessarily how the process model would be impacted or changed in the process context. The following excerpts were taken from interview transcripts after the process variation requiring manager approval was introduced in both Treatments A and B:

**Process Engineer:** “How do you think the process will be impacted as a result of this change? Where do you think the process will be impacted?”

**Process User:** “I think it will be negatively impacted because of time constraints. “ [PT,M(PC)] [PC].

In the example above, the participant is reasoning about the process using their memory of the process context. However, the feedback provided relates more to the performance of the process, not necessarily how or where the model will be impacted as a result of the process change. We see this finding again with another participant:

**Process Engineer:** “How do you think the process will be impacted as a result of this change? Where do you think the process will be impacted?”

**Process User:** “You are assuming that I remember what the template looked like? Are you talking about the element name?” [PT,M(EC)][PC]

**Process Engineer:** “That’s ok, if you want to remember the element names then that’s fine…Do you remember them?”

**Process User:** “I remember some of them but not all of them. I think that the process will be slowed down incredibly: “ [PT,M(PC)][PC].

Contrasted to the above we next turn our review to Treatment B. In following excerpt we observe a process user documenting where they thought the change would occur using the use case template as reference. We include a prior cited example from Section 4.2.2 and have bolded and underlined where the process user actually documented that the template would change after the process variation was introduced:

**Natural Language Instruction:** “Please list each step that you take to complete this process””
Process User: “Receive a call from a customer. Determine that there is a problem. Verify contact information. Determine which department would handle said problem and assign to that department. Input the problem in words by the customer into the Long and short description of the ticket. Verify that what was input was correct. The ticket at this point would be forwarded to the OIT Manager for approval and once it is improved then submit the ticket which generates the ticket number” [PD,LoD(E)] [PA, AD(E)] [PT,M(PC)].

The axial relationships between perceptions of process change and the elicitation and process context are relevant given the observations above. Our findings suggest that perceptions of process change may be based upon thoughts, memory recall and whether or not the process user has the reference of a selective abstraction of the process model (in this case, the use case template). Those users that used the template to document the process change generated more direct articulations of how the process would be impacted and were able to generally identify how and where the process model would be impacted.

5. Contributions and Limitations

In terms of assessing contribution, we adopt grounded theory principles to understand if our analysis is well formed given the scope of this study. In accordance with Coleman and O’Connor’s interpretation of Corbin and Strauss’ criteria we ask whether the analysis fits with the narrative theory given the criteria below [8]:

- **Fit:** The theory must fit with the substantive research area and corresponds to the data.
- **Understanding:** The theory makes sense to practitioners in the study area.
- **Generality:** The theory must be sufficiently abstract to be a general guide without losing its relevance.
- **Control:** The theory acts as a guide and enables the person to fully understand the situation.

We believe that the data and analysis fits with the criteria above and has been sufficiently explained with the area of software process elicitation through the grounded theory. This was exploratory research aimed at using grounded theory as a qualitative technique to analyze language and cognitive perspectives of process during elicitation. In this respect, we feel this approach is unique and directly contributes to future empirical research in software process improvement and elicitation.

Additionally, this study identifies how cognitive perspectives may play an important role in elicitation. Analyzing the written texts from the process user using grounded theory is meaningful for software process elicitation in that it has the potential to help researchers understand how the mind organizes and interprets thoughts about the process relative to a selective abstraction of the model. It was hoped that our analysis could provide insights into how the thoughts are directly related to the specific components of a process model. The use case template provided the opportunity to make these structured inferences. Unfortunately, given our sample size, it was difficult to make any inferences relating thoughts and process model elements (e.g. inputs, outputs, exceptions etc).

Due to the interactive nature of the protocol, we recognize that the researcher interventions in either of the treatments could have been a source of potential bias. Biases of these types, however, are impossible to avoid in small, qualitative, exploratory, grounded theory studies such as this one. Moreover, we feel this creative approach adds value for future research because it gives us a framework by which to evaluate our grounded theory narrative. Finally, there are areas for improvement in this line of research specifically in conducting analysis to determine if other elicitation approaches in different research disciplines consider the cognitive perspectives of people documenting process and if our approach can be augmented to include these methodologies.

6. Implications for Future Research

We believe that the majority of people involved in software process improvement want to have an ownership role in describing and documenting process models, especially given the demand placed upon them in practice to respond to process improvement change and organizational cost control. From a practical point of view, in terms of process elicitation, the observations in this study suggest that stakeholders involved in documenting software processes may benefit more directly by using templates to document the process details on their own rather than having consultants do it for them. In our opinion, based upon the grounded theory and fit of the data, it may be useful for us to investigate elicitation methods that directly harness the human potential to articulate, recall and construct process models from long-term memory. We see our future research efforts in software process elicitation strategy requiring a more thorough literature review of those factors in cognitive psychology and cognition and learning theories that inform our observations. Therefore, future work in this area will attempt to build upon the foundations of the grounded theory proposed to determine the extent of generalizability.

7. Conclusion
This study leveraged a use case template to elicit processes at an IT Help Desk. The researcher in Treatment A had difficulty in eliciting accurate descriptive representations according to the interactive observation session protocol. The process was challenging, confusing and inexact partly because the participants shifted between various levels of detail in process description. In Treatment B the participants in the study indicated that using the template to document the process assisted with eliciting and defining the various elements of the process. This may suggest that writing in software process elicitation and using templates as a selective abstraction of the process model assists with memory organization and mental reinforcement.

Our research questions were very open-ended and were intended to help us derive a study protocol. In particular, they were instrumental in generating initial open codes to begin the analysis process, as described in section 3.2. In Figure 3 and section 4 we presented the findings related to these codes, and thus back to the original research questions. Our first research question asked What thoughts are occurring in the process user’s mind while attempting to document a process? We observed that the process user’s thoughts tended to jump between high-level abstractions of the process and specific examples, with a greater ability to remain at the abstract level when using the use case template. This was directly related to our second research question, How do process users describe the process model elements in natural language? Their descriptions of specific elements also reflected this duality between the general and the specific, with a lesser reliance on specific examples when using the template. This theme continues when examining the evidence addressing our third research question, How do process users articulate existing processes? Process users were able to more coherently articulate processes when using the template to organize their thoughts. Our study also found similar findings for answering our last research question, How do process users respond when having to represent changes to existing processes? We noted that participants in Treatment A that did not have access to the template, responded to process change by reflecting on how the performance (execution) would be impacted. However, participants in Treatment B were able to generate more direct articulations of how the process would change at an abstract level when using the template to document the process.

Based upon the research questions we were able to formulate a grounded theory to explain the set of perspectives that were present in the data. We propose that the grounded theory narrative and representation suggest that the use of a selective abstraction of the process model may have a direct impact upon factors that are important for communication in elicitation such as level of detail, the ability of the user to articulate the process and the perceptions of process change relative to the elicitation and overall process context.

The grounded approach that we provide is unique in that we are fitting and evaluating the created theory within the borders of the data alone not based upon our presumption of the applicability of extant research. The use of grounded theory within the context of this exploratory protocol is quite appropriate given the open-ended research questions. We believe that in order to construct the appropriate tools for industry a comprehensive understanding of how people understand and articulate the complexities of process is necessary.

8. References


