Requirements Elicitation via Fit-Gap Analysis: A View Through the Grounded Theory Lens

Tjerk Spijkman[1,2(✉)], Fabiano Dalpiaz[2], and Sjaak Brinkkemper[2]

[1] fizor., Utrecht, The Netherlands
tjerk@fizor.io
[2] Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands
{f.dalpiaz,s.brinkkemper}@uu.nl

Abstract. While requirements elicitation remains a key success factor for software projects, there is little empirical research on the elicitation methods. We focus on *fit-gap analysis*, a requirements elicitation technique that is common in practice, but hardly studied in requirements engineering research. Fit-gap analysis is a method for matching software products with the needs of customers, with the aim to identify needs that are supported as *fits*, and needs that are not as *gaps*. Through a grounded theory investigation of recording transcripts from fit-gap analysis sessions, we provide empirical knowledge about this elicitation technique. We determine and discuss the different categories of the topics contained in a fit-gap analysis. Additionally, as a first step toward assisting analysts in processing and exploring their analyses, we build and share a set of keywords and phrases that can help automatically identify those categories within the transcripts. We conduct an experiment for early validation, involving both students and practitioners, that determines the relative perceived importance of the identified fit-gap categories. Finally, we derive implications for research in the field that include our perspective on how tooling can assist analysts in fit-gap analysis.

Keywords: Requirements engineering · Fit-gap analysis · Grounded theory · Elicitation techniques · Requirements elicitation

1 Introduction

Requirements engineering (RE) is a crucial phase in software and information systems engineering. Reaching a good understanding of the application domain, of the important stakeholders and of the system goals is one of the key factors of preventing project failure (e.g., over budget, cancelled, etc.) [11]. Requirements *elicitation* is concerned with the activities of seeking, uncovering, acquiring and elaborating requirements [26].

M. La Rosa et al. (Eds.): CAiSE 2021, LNCS 12751, pp. 363–380, 2021.
https://doi.org/10.1007/978-3-030-79382-1_22

Although non-conversational approaches to elicitation exist (e.g., surveys or social media analysis [14]), most requirements elicitation is done through conversational scenarios, such as interviews, workshops, laddering [24,26]. This opens up mostly untapped opportunities for *speech-driven RE*: the analysis of conversation contents aimed at detecting and extracting requirements-relevant information.

Fit-gap analysis is a commonly used elicitation method, especially in a business-to-business setting for enterprise applications such as enterprise resource planning systems [2,8,9,12,13]. Fit-gap analysis (FGA) compares the capabilities of a software product and certain characteristics of the target organization [9]. FGA's outputs include the identification of a need for customization [2] or product evolution as new customer scenarios need to be met. Furthermore, the software vendor gathers information to configure the software. Despite its adoption, fit-gap analysis is largely overlooked by the research community.

In this paper, we conduct an empirical investigation of transcripts of FGA elicitation sessions conducted in the software industry. Our study characterizes, through the grounded theory lens [3,22], the key categories of information we could identify in the FGA transcripts. Based on our study and a survey with practitioners and students, we put forward hypotheses for future research.

We make use of FGA transcripts as they represent a comprehensive report of the verbal discussion between business analysts and customer representatives. Our research is enabled by recent trends in AI, particularly in automated speech recognition: several high-quality, off-the-shelf, inexpensive solutions exist for converting audio recordings into text. Our transcripts are generated automatically using Azure Speech to text. Since this paper is concerned with theory building, we manually analyze the transcripts.

As this research takes a grounded theory approach to investigate the contents of a fit-gap analysis session, without making a-priori hypotheses, we define a set of open-ended research questions [3]. We define a main research question and two sub-questions:

MRQ. How do the contents of a fit-gap analysis transcript assist a business analyst in identifying configuration and customization requirements?
RQ1. How to categorize content segments in a FGA transcript in order to support the identification of configuration and customization requirements?
RQ2. What content segments in a FGA deliver the highest value to an analyst?

Through these research questions, we make initial steps in the under-explored area of speech-driven RE, which emphasizes the identification of requirements-relevant information in conversational settings. For example, in elicitation sessions with stakeholders, and other kinds of workshops. In this paper, we focus specifically on transcripts in FGA sessions, and on the categorization of these conversations.

The foundations we lay are meant to guide future works in the speech-driven RE and to create software tooling that assist analysts in reviewing transcripts by highlighting the most relevant content segments for the fit-gap analysis. Automated processing of fit-gap analysis transcripts is aimed to reduce manual effort

(if an analyst would have to manually go through the entire recording), and to improve the FGA process by allowing an analyst to fully focus on the conversation at hand and to minimize note-taking. The automation would also provide assistance in the creation of documentation, for part of the relevant content segments would be identified in the transcript.

Through our research, we make the following contributions to the RE literature:

- We position fit-gap analysis within the landscape of requirements elicitation;
- Through grounded theory, we determine and discuss a categorization of fit-gap analysis, and provide a set of keywords and phrases for identifying such categories;
- We report on an experiment with 7 practitioners and 36 students that assesses the perceived importance of the resulting categories;
- We empirically build a set of hypotheses to guide future research in speech-driven requirements elicitation.

Organization. In Sect. 2, we position fit-gap analysis within requirements engineering. Section 3 presents our research method. In Sect. 4, we discuss the case study and the relevant findings. We report on the validation of the fit-gap categories importance in Sect. 5. We present our implications in Sect. 6, followed by a discussion and outlook in Sect. 7.

2 Fit-Gap in Requirements Engineering

Unlike software that is tailor-made for an organization, software products are developed for a specific market to be sold to many customers [25]. Therefore, there is a need to meet a constant stream of requirements from the market [25]. Software products are in constant evolution, and are in different levels of maintenance. The study by Schach *et al.* [19] revealed that 42.8% were in a state of emergency fixes and routine debugging; 13.8% accommodated changes to the software, 26.8% are actively being improved for user enhancements, and 16.7% did not fall in any of these categories. This indicates that the majority of software products are in a state of change.

Software product vendors need to deal with the heterogeneity in requirements from the market. This often leads to customization of the product, on different levels of granularity. This can impact either a customer-specific version of the software, or the product overall. We use a definition for customization based on the work by Light [13].

Definition 1 (Customization). *Any customer specific change or addition to the functionality available in the standard software product.*

A popular method for managing mass customization is the creation of a product line, a set of products that share similarities and are created from reusable parts [1]. Additionally, many of the products can be tailored to customers through configuration. For this research, we use the following configuration definition from Apel *et al.* [1]:

Definition 2 (Configuration). *Set-up of a software product concerning a predefined set of options used to tailor the software to the customer.*

Configuration supports standardization in a software product while enabling customer-specific deployments. A set of alternative options give the customer an opportunity to configure the product. Unfortunately, designing software for reuse is a notoriously difficult and costly endeavor [12], and even then it typically supports a finite set of options. Another common technique for reuse is *clone-and-own* development, which starts from a template of the product that is customized for customer-specific deployment [17]. Cloning goes beyond configuration, facilitates development, and provides software independence. However, cloning is difficult to manage [4] and software developers often don't know the specifics of the instance they have to work on. Similarly to cloning, other customization techniques such as controller change or interface change can be utilized, each with their own set of challenges [10].

Regardless of the scope and implementation of customization, the underlying requirements need to be identified through elicitation from customer organizations. This is often done through a fit-gap analysis, where current product functionality is mapped against the processes and requirements of existing or potential customers.

In the literature, FGA is mostly studied within the ERP domain, as these are complex software products covering many different business processes that might or might not fit those of the customer [2,9,13]. However, the principles are valid for any software product as they aim to support a process that might differ per customer. As the name suggests, the outputs from a fit-gap analysis can be a *fit* between the customer needs and the software product, or a *gap* in the functionality required by the customer. These fits can be out of the box functionality or configuration requirements. Similarly, gaps indicate customization requirements. Figure 1 visualizes the inputs, process and outputs of a FGA. For fit-gap analysis we use the following definition:

Definition 3 (Fit-gap analysis). *A requirements elicitation technique that, based on matching a customer's needs with the functionality of a software product, identifies needs that are supported by the current functionality as fits, and needs that are not as gaps.*

Identified gaps can be dealt with in different ways. They can be developed specifically for a customer, added to the product road-map [8], or ignored. This can depend partly on the market position for a vendor. Tech giants like Google, Microsoft and Netflix have a strong market position which gives their customers few options for tailoring the software to their processes. Additionally, they have so many customers that they tend to use a data-driven approach in determining their road-map.

In a more general sense, consumer apps only support configuration in most cases, while enterprise applications are often configurable and customized [25]. When the customer has a position in which they can, and see the need to demand

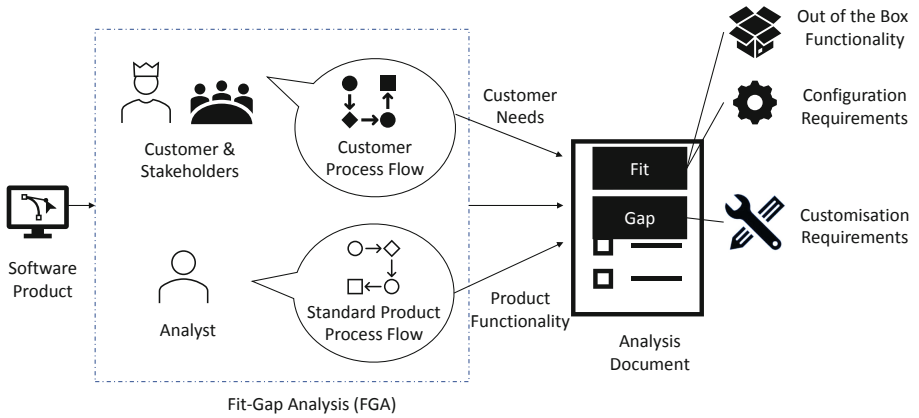

Fig. 1. Fit-gap analysis in the context of requirements elicitation for software products

customization, this needs to be scoped. In these cases, vendors with a less dominant market position may have to meet their customers needs, or convince them to adapt their process to the software to ensure product sales. Often leading back to a FGA where all gaps are identified, resulting in a document with suggestions for mitigating the gaps (e.g., wait for a next release, or use this workaround) or costs for modifying the software.

In our research, we focus on identifying categories and key content segments in these fit-gap analysis. We will discuss these in depth in the following sections.

3 Research Approach

We apply a relatively uncommon research method in information systems engineering: grounded theory (GT). In line with the observations by Stol *et al.* [22], who argue that most articles do not provide enough details about their application of grounded theory, we provide a detailed view of our approach to make research rigor transparent.

While we make no prior hypothesis, as suggested by grounded theory guidelines, our investigation is guided by our main research question (see Sect. 1), which aims to analyze how an FGA transcript may contain content segments that assists an analyst in the identification of configuration and customization requirements.

1. Data Collection. We conduct an industry case study, whose data was made available by having the first author embedded in the case company. This data met our research use case of an industry requirements elicitation with analyst-customer dynamics. The data set contained recordings of the complete FGA performed at the customer. Therefore there was no additional data to gather during the research for this case study.

2. Transcription of Audio Recordings. While we aim to build a low-effort approach that can assist analysts, the time-intensive nature of transcribing contrasted with our goals. We chose to use Azure speech-to-text system, as it supports creation and training of a custom model. We trained this model using 14 sentences with domain terms that visibly changed the performance, especially, with acronyms. Further training of the model is likely to lead to higher accuracy levels. The resulting transcripts of the case study recordings were used for the grounded theory analysis.

3. Grounded Theory Analysis. We follow the guidelines by Corbin and Strauss [3]. This specific iteration of grounded theory was selected as we have a specific research goal formulated in the open-ended research questions in Sect. 1. As suggested by Corbin and Strauss, these are partially inspired by literature as well as our previous research. Within our prior research, we were inspired by: *i.* a study of requirements-architecture alignment, which revealed the need for customization in software products [20,21], and *ii.* automated documentation in medical consultations, which showed feasibility of automated processing of conversations [15].

4. Coding the Data. Data coding was performed using Nvivo 12 by the first author (the Principal Investigator, PI). While coding the data, notes were taken and documented. The set of categories used for the coding was not predetermined, and if a finding did not meet the existing categories, a new (sub)node was added.

5. Memoing. Memo sorting was performed after each coding session, and used as an input for the collaboration described in the next paragraph. The resulting categories, and memos have been discussed on a weekly basis with the other authors. Pieters and Dornig [18] discuss their experience in collaboration in GT, detailing how their approach helped the principal investigator challenge, clarify and support their critical thinking. As they suggest, the PI made the final analytical decisions. These structured discussions on the findings facilitated the continuous comparison of the data and memo's collected during the research. They also provided an opportunity for discussing and determining the central category, and validating if the categorization was still valid for the new data. Finally, part of the analysis has been coded by an independent business analyst, who is not an author, to obtain feedback on the tagging and comparing the coded segments.

4 Fit-Gap Analysis Through the Grounded Theory Lens

We study an on-site FGA that took place over three days, contained in nine separate recordings spanning a total of 12 h with a word-count of 79.938. This analysis was mostly in a one-to-one setting between a business analyst and the process owner at the customer. The FGA focused on the implementation of the software product SCANMAN, an accounts payable automation software. The product is a software solution integrated in the ERP system JD Edwards, and

provides automation for invoice processing and approvals. It is offered through customer-specific installations that are configured to match the processes of the customers, and in some cases customized. The installation is often difficult due to existing customizations in the ERP system.

The case chosen for this research is considered unique and challenging by the case company. The customer had unique scenarios, and was in a state of process transformation. We chose to study this specific case because we expect challenging FGAs to result in documentation where the analyst is likely to miss out on some details. These details could however, be found in the interviews transcripts. In the approximately 250 pages of transcript content, 304 segments were tagged in eight categories and nine sub-categories. A set of 207 FGA keywords and phrases detected in the transcripts, the survey used in Sect. 5, survey results split between students and practitioners and tagging statistics can be found in the online appendix[1]. As it contains confidential information, the full transcripts cannot be shared.

4.1 Central Category: Fit-Gap Analysis Topics

In the execution of the grounded theory process, to *determine the central category*, we went over multiple iterations and potential categorizations of the nodes identified in NVIVO. As categories were identified, or existing categories were changed, the central category was revised. Finally, we decided on the central category that matched our scope and was general enough to able contain all but one of the categories we wanted to focus on, ***Fit-gap analysis topics***.

Each category that specializes the central category is discussed below, and a summary of the categories and their frequency (both in terms of tagged text and number of tags) is presented in Table 1. We leave out *Uncertainty*, an identified category that does not fit the central category. Uncertainty indicates segments in which the speaker seems to have doubts or be unsure of their statements. While this is left out of the scope of this research, we expect that it could be important in the detection of knowledge gaps, completeness of the information, or importance of a statement. We leave this investigation to future research that could focus on the sentiment perspective of elicitation sessions.

1. Current Process Description or Discussion (As-Is). Within the *as-is situation* we distinguish two sub-categories. The general category is for segments that discuss the current processes of the customer, relevant for the software product. In our case study, these were the current accounts payable processes. The *1a. Customer-specific scenario* sub-category was used for segments that were not common use cases within the domain, and either assumed by the customer to be unique for them, or identified as unique by the analyst. These processes are expected not to be covered in the standard use case of the software, and might lead to customizations or process change. Next to that, we use the sub-category *1b. Example* to indicate a specific instance which is described to illustrate the current situation more specifically. For instance, consider the following quote:

[1] Our online appendix can be found at https://zenodo.org/record/4587226.

Table 1. Observed presence of the FGA categories within the case study, and example phrases for the categories (more can be found in the online appendix)

Category	Text tagged	Tags	Example keywords & phrases
1. Current process	31.2%	35.2%	"What we do", "Different for us"
2. Future process	16.8%	18.1%	"Look at", "Our intent is"
3. Explicit requirements	12.0%	8.6%	"We need", "What would be nice is"
4. Questions	4.5%	19.1%	"Could we say", "What happens after"
5. Product functionality	7.2%	12.5%	"Can [product]", "General [product] practice is"
6. Organizational problem	2.1%	4.0%	"Our pain points", "Complicated"
7. Organizational details	0.9%	2.0%	"All of our", "Our company is know for"
8. Product motivation	0.1%	0.7%	"The main reason", "We're looking to"

> *"The IT department has their own ticketing system for their services, for example repairs that need to be done. This means they have their own purchase orders, which are not within the ERP system."*

This fragment has multiple possible implications for the FGA and the documentation. Using the current process, the invoices for IT services do not map to a purchase order (PO) in the ERP, and would have to be configured to go through a non-PO process in the software. Alternatively, the process could be revised to ensure that the PO will be available within the ERP so it can be used in matching scenarios. The current process and its sub-categories were the most occurring segments in the case study, both in frequency (35.2% of tags) and presence (31.2% of text tagged).

2. Future Process Description or Discussion (To-Be). This category is used to denote situations in which the analyst and/or the customer discuss the to-be situation of the customer. This can be due to the use of the software product, or intended and suggested revisions of customers processes. For the to-be process we use a sub-category *2a. Example*, similar to the as-is. Comparing the examples to those of the current process we see that this second category is more conceptual in nature as they denote a hypothetical future. For example, some are introduced with expressions such as *"let's say"*, *"a scenario"*, instead of terms like *"what we do is"*. They can also be differentiated from general to-be

process segments due to the use of employee names, or gender pronouns like *"he"* or *"she"* instead of business unit names or *"we"*. A fragment of a to-be process:

> *"We have the opportunity to have some of our big vendors send us EDI (electronic data interchange) files, or invoice files. So we're debating whether we just let these come in as EDI and load them in our ERP, or to come in as invoice files for automatic processing by [product]. But we're leaning towards having everything run through [product], so everything stays in one place."*

In this quote, the customer prompts the analyst on two alternative future scenarios, aiming to assess which of these would be recommended from the software product point of view. In the transcripts, we see similar examples of how the analyst and customer cooperate to determine this to-be scenario. The to-be categories had the second highest presence in the data-set with 18.1% of tags and 16.8% of text tagged.

3. Explicit Discussion of Certain Requirements. In these segments, the customer states a need regarding the software product. Additionally, we identified the sub-category *3a. Negotiation*, in which customer and analyst discuss the priority and necessity for these requirements. Especially those requirements that we related to potential customization led to negotiation, while configuration requirements were mostly stated and accepted. This is logical from the software vendor point of view, for minimizing customizations is the preferred option. These segments were often indicated by keywords like *"we need"* or *"we want"*, with less obvious keywords including *"would be an improvement"*, or *"what would be good"*. The following requirement, for instance, leads to customization:

> *"What will be really good really, really good is in the email if we have. If someone is going to hit reject in the email they can select the different reject reasons so they can click on one of them and hit rejects and it is moved back to the ERP. And so we know exactly why they rejected it."*

From a software point of view, direct rejection through an email is challenging for both design and technical reasons. For instance, allowing an email link to make direct changes to ERP data is a potential security risk. Additionally, emails are generally static, which makes it difficult to also link this to the different rejection reasons. For this use case of approvals outside of the ERP, the software product has the option to use a mobile application that provides these functionalities. This resulted in a longer negotiation where the analyst was trying to find a way in which the existing functionality might meet the customers need. In the end, this requirement led to a potential customization for this customer, which was scoped and priced. Requirements had a presence of 12.0%, of which 6.2% was negotiation in the percentage of text tagged. Requirements made up 8.6% of tags respectively.

4. Questions Made By or to The Analyst. This category groups three different categories of questions. Questions often overlap the other categories, and

can indicate the start of a conversation on a new topic, or in-depth discussion of an aspect of the current topic. The first category we discern is *4a. Analyst question or prompt*, in which the analyst asks the customer a question, mostly regarding the current and future process. The second category is straightforward: *4b. Customer questions about the product*. Finally, and perhaps less obvious, we have *4c. Customer questions on own process*. These can arise when the customer questions the rationale behind the current processes, or is asking other stakeholders for their opinions and expertise. We hypothesize questions to be very useful in determining the topics and categories of segments of the transcript. For example, consider the following question made by the analyst:

> *"Could you repeat again why we want to have approval for some vendors but not others?"*

This question concerns a certain requirement for the to-be situation, and indicates that the following segment will discuss this. Similarly:

> *"The question is, can [product] read off of an invoice statement like this?"*

Here, the customer makes a question that leads to a discussion of the product functionalities. While questions had a low presence in text coverage (4.5%), although they represented 19.1% of the tags. This occurred since they are typically short and their role is to give the discussion a different direction.

5. Elaboration on the Existing Functionality of the Product. This category is used to collect segments in which the software product functionalities are discussed. The analyst might bring up configuration (sub-category *5a*), or give an example of certain functionalities (sub-category *5b*). The goals include elaborating on any uncertainties the customer might have, discussing functionalities that need to be configured in one way or another, or managing expectations about those that can be met by the software. An example of expectation management can be seen here in the following fragment:

> *Customer: "I think in the demo it was said you can train the system. E.g., when you see the words tax, freight or whatever, it will automatically add a line."*
> *Analyst: "How it works. It will check the tolerances you set ùp for voucher match automation. So from there it will check is it within my bounds to automatically accept this and if so process it. If this shipping makes it go above this percentage or total limits that you set, e.g., so say they charge you 600 for shipping while your gap is 300, it will fail to match."*

We see that there are limitations to "automatically" adding a line, based on tolerances to prevent over-billing. Another observation from the customer's comment is that the demo of the application has created certain expectations and assumptions regarding the software product. In this example the analyst manages the expectations by providing further details about the functionality, and discussing its capabilities. These segments made up 7.2% of the tagged text and 12.5% of the tags.

6. Organizational Problem Explanation. In these segments, the (potential) customer explains some root causes for their challenges or problems in their organization. These can impact the implementation of the software, and can for example be due to social factors. For instance, resistance to change due to the impact on jobs. Or it may have to do with the domain and circumstances of the customer, e.g., complex legal requirements. For confidentiality reasons, we provide generalized examples as quotes can be of a sensitive nature. These had a presence of 2.1% of the text, and 4.0% of tags.

7. Organizational Details. This category is used to denote segments in which the customer introduces or talks about their organization, providing context about the setting in which the product will be used. These details include the domain or sector as well as the scale of their operations. For example:

> *"We have offices in multiple European countries, including the Netherlands, Germany, France and Ireland."*

This can impact the configuration, for instance, because different tax setups are required for the customer. Note that such information can often be found outside of the analysis context, e.g., on the company website. They give the analyst a better view of the customer, and we found them mostly in the early segments of the analysis. This category had an overall presence of 0.9% in text with 2.0% of the tags. While this is true for our case study, we expect that these segments will have relatively higher presence in an analysis with shorter duration.

8. Motivation for Using or Acquiring of Software Product. This category refers to segments in which the customer actively discusses their motivation for acquiring or considering the software product. This might be because it helps them mitigate a problem they face, or because of an advantage compared to other competing products.

> *"The main reason we're looking to [product] is not only to reduce amount of time it takes to enter invoices in the system, but also the approval process."*

These quotes can help during prioritization, and they provide the software vendor with further understanding of the customer's needs. They can also aid an organization in defining their product road-map, and focusing their sales pitches. These segments were not very common in the case study, with less than 1% presence in text and references. However, we did include these as they might play a bigger role in other cases.

5 Validation of the FGA Categories

The eight identified FGA categories were validated through a survey filled out by both industry experts and students. The students took the survey as part of a workshop in a master's level Requirements Engineering course. After a short introduction on the research, fit-gap analysis goals, and context of the software product, the survey participants were asked to conduct two separate tasks:

1. To rate the importance of two textual transcript fragments for each of the eight categories on a four point scale (essential, important, not-important or irrelevant), without being informed of the categories at this stage. A four-point scale was used to minimize desirability bias [7]. In total, the participants rated a total of 16 fragments randomly selected out of a set of 50 fragments.
2. To provide a total order of the importance of the eight categories, based on their perception. Participants could drag the categories in their preferred sequence from high to low importance.

This allows us to conduct triangulation; our findings concerning the importance of the categories are therefore based on three sources: the participants' evaluation of fragments, the participants' ranking of the categories, and our own experience. The survey was filled in by 36 students and 7 industry practitioners who were familiar with the product domain. The average time spent on answering the survey was 16 min.

The box-plot chart in Fig. 2 illustrate the perceived importance of the categories of our 43 respondents from the second survey task. In the x-axis, the value 1 represents the highest rank (the most important category), while the value 8 represents the lowest. The category means clearly show two groups. Of these the current process, future process, and explicitly discussed requirements are consistently ranked as the most important categories.

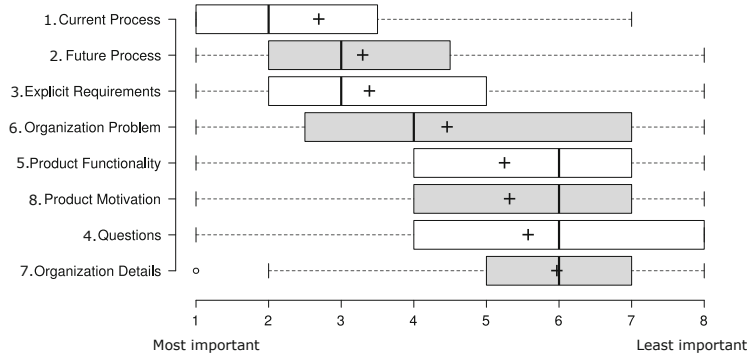

Fig. 2. Box-plot chart showing the perceived importance of the categories, derived from the ranking provided by the participants in survey task 2 (n = 43).

When we look at the rating of the transcript fragments in the first task, shown in Fig. 3, we see similar results to those in the category rankings (Fig. 2). This confirms the importance of the three categories that had the highest rank: current process, future process, and requirements. The "questions" category seems to be an outlier: its fragments are rated as quite important, while it is not highly ranked as a category. We reckon that this may be due to either the selection of the questions, or the formulation of the category name, which was slightly ambiguous for some participants.

Comparing the student answers to those of the practitioners for the importance of the *fragments*, we draw four main observations regarding the FGA categories:

- **Current process:** *Practitioners judged fragments belonging to the current process as more important than the students.* With 86% of the ratings being positive, compared to 68% by the students. Practitioners rated fragments in this category most positively out of all categories, while in the student rankings it comes in 4th place. We hypothesize that this might be because the students lack the domain knowledge to identify the current process in the fragments.
- **Product motivation & customer context:** *Students rate both product motivation and customer context less importantly than practitioners.* Practitioners rated the fragments positively at 43%, and 36% respectively, while students rated these positively at 65% and 49%. The product motivation, also had the highest count of "essential" ratings in the student survey. This discrepancy may arise because practitioners have access to these details via interactions that precede the FGA.
- **Questions:** *The students rated the questions fragments more positively than practitioners.* The questions received 76% positive ratings by the students, making this the second most positive ranking out of all categories. For practitioners, this was 57% positive, leading to a shared fourth ranking out of the categories.
- **Positive answers:** *In general, the students rated more of the fragments as positive than practitioners.* They had an average positive answers over the categories of 64% while the practitioners had 57%. This may be ascribed to the fact that practitioners are more aware of which information can be found from other sources.

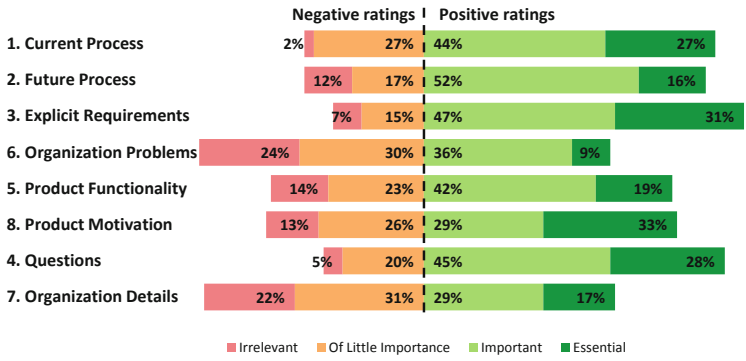

Fig. 3. Diverging stacked bar chart of the transcript fragment ratings in survey task 1

6 Implications for Research and Practice

This research aims to create, in an empirical fashion, the foundations for assisted transcript exploration. Through grounded theory, we provide empirical evidence and minimize assumptions on important content segments within FGA. Based on our observations, we form hypotheses for future work in this research field.

For the FGA categories we identified, the perceived rankings indicated that three of these are more important than the others for an analyst. While the student ratings of the transcript fragments have a discrepancy as questions have the second most positive rating, the remaining categories and practitioner ratings support the following hypothesis:

H1 The most important categories for processing a FGA into documentation are 1) the current process, 2) the future process, and 3) explicitly discussed requirements.

Therefore, these categories should be the focus in generating automated documentation as well as in any tooling that can interactively assist an analyst to explore transcripts.

Only a few requirements were *explicitly* mentioned in the transcripts. The main focus of the analysis instead was on the current situation. Thus, we posit:

H2 The analyst uses bits and pieces of information gathered in the analysis to specify the requirements for the project. Such information compares the *current process* with existing *product functionality* to determine the *future process*.

For future research, it is important to investigate to what extent the resulting requirements in a specification can be traced back to various segments of the transcript, and to identify the sources for requirements not covered in the transcript.

Specifically looking at the configuration requirements in the transcripts, we observe that these are mainly discussed together with the product functionality, driven by the analyst. For example: "So the product can support approvals in multiple ways, which of these would you prefer?". This can then be discussed as part of a future process for the customer. Additionally, there are segments of the current process that inform an analyst how the software should be configured to minimize the process changes.

H3 Configuration requirements can mainly be found in transcript segments within the categories on product functionality, current and future processes. Specifically when discussing processes within the product software domain.

From anecdotal experience, configuration requirements are generally covered in the analyst's notes. However, detecting these in the transcripts can decrease the time spent on digitizing notes and reduce the chance of missing configuration requirements.

In contrast, customization requirements are more often part of future process aims that cannot be met by the software product. Generally, they find their motivation in current processes, as the customer prefers not to change part of the process. Or in processes unique to the customer's domain. These unique processes are likely to result in customization, as rationale for exclusion is less likely. Therefore, we posit:

H4 Customization requirements can be detected from future process discussions that do not match the product functionality. Their justification can often be found in current processes, especially for unique processes.

Detecting customer domain knowledge and information about the product functionality is key in recognition of customization requirements. Also, domain knowledge seems to impact the recognition of current process, as observed in the tagging differences. Therefore, future research should take the role of domain knowledge into account, both to conduct experiments as well as for building automated tools.

7 Discussion and Outlook

Summary. In this grounded theory study, we investigate and position fit-gap analysis as a source of requirements. We performed extensive coding of a real-world data-set and empirically discovered eight categories to categorize the contents of a FGA. We discussed each of these categories through examples, and use the findings to build hypotheses to guide further research in the speech-driven requirements engineering field.

We answer our RQ1 on the categorization in FGA through the eight categories presented in Sect. 4. RQ2 on the most valuable content segments in FGA has been discussed through the experiment in Sect. 5, which indicates current process, future process, and discussed requirements as most valuable.

To support further research in speech-driven requirements engineering and to allow other researchers build on our work, we made publicly available, in a persistent repository, the keywords and noun phrases identified in the categories. Though we acknowledge that this is only an initial set, we expect this can be used to build NLP-powered tools that can recognize the categories. Our dataset also includes frequency and tagging statistics to support replication.

Vision. We answer the MRQ by putting forward our vision. Our findings make us surmise that a fully automated approach is unlikely. Many decisions need to be made while processing a fit-gap analysis into a fit-gap document that guides the rest of the software project. Most of these decisions cannot be traced back to the FGA transcript, as they are essential part of the thinking process of the business analyst during requirements engineering. We see the processing of FGA as an example of a non-algorithmic RE task, using Tjong and Berry's words [23].

We envision an interactive tooling that presents an analyst with relevant content segments, and an overview of the topics that are discussed in the analysis sessions. From there, the analyst can determine which of these are interesting for the document they are working on, and explore relevant segments in the transcripts. This perspective of investigating conversation contents is shared with other domains, prominently healthcare [5,15]. Future research is necessary to determine similarities and differences.

Panichella and Ruiz [16] share a similar, yet different, vision. They argue for the use of machine learning and ontology crawlers for automatic requirements documentation. While we are too interested in the use of automation, our empirical observations indicate that only few requirements are explicitly mentioned. Therefore, at least in the context of fit-gap analysis, automation will have an important yet only partial role.

Validity. Due to the exploratory nature of this research, there is a limitation to *external validity*. Despite its size (12 h of recordings), we conducted a single case study; therefore, it remains to be validated whether the findings can be generalized to different software products, domains and elicitation techniques. For this reason, we only build hypotheses from our findings as opposed to forming conclusions. To prevent *conclusion validity* issues in our results, we have opted to only share observations from the data collected from the survey as opposed to statistical conclusions. Both students and practitioners participated in the validation of our research. Using students as a proxy for practitioners in software engineering is often a point of discussion [6]. While there were similar results in the perceived rankings of category importance, we did see differences in the ratings of the segments. These are expected to be mostly a result of difference in domain knowledge, and to a lesser extent experience. We minimized *construct validity* threats by randomizing the initial category order, giving participants a random subset of fragments and not explicitly showing to which categories the fragments belong.

References

1. Apel S., Batory D., Kästner C., Saake G.: Software product lines. In: Feature-Oriented Software Product Lines, pp. 3–15. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37521-7_1
2. Blick, G., Gulledge, T., Sommer, R.: Defining business process requirements for large scale public sector ERP implementations: a case study. In: Proceedings of the ECIS, p. 157 (2000)
3. Corbin, J., Strauss, A.: Basics to Qualitative Research, 3 edn. SAGE (2008)
4. Dubinsky, Y., Rubin, J., Berger, T., Duszynski, S., Becker, M., Czarnecki, K.: An exploratory study of cloning in industrial software product lines. In: 17th CSMR, pp. 25–34. IEEE (2013)
5. Epure, E.V., Compagno, D., Salinesi, C., Deneckere, R., Bajec, M., Žitnik, S.: Process models of interrelated speech intentions from online health-related conversations. Artif. Intell. Med. **91**, 23–38 (2018)

6. Falessi, D., et al.: Empirical software engineering experts on the use of students and professionals in experiments. Empir. Softw. Eng. **23**(1), 452–489 (2017). https://doi.org/10.1007/s10664-017-9523-3
7. Garland, R.: The mid-point on a rating scale: is it desirable. Mark. Bull. **2**(1), 66–70 (1991)
8. Grabis, J.: Optimization of gaps resolution strategy in implementation of ERP systems. In: Proceedings of the of ICEIS, pp. 84–92 (2019)
9. Gulledge, T.R.: ERP gap-fit analysis from a business process orientation. Int. J. Serv. Stand. **2**(4), 339–348 (2006)
10. Jansen, S., Houben, G.-J., Brinkkemper, S.: Customization realization in multi-tenant web applications: case studies from the library sector. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 445–459. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13911-6_30
11. Jones, C.: Software project management practices: failure versus success. J. Defense Softw. Eng. **17**(10), 5–9 (2004)
12. Kuo, T.C.: Mass customization and personalization software development: a case study eco-design product service system. J. Intell. Manuf. **24**(5), 1019–1031 (2013)
13. Light, B.: The maintenance implications of the customization of ERP software. J. Softw. Maint. Evol. **13**(6), 415–429 (2001)
14. Maalej, W., Nayebi, M., Johann, T., Ruhe, G.: Toward data-driven requirements engineering. IEEE Softw. **33**(1), 48–54 (2015)
15. Lepenioti, K., et al.: Machine learning for predictive and prescriptive analytics of operational data in smart manufacturing. In: Dupuy-Chessa, S., Proper, H.A. (eds.) CAiSE 2020. LNBIP, vol. 382, pp. 5–16. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49165-9_1
16. Panichella, S., Ruiz, M.: Requirements-collector: automating requirements specification from elicitation sessions and user feedback. In: Proceedings of the RE, pp. 404–407. IEEE (2020)
17. Pérez, F., Ballarín, M., Lapeña, R., Cetina, C.: Locating clone-and-own relationships in model-based industrial families of software products to encourage reuse. IEEE Access **6**, 56815–56827 (2018)
18. Pieters, H.C., Dornig, K.: Collaboration in grounded theory analysis: reflections and practical suggestions. Qual. Soc. Work **12**(2), 200–214 (2013)
19. Schach, S.R., Jin, B., Yu, L., Heller, G.Z., Offutt, J.: Determining the distribution of maintenance categories: survey versus measurement. Emp. Softw. Eng. **8**(4), 351–365 (2003)
20. Spijkman, T., Brinkkemper, S., Dalpiaz, F., Hemmer, A.F., van de Bospoort, R.: Specification of requirements and software architecture for the customisation of enterprise software. In: Proceedings of the RE Workshops, pp. 64–73. IEEE (2019)
21. Spijkman, T., Molenaar, S., Dalpiaz, F., Brinkkemper, S.: Alignment and granularity of requirements and architecture in agile development: a functional perspective. Inf. Softw. Technol. **133**, 106535 (2021)
22. Stol, K.J., Ralph, P., Fitzgerald, B.: Grounded theory in software engineering research: a critical review and guidelines. In: Proceedings of the ICSE, pp. 120–131 (2016)
23. Tjong, S.F., Berry, D.M.: The design of SREE – a prototype potential ambiguity finder for requirements specifications and lessons learned. In: Proceedings of the REFSQ, pp. 80–95 (2013)
24. Wagner, S., et al.: Status quo in requirements engineering: a theory and a global family of surveys. ACM Trans. Softw. Eng. Methodol. **28**(2), 1–48 (2019)

25. Xu, L., Brinkkemper, S.: Concepts of product software. EJIS **16**(5), 531–541 (2007)
26. Zowghi, D., Coulin, C.: Requirements elicitation: a survey of techniques, approaches, and tools. In: Aurum, A., Wohlin, C. (eds.) Engineering and Managing Software Requirements. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-28244-0_2