Uncertainty expressions in software architecture group decision making

Explorative study

Klym Shumaiev
Technical University of Munich
Boltzmannstr. 3
Garching b. Muenchen, Germany
85748
klym.shumaiev@tum.de

Manoj Bhat
Technical University of Munich
Boltzmannstr. 3
Garching b. Muenchen, Germany
85748
manoj.mahabaleshwar@tum.de

Oleksandra Klymenko
Technical University of Munich
Boltzmannstr. 3
Garching b. Muenchen, Germany
85748
alexandra.klymenko@tum.de

Andreas Biesdorf
Siemens CT
Otto-Hahn-Ring 6
Munich, Germany 81739
andreas.biesdorf@siemens.com

Uwe Hohenstein
Siemens CT
Otto-Hahn-Ring 6
Munich, Germany 81739
uwe.hohenstein@siemens.com

Florian Matthes
Technical University of Munich
Boltzmannstr. 3
Garching b. Muenchen, Germany
85748
matthes@in.tum.de

ABSTRACT

Software architecture can be seen as a set of architectural design decisions (ADDs) that shape the resulting software solution. To make an ADD, stakeholders follow some organization- or team-specific group decisions making process. In this study, we aimed to advance the understanding of how ADDs are made by observing and learning how architects handle uncertainties in real-life settings. We employed a multiple-case studies research method. First, we examined the discussions in task management systems of three software engineering projects. Second, we conducted interviews with the projects' software architects to investigate (a) uncertainties expressed in the observed discussions and (b) how those uncertainties are comprehended by their respective authors or readers. We systematically analyzed the interviews and derived different types of uncertainties as well as proposed a hypothesis that should be verified in the future work. Results of our qualitative study show how uncertainty is used and perceived by the software architects in the group decision-making process.

CCS CONCEPTS

• **Software and its engineering → Software implementation planning**; *Agile software development*; Software maintenance tools;

KEYWORDS

Uncertainty, software architecture decision-making, group-decision making, architecture sustainability

ACM Reference format:
Klym Shumaiev, Manoj Bhat, Oleksandra Klymenko, Andreas Biesdorf, Uwe Hohenstein, and Florian Matthes. 2018. Uncertainty expressions in software architecture group decision making. In *Proceedings of The 4th Workshop on Sustainable Architecture, Madrid, Spain, September 2018 (ECSAW'18)*, 8 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ECSAW'18, September 2018, Madrid, Spain
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A software architecture (SA) can be seen as a set of architectural design decisions (ADDs) that shapes the resulting software solution [9]. The industrial case-studies [20, 25, 26] show that architectural decision making (ADM) is a group activity. To make an ADD, stakeholders follow some organization- or team-specific group decisions making (GDM) processes [28]. An important step in GDM includes establishing a shared understanding of the goals, the requirements, the design problems, the existing system's behavior, architectural elements, and context such as assumptions, constraints, and tradeoffs. Moreover, studies [25, 26] show that teams of architects and developers predominantly use unstructured approaches, in particular, **discussions** to arrive at consensus during ADM.

On the one hand, choosing discussions seems to be intuitive for stakeholders while making group decisions; it does not require any initial training for the participants and everyone is familiar with it as they would have experienced it in daily lives. Discussions provide a lot of freedom to the participants and the lack of any fixed structure allows them to think openly and communicate without being constrained by any decision-making process. On the other hand, freedom comes with cost; making group decisions can be less effective and can impair the quality of a decision by introducing flaws like (1) *information asymmetry*: discussions do not encourage any systematic participation of the stakeholders, which can bias the information shared in the group [4]; (2) *lower accountability*: acceptance of excessive informality in groups due to the absence of information management and control protocols [24]; 3) *architectural knowledge evaporation* [26].

One way to improve the GDM process in SA is to adopt structured approaches or introduce some structure in certain phases of discussions (for example, facilitating stakeholders to rely on an expert or a tool to channel a discussion in a structured manner). To enable this shift, researches see potential in conducting more empirical studies, that help to deepen the understanding of industrial

GDM practices and specific needs that can enhance the effectiveness of the decision-making process through tools, GDM methods, structural changes in groups, etc.

In our empirical study, we focus on a particular phenomenon, **expressions of uncertainty** in the process of architectural GDM. We borrow the definition of uncertainty from the work of Jordan et al. [10], where they refer to expressed uncertainties as situations when individuals have a sense of wonder, doubt, or unease about how the future will unfold, what the present means, or how to interpret the past. For instance, during the process of GDM in SA, stakeholders can be uncertain about possible consequences of the changes made to the architectural elements, the likelihood of external environmental changes (hosting infrastructure or users' input); they can be unsure about their knowledge or understanding, can be ambivalent about their choices, preferences, or values; and can perceive that they hold incomplete, contradictory, ambiguous, or untrustworthy information. Furthermore, since uncertainty in human life is ubiquitous, multifaceted, and unavoidable, the discussions during GDM in SA would also capture these feelings, shaped and revealed in the form of natural language.

The goal of our study is to identify uncertainty expressions in dialogs of stakeholders during the GDM process, and document writers' intention in using uncertainty expressions, and readers' perception of those uncertainties. The results of our exploratory empirical study can be reused in order to derive strategies for designing approaches and tools, that guide stakeholders towards a structural GDM process in SA.

2 RELATED WORK

There is a plethora of studies in the context of software development that proposes approaches to deal with uncertainties. Due to the broadness of the term, we explicitly differ our work from studies [7, 14], that approach uncertainty as a quantifiable property, that can be simulated in the early phases of requirements engineering and ADM process. In theory, if such approaches were applied to a GDM process, then they would introduce structure in GDM.

H. Yang et al. [30] focus on speculative statements in requirements captured in natural language. Results of their work show that automatic identification of uncertainty expressions in requirement specifications can be done with a high accuracy (more than 80%). In this regard, authors point out that future research should help to understand what effect uncertainty has on requirements' (mis-)understanding, do stakeholders have to clarify speculative requirements as early as possible or just leave it "as-is" for architects and developer to have a larger design space to explore solutions.

In the same work, authors present advances made by natural language processing (NLP) communities in detecting uncertainty expressions for different domains [16, 18]. Moreover, we found similar evidences in more recent works [15, 23]. Nevertheless, to the best of our knowledge, there is no work on automatic detection of uncertainty expressions in the artifacts of ADM or GDM containing textual information.

C. Yang et al. [29] draw SA research community's attention to the phenomenon of architectural assumptions (AAs) and their management in industry. In their work, authors interview 21 architects and conduct a set of workshops to explore the role of AAs in designing SA. Authors speculate that uncertainty could be a key element of an AA and its elimination might transform these AAs into ADDs. Authors conclude that there are no approaches or tools, which help architects to manage AAs. Thus, we see our work as a step towards understanding how AAs can be identified and managed.

The study performed by Jordan et al. [10] shows how discussions are formed and the role uncertainty expressions play in discussions. In their exploratory study, researchers observe students interacting in a computer-mediated environment to construct an understanding of new concepts from the course readings. Results of the study demonstrated that students enjoyed discussions and had a belief that it contributed to their learning. Contrary to the opinion that learning is a process of reducing uncertainty, authors propose to treat learning as the process of cultivating uncertainties in discussions. Authors explain how students during their discussions managed to create an open space for idea exploration using uncertainty expressions: *"The expression of uncertainty possibly enabled computer-mediated discourse participants to play with new intellectual ideas without running into the risk of sounding like a know-it-all, of being impolite, or being held accountable for their claims."*.

3 APPLICATION OF A MULTIPLE-CASE STUDIES DESIGN

We followed a multiple-case studies design adopted from the work of Yin et al. [31]. The strength of the multiple-case studies design lies not only in its ability to demonstrate consistent patterns of behavior but also, and perhaps more importantly, in its ability to uncover new divergent themes [32]. Exploration of these themes powered by the replication process, which is a cornerstone of multi-case studies design and can be seen as conducting a set of independent experiments on related topics [31]. In contrast to a single case study design, the presence of the replication process in multiple-case studies allows researchers not only to observe the phenomenon in unique settings but also to build a preliminary theory that describes a phenomena [5]. We performed our study in eight consequent steps as illustrated in Figure 1 and elaborated those steps in the subsequent subsections.

3.1 Determining study boundaries

Through a survey [26], Rekha and Muccini showed that 87% of the respondents acknowledged that there is an omission of major decisions and 40% of them mentioned that such omission happen quite often. Such responses indicate that quite often, ADDs in projects stay implicit, which makes it difficult not only to identify final decisions, but also to recover the process of how those decisions were made. Thus, to study the GDM process in SA, researchers have to be real-time observers, participating in face-to-face meetings between stakeholders [27], study chat communications [1], or explore the content of task management systems (TMS) [3], and its relationship to the source code[21]. In our study, we focus on task management systems due to several reasons: 1) there is evidence of their wide adoption within industrial and open-source software engineering projects [3, 22]; 2) discussions within TSM are chronologically organized around specific problems and often contain records of communication between architects and other

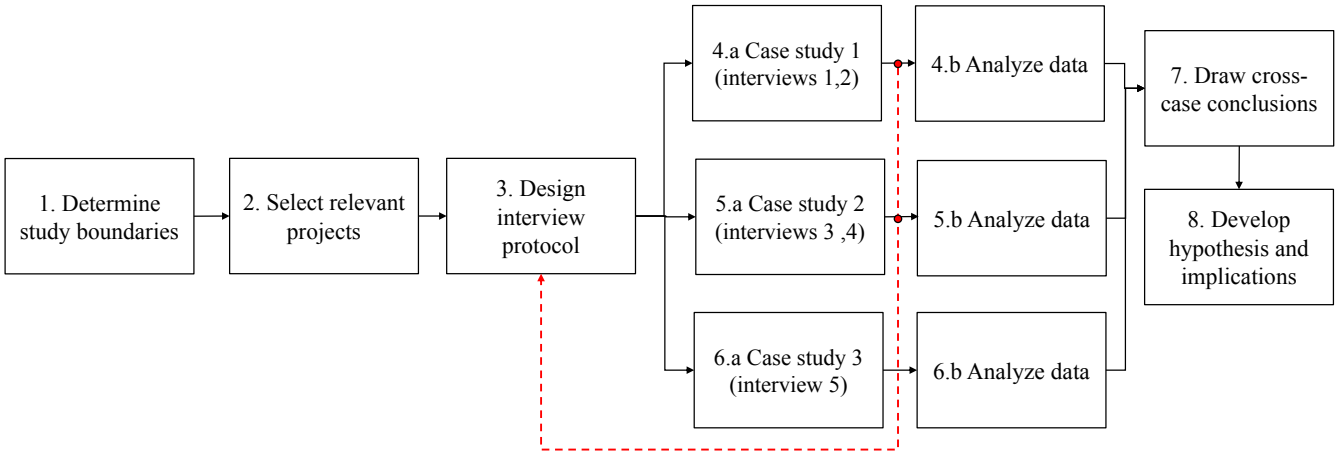

Figure 1: Steps in a multiple-case studies design

stakeholders; 3) relative accessibility of TMS in comparison to other artifacts (like, emails).

3.2 Sample selection

The selection of cases for further investigation was restricted to software engineering projects, where 1) software architects involved in the software engineering projects were ready to participate in a qualitative study; 2) TMS was used during the development process and we were able to access its data; and 3) tasks in TMS contained common attributes that relate to the decision rationale and contributed to the final ADD.

While first two restrictions were dictated by the willingness of practitioners to cooperate with us, the third one required a preliminary study of the content in each of the project's TMS. Two authors of this paper used filtering and sorting functionalities of the TMS to check if candidate projects had tasks consisting of textual description and stakeholders discussions around them. Tasks were classified as leading to ADDs, if their textual description or any comment in the task matched the classification rules summarized in an ADD annotator guideline [1] which was also used in our previous work on automatic extraction of ADDs [3]. For example, a task containing discussions should be labeled as an ADD, if stakeholders discuss adding or removing plugins/libraries, altering the process flow or the functionality of certain components, adding/removing classes or methods, etc. Furthermore, English had to be the main language of communication in TMS. As a result of our preliminary study of the available projects' TMSs, only three projects (described in Section 4) met the sample selection criteria.

3.3 Data collection

Each of the selected projects were considered as a unique environment for an individual case study. Each case study has a number of interviews with project stakeholders who were involved in ADM in those projects. To initiate the first case study (step 4a in Figure 1), an initial interview protocol was designed (step 3) which resulted in

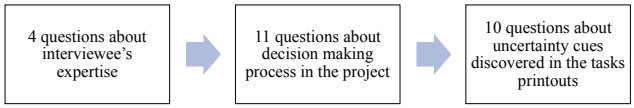

Figure 2: Interview blocks

three interview blocks (cf. Figure 2) comprising of 25 questions[2]. In total, 5 interviews with practitioners were conducted sequentially by the first and the third authors of this paper.

The first interview block contained questions about the professional experience of interviewees, how long they have been involved in the project, and what their role in the project. The second block included questions about how many developers are involved in the project and what communication means they used, how decision-making process was organized, and how elaborate were technical specifications in the projects. The first two blocks of interviews helped us to study how GDM process was organized and how practitioners perceived their role and roles of their colleagues during discussions.

In the third interview block, we aimed to discuss with interviewees particular occurrences of uncertainty expressions in the ADM process. Therefore, tasks from each project were filtered and sorted. Priority was given to those tasks where an interviewee was actively participating and contained relatively a large number of comments. Depending on the amount of comments in the task, the number of printed tasks per case study differed from 3 to 6 (due to the constraint that only a few tasks could be discussed with interviewees in a limited amount of time). Eventually, all selected tasks from specific projects were printed twice before each interview. Final printouts contained textual description of the task and discussions around it; all meta-information related to the task, for example, type of the task (bug, feature, blocker), priority, creation date, authors of the task, assignee, and commentators were also included.

[1] https://wwwmatthes.in.tum.de/file/39cg0w3tgji7

[2] https://wwwmatthes.in.tum.de/file/1p3sf2cd0ak81

The second copy of each tasks' printout was used by two authors of this paper to manually annotate the occurrences of uncertainty in textual descriptions and comments. The annotation process adhered to a guideline used in the study on linguistic uncertainty [6]. For instance, a statement must be labeled as "uncertain" if it contains cues that match one of the uncertainty cues from the set compiled by Hyland et al. (e.g., guess, think, probably, etc.) [8]. A real-world example of such an uncertain statement can be: *"I'm assuming that this change was unintentional, so I **think** we should **probably** revert."*. Another rule is that questions inherently include uncertainty and, therefore, they were not annotated as uncertain statements.

At the beginning of the third interview block, printouts without annotations were given to the interviewees. We asked interviewees to manually annotate statements, that in their opinion contained uncertainty and to justify why they considered them as uncertain (**Task 1**). To avoid study biases, interviewees were not provided with any definition of the term "uncertainty" beforehand. As soon as the annotation process was completed, interviewees were given the second printout of the same task containing already highlighted annotations. In this way, interviewees were able to compare both the printouts (**Task 2**) and we had a chance to ask the following questions:

- How do they interpret uncertainty cues that hadn't been highlighted by them?
- How do they interpret their own uncertainties in the comments or task descriptions?
- How do they interpret uncertainties expressed by their teammates in the same task?
- Would they rephrase statements that contain uncertainties?
- How did the expressed uncertainties influence their understanding of the problem?
- If and how did the expressed uncertainties influence their decisions in the context of current task?

Neither the interview questions nor the printouts were provided to the participants before the interview, except for interview 5, where the questions catalog was forwarded to the participant due to legal reasons. For 3 out of 5 interviewees, it was possible to find examples of their own comments which included uncertainty cues, so that some first-hand feedback could be obtained and later compared with the teammate's perception of the same uncertainty cue. More time was spent on discussions related to the project's organization and how ADDs are made within projects, rather than on the exploration of uncertainties in tasks. This was caused by the eagerness of interviewees to dive deeper into the topic and authors' conscious unwillingness to interrupt them. As a result, interviews took longer time than planned - around 80 minutes on an average.

3.4 Data analysis

An important feature of the multiple-case studies design is that it naturally enforces the employment of a "constant-comparative" method, which is characterized as an iterative process of data collection, analysis, comparison, and revision during the entire study [2]. The replication strategy helps to identify possible patterns in data and explore them by returning to the field for more data. In Figure 1, the iterative process is illustrated using dashed lines.

Following our study design, after each interview, we revised the notes that had been taken during the interviews and extended them by transcribing interview highlights captured in audio recordings. Information was codified and linked to the printouts used during the interviews. As a result, observations made after each interview could be reused for asking additional or clarifying questions in the upcoming interviews. In case interviewees were participants of the same case study, we were able to omit some general questions about the project's organization in the later iterations.

As soon as all the interviews were performed, all transcripts were reviewed again, common responses were grouped and used as evidence to support the derived hypothesis described in the next Section. We counted all uncertainty expressions annotated by practitioners during the interviews and the number of sentences in tasks' descriptions, and discussions around them.

4 FINDINGS

As illustrated in the Figure 1, we conducted three case studies. In each case study, we analyzed specific software engineering projects with unique organizational and development environments as described below.

4.1 Case study 1 (CS1): A healthcare management system

The first two interviewees had been involved in the development of a healthcare management system for the European hospitals. At the time of the interviews, the duration of the project was eighteen months, the project team consisted of ten contributors spread over six European countries, and used one shared TMS, in particular, a Jira instance containing 318 tasks. For the interviews, we got in touch with two of the contributors, who were located in the same office and were responsible for the development of the data storage layer for the entire system. The other eight contributors were located in five other cities and had to rely on the work performed by those two contributors, since the data layer was the most critical and integral part of the system.

Interviewee 1 is a software architect. With five years of professional experience, he has been involved in this project since its initiation. As the team is geographically distributed, a lot of effort was spent on clarifying with other contributors how the communication protocol between the clients and the storage layer should be implemented. At the beginning of the project, all communications were done via e-mails. Later, the contributors switched to a web-communication tool for a more direct communication with less waiting time. The TMS had been introduced to the project around three months before the interviews were held and therefore, contained a rather small amount of tasks. According to Interviewee 1, currently, there is a disagreement on how to use the TMS and how to document tasks in the project. Team members are encouraged to put all the tasks in TMS, however, this is usually done after preliminary discussions via a web-conference tool. Nevertheless, we were able to identify tasks, where Interviewee 1 was contributing to the discussions that led to ADDs.

Interviewee 2 with six years of professional experience performs the role of a software developer and has been working on this project for the last twelve months, supporting Interviewee

1 in the development phase. In his case, we were able to detect tasks where he was responsible for the implementation but was not taking part in discussions within those tasks. Therefore, we were not able to identify any discussions in TMS, where he himself was personally involved, though there were tasks assigned to him and he was responsible for them. As a consequence, to observe how Interviewee 2 would perceive uncertainties expressed by his teammate, we asked Interviewee 2 to annotate and comment on uncertainty expressions in the tasks where Interviewee 1 had been involved in the discussions.

4.2 Case study 2 (CS2): A platform for mobility services

The goal of this software engineering project was to develop a software platform that enables building innovative mobility services for urban spaces (for example, providing optimal mobility options for individuals and reducing traffic congestion in cities with more than a million inhabitants). This project had lasted for two years. During that time period, the number of team members in the project grew from six to fourteen developers. However, at the time of the interview, it had decreased to seven contributors and most of them were collocated in the same office. The team used Jira as a TMS, which contained 1,232 tasks.

Interviewee 3 performed the role of a software developer and had been involved in making ADDs within the project. He had seven years of professional experience and had joined the project a year ago.

Interviewee 4 joined the project six months ago and performed the role of a software architect with four years of professional experience.

Mostly, ADDs within this project were made in face-to-face meetings. If the participants were not available in the same office, then a web-conference tool was used. Interviewee 4 mentioned that there was a Wiki page for each of the architecture element. After discussions, each team member responsible for a specific architecture element would document the ADDs in the corresponding Wiki pages. However, our request to access those Wiki pages was declined due to security reasons. Interviewee 3 described how tasks in Jira were created: *"Usually, we sit down, discuss, and after we have already decided, we create a task in Jira. Therefore, in my case, my tasks were always clear to me."*. However, interviewees acknowledged that most of the time uncertainties arise when someone creates a task for someone else without preliminary discussions and therefore, some of the tasks contain comments towards clarifying those uncertainties.

4.3 Case study 3 (CS3): An open-source database management system

In the third case study, we investigated the development of an open-source distributed NoSQL database management system that has been developed for more than ten years. The development process in the project includes a decentralized community of developers from around the globe. The team which consists of twenty four contributors, out of which, around ten to fifteen contributors are active. The project contributors actively use Jira as the TMS tool, which contained 14,193 tasks at the time of the interview. The project is not strictly hierarchical; there is no single responsible person and everything is based on merit. However, there is a Project Management Chair (PMC) which has regular members and an implicit project lead who may give a general direction to the project.

With eleven years of professional experience, **Interviewee 5** has been actively contributing to the project in the role of a software developer for the past two years.

Since the team of contributors is widely distributed, most of the discussions that lead to ADDs happen asynchronously via Jira. Though, when developers want to short-circuit something and discuss smaller questions (as Interviewee 5 put it, *"do not deserve to be historically noted in Jira"*), IRC channel may be used with someone who is currently online from the same time-zone. In case a decision is made in IRC, it is then documented in the TMS.

4.4 Uncertainty expressions in ADM discussions

In Table 1, we present the number of the investigated tasks per each case study along with the number of comments, sentences, and uncertainties in each ADM discussion around those tasks. Each interview took around eighty minutes. On average around hundred and thirteen sentences were annotated and discussed by each interviewee. As Jira was the main communication tool in CS3, it contained the largest number of sentences, messages, and uncertainties in comparison to other cases. Therefore, during the interview, it was possible for Interviewee 5 to annotate only one task and its discussions. In total, six tasks were annotated by interviewees in CS1 and five tasks in CS2. Unfortunately, only one interviewee from CS3 agreed to participate in this case study.

As the goal of our study was to explore the phenomenon of uncertainty expressions in discussions from the perspective of practitioners, during the interview, we provided participants with preselected tasks that contained at least two uncertainty cues identified by us in the data collection phase. Thus, the numbers presented in Table 1 should not be treated as a normal distribution of uncertainties over the whole number of tasks in the corresponding projects. Nevertheless, data in Table 1 should give an overview of the task discussions that were presented to the interviewees in each of the case studies.

Based on the responses of the interviewees, one of the main observations made in our studies is that "quite often, uncertainty expressions in discussions are not used to express uncertainty itself". Below we list some observed examples, where interviewees shared with us their intentions behind using uncertainty expressions in their discussions.

(1) **Feedback trigger** - politely sharing a possible solution with the group to get feedback. In CS1, one of the task's comments contained *"To solve this issue, we **think** that the more efficient way is to..."*. Interviewee 2 did not consider this comment to be uncertain. He argued that *"they discussed it already in a group, came up with this solution, and wanted to share it with the group. So this is more like a feedback trigger."*
(2) **Preference** - polite expression of a preference among multiple alternatives. For instance, in CS3, a colleague of Interviewee 5 wrote: *"**In my opinion**, the useFiltering name is a bit*

Table 1: Numbers of comments and uncertainties analyzed per CS

	CS1							CS2						CS3
	Tasks							Tasks						Tasks
	T1	T2	T3	T4	T5	T6	*Total*	T1	T2	T3	T4	T5	*Total*	*T1 / Total*
Task Description														
Number of sentences	2	5	6	5	9	3	*30*	6	2	2	1	1	*12*	*2*
Number of uncertainty cues	0	0	0	2	0	0	*2*	1	0	0	0	0	*1*	*0*
Comments														
Number of comments	5	4	7	8	8	2	*34*	9	2	5	6	2	*24*	*36*
Number of sentences	15	8	19	13	24	8	*87*	33	27	18	13	14	*105*	*146*
Number of uncertainty cues	4	4	2	3	2	3	*18*	6	3	4	3	3	*19*	*38*
Number of uncertainty cues per comment	0,80	1,00	0,29	0,38	0,25	1,50	*0,53*	0,67	1,50	0,80	0,50	1,50	*0,79*	*1,06*

confusing, we should use allowFiltering instead." As Interviewee 5 explained: *"There were two solutions, both would work, and he wouldn't say that it is completely wrong to do it the other way, but he would prefer this way. So this is a preference rather than uncertainty."*

(3) **Reassurance** - use of uncertainty expressions for the sake of reassurance, in case, the expressed certain assumption turns out to be false. For example, one of the observed statements in CS1 was: *"It **seems** to be that the error is **maybe** related with not refreshed token."* When Interviewee 2 was asked to reflect on this statement made by his teammate, he claimed that it was not uncertainty and provided the following explanation: *"You write it like this, so that, if in the end, if it is not really an error, you don't look like an idiot and you have a back door."*

(4) **Figure of speech** - personal preference to use a figure of speech to a make statements, generally, more polite. For example, in CS3, on Interviewee 5's own statement in the task's discussion: *"I've implemented a fast prototype locally and it **seems** to fit both the cases."*, he commented that *"It [usage of 'seems'] was a figure of speech, I didn't want to look like I was overly confident."*

We also observed cases where interviewees admitted to be uncertain or stated that the author of the comment expressed uncertainty. For example, in Task 1 (T1) of CS1, Interviewee 1 demonstrated uncertainty in regards to his colleague's decision to implement changes to the functionality of the system: *"... **I'm still not sure if** it should be part of the backend [implementation], ..."*, which was immediately followed by the assumption on how it could be implemented: *"..., but it would work this way if the request of a patient would be discarded on the backend while requesting professional information."* . As Interviewee 1 explained: *"He [his colleague] was really new to the project and he didn't understand this big picture [showing layered architecture of the system]."*. Therefore, Interviewee 1 wanted to express his uncertainty towards previously made ADD.

In CS3, Interviewee 5 chose to perform the task (T1) to improve the current functionality of the system. According to Interviewee 5, T1 was one of his first tasks since his involvement in the project. He was experiencing lots of uncertainties in the decisions that he was taking during implementation of that task. Interviewee 5 followed an iterative development process. He created small patches to the system and submitted them for code review. At the same time, he commented after each submission to the T1, explained his decisions, as well as, expressed his uncertainty towards the behavior of the system that could be affected by his changes. As Interviewee 5 explained: *"In complex projects, you are never 100% certain, you kind of understand what is going on, but it can always happen that there is a certain facet/aspect, that you are missing. The more complex the system is, the harder it is to be 100% sure about something."*. Here are some examples of uncertain statements towards the implementation details expressed by Interviewee 5 in T1:

- *"**I'm still not sure if** the handling of multi-columns is correct, since their SliceRestriction::addRowFilterTo is not permitted in the moment."*
- *"..., I am **most likely** missing something, but at least so far the collection types aren't allowed to be primary key at all ..."*

Moreover, as indicated by Interviewee 5, the reviewer of T1 was also expressing uncertainty towards the implementation details:

- *"I **guess** that problem will also happen for CONTAINS and CONTAINS KEY restrictions as they cannot be used to build a Clustering, but can be used to filter..."*
- *"Instead of modifying getRowFilter, I **think** that you should add ClusteringColumsRestrictions to inrexRestrictions if filtering is allowed and some clustering columns restrictions require filtering."*

Interviewee 5 commented on the reviewer's last statement: *"Probably he checked it but he was not 100% sure. So he was giving me the general direction, but he didn't really do 100% of the work to make sure it's correct"*.

In CS3, Interviewee 5 also identified uncertainty towards the estimates of time and complexity of implementation of certain decisions. For example, *"**I'm not sure** how fast I can pull it out"*, *"I am making changes that you have listed, hope to have them ready shortly"*. In some cases, participants of the discussion expressed uncertainty on a meta-level; towards their understanding of the

previously provided information or assumption, such as, *"**I think**, you are just asking me about views here, so I'll answer that"*.

In CS2, Interviewee 4 identified suggestions made by his colleague regarding architectural changes to be uncertain. For instance, *"In public subnet, **I suggest** to have reverse proxies only,..."* .

In CS1, we observed authority biases. Interviewee 2 expressed his assumption saying: *"Since Interviewee 1 is writing this, I think he is already pretty sure that this is a backend issue."* However, during the discussion with Interviewee 1 about the same uncertainty expression, he stated that he had actually been uncertain.

In general, irrespective of who authored the uncertainty expressions in discussions (be it the interviewee himself or one of his/her teammates), disambiguation of uncertainty expressions in discussions required certain cognitive effort from the interviewees and even then, in some cases, interviewees were uncertain if they perceived and identified uncertainty expressions correctly. Moreover, all the interviewees mentioned that in their daily lives they never paid so much attention to uncertainty cues but rather acted on them unconsciously.

5 DISCUSSION

Treating the SA artifact as a set of ADDs is a well-established practice among researchers. However, to the best of our knowledge, researches do not have any well-established measurements for objectively differentiating "good" and "bad" ADDs. Especially when ADDs are made in different project contexts and organizational environments. One way to evaluate ADDs is to measure their sustainability over time; similar to the measurements performed over source-code artifacts (modularization, architecture decay, code smells) [11, 13], systems' functionality [12], and life-span of technologies used to develop software systems [17]. Focusing on uncertainty expressions, their disambiguation, and their evolution throughout team communications contributes to a better understanding of how sustainable ADDs could be made in software projects, particularly when stakeholders rely on unstructured discussions. In parallel, studies related to decision-making contribute to better understanding of organizational and stakeholders' behavioral aspects influencing the sustainability of SAs [19, 33].

The number of comments and the number of sentences in each task shown in Table 1 reflect on the way stakeholders communicate with each other in a project. For instance, the largest number of communication messages (comments and sentences) were observed in CS3, where stakeholders communicated mostly using TMS, contrary to the first two case studies, where TMSs was rather used in an 'ad-hoc' manner. In each of the studied software engineering projects, we found discussions around ADDs that contained uncertainty expressions.

Based on the interviewees' responses, we observed that uncertainty constructs are used by architects in group discussions to express not only subjective estimates with regards to ADDs but also to trigger feedback regarding the proposed ADDs and to express their preferences in a polite way.

In this study, we only focused on one type of communication artifact – discussions in TMS. Discussions in TMS are mostly explicit and provide evidence of teams' negotiations to address design concerns. However, tracing uncertainty expressions during the entire life-span of ADDs must also involve the observation of synchronous interactions, which can be done by recording audio during team meetings and transcribing them thereof. Those transcriptions can then be used to create conversation coherence graphs for further analysis (similar to the study by Jordan et al. [10]). However, to create such graphs for analysis, [human] annotators would have to be aware of the different types of uncertainty expressions in ADM as presented in this study.

As Interviewee 4 put it: *"In most cases, it's always about missing knowledge. During development, you always have to get rid of uncertainty by collecting knowledge"*. One could suggest that early detection and elimination of the uncertainties in discussions could help to make discussions more productive. However, guided by the observations of Jordan et al. [10] and based on our observations of discussions in TMS, we are rather inclined to suggest that uncertainty expressions usually provoke the involvement of stakeholders in a project. The existence of uncertainty expressions in discussions could lead to stakeholders asking new questions for clarification and unearthing new design alternatives during the process.

Architects who might act as facilitators of group discussions might also consider cultivating [certain] uncertainty expressions in those discussions. Thus, participants of group discussions would stay motivated to resolve the expressed uncertainties, acquire new architectural knowledge, and in the end, make informed sustainable ADDs. However, to (in)validate such a hypothesis, further research on the analysis of stakeholders' communication in group decision making process has to be performed. Such studies should take into consideration the types of uncertainties presented in this paper as well as the challenges of disambiguating uncertainty expressions.

Interviewees in our study admitted that as the rule of thumb they use uncertainty expressions so that they do not look overly confident in front of their colleagues. Using uncertainty expressions in discussions rather seems to be an unconscious activity. Further studies need to be performed to understand how architects communicate doubts during decision making and if and how such behavior could influence architects' preference not only while choosing one of the alternative decision choices but also while involving those individuals who communicate doubts in a subsequent GDM process.

6 CONCLUSIONS

In this paper, we examined the discussions in task management systems of three software engineering projects and conducted interviews with two software architects and three software engineers to investigate uncertainties expressed in those discussions and how those uncertainty expressions are comprehended by their respective authors and readers. We systematically analyzed the interviews and documented different types of uncertainties in architectural decision making. We observed and presented how practitioners react to their own uncertainties and uncertainties of their colleagues. Furthermore, we also discussed the challenges of disambiguating uncertainty expressions in discussions.

The results of this work can be used for further investigation of GDM in SA, in particular, how uncertainty expressions influence the process of ADD making in software engineering teams. For example, the findings relate uncertainty to communication techniques

that aim to elicit important information by querying and triggering for more information. Moreover, for instance, communication techniques of architects may include expressions of uncertainty as invitations to other stakeholders to come forward, explain and complete missing knowledge.

REFERENCES

[1] R. Alkadhi, M. Nonnenmacher, E. Guzman, and B. Bruegge. 2018. How do developers discuss rationale?. In *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 357–369. https://doi.org/10.1109/SANER.2018.8330223

[2] Strauss Anselm and Juliet Corbin. 1998. Basics of qualitative research: Techniques and procedures for developing grounded theory. *Thousand Oaks, California: Saga Publication* (1998).

[3] Manoj Bhat, Klym Shumaiev, Andreas Biesdorf, and Uwe Hohenstein. [n. d.]. Automatic Extraction of Design Decisions from Issue Management Systems : A Machine Learning Based Approach. ([n. d.]). https://doi.org/10.1007/978-3-319-65831-5_10

[4] Felix C Brodbeck, Rudolf Kerschreiter, Andreas Mojzisch, and Stefan Schulz-Hardt. 2007. Group decision making under conditions of distributed knowledge: The information asymmetries model. *Academy of Management Review* 32, 2 (2007), 459–479.

[5] Kathleen M. Eisenhardt. 1989. Building Theories from Case Study Research. *The Academy of Management Review* 14, 4 (1989), 532–550. http://www.jstor.org/stable/258557

[6] Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 Shared Task: Learning to Detect Hedges and Their Scope in Natural Language Text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task (CoNLL '10: Shared Task)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 1–12. http://dl.acm.org/citation.cfm?id=1870535.1870536

[7] C. Ghezzi, L. S. Pinto, P. Spoletini, and G. Tamburrelli. 2013. Managing non-functional uncertainty via model-driven adaptivity. In *2013 35th International Conference on Software Engineering (ICSE)*. 33–42. https://doi.org/10.1109/ICSE.2013.6606549

[8] Ken Hyland. 1994. Hedging in academic writing and EAF textbooks. *English for specific purposes* 13, 3 (1994), 239–256.

[9] Anton Jansen and Jan Bosch. 2005. Software architecture as a set of architectural design decisions. In *5th Working IEEE/IFIP Conf. on Software Architecture (WICSA'05)*. IEEE, 109–120.

[10] Michelle E Jordan, Diane L Schallert, Yangjoo Park, SoonAh Lee, Yuehhui Vanessa Chiang, An-Chih Janne Cheng, Kwangok Song, Hsiang-Ning Rebecca Chu, Taehee Kim, and Haekyung Lee. 2012. Expressing uncertainty in computer-mediated discourse: Language as a marker of intellectual work. *Discourse Processes* 49, 8 (2012), 660–692.

[11] Heiko Koziolek. 2011. Sustainability Evaluation of Software Architectures: A Systematic Review. In *Proceedings of the Joint ACM SIGSOFT Conference – QoSA and ACM SIGSOFT Symposium – ISARCS on Quality of Software Architectures – QoSA and Architecting Critical Systems – ISARCS (QoSA-ISARCS '11)*. ACM, New York, NY, USA, 3–12. https://doi.org/10.1145/2000259.2000263

[12] H. Koziolek, D. Domis, T. Goldschmidt, and P. Vorst. 2013. Measuring Architecture Sustainability. *IEEE Software* 30, 6 (Nov 2013), 54–62. https://doi.org/10.1109/MS.2013.101

[13] Duc Le, Daniel Link, Arman Shahbazian, and Nenad Medvidovic. 2018. An empirical study of architectural decay in open-source software. In *In IEEE International Conference on Software Architecture (ICSA) 2018*. IEEE.

[14] Emmanuel Letier, David Stefan, and Earl T. Barr. 2014. Uncertainty, Risk, and Information Value in Software Requirements and Architecture. In *Proceedings of the 36th International Conference on Software Engineering (ICSE 2014)*. ACM, New York, NY, USA, 883–894. https://doi.org/10.1145/2568225.2568239

[15] Xiujun Li, Wei Gao, and Jude W Shavlik. 2014. Detecting semantic uncertainty by learning hedge cues in sentences using an HMM. In *Workshop on Semantic Matching in Information Retrieval (SMIR*. World Scientific, 11.

[16] Marc Light, Xin Ying Qiu, and Padmini Srinivasan. 2004. The language of bioscience: Facts, speculations, and statements in between. In *HLT-NAACL 2004 Workshop: Linking Biological Literature, Ontologies and Databases*.

[17] Robert C Martin. 2002. *Agile software development: principles, patterns, and practices*. Prentice Hall.

[18] Ben Medlock and Ted Briscoe. 2007. Weakly supervised learning for hedge classification in scientific literature. In *ACL*, Vol. 2007. Citeseer, 992–999.

[19] Nils Brede Moe, Aybüke Aurum, and Tore Dybå. 2012. Challenges of shared decision-making: A multiple case study of agile software development. *Information and Software Technology* 54, 8 (2012), 853–865.

[20] V. S. Rekhav and H. Muccini. 2014. A Study on Group Decision-Making in Software Architecture. In *2014 IEEE/IFIP Conference on Software Architecture*. 185–194. https://doi.org/10.1109/WICSA.2014.15

[21] Arman Shahbazian, Youn Kyu Lee, Duc Le, and Nenad Medvidovic. 2017. Uncovering Architectural Design Decisions. *arXiv preprint arXiv:1704.04798* (2017).

[22] Christoph Johann Stettina and Werner Heijstek. 2011. Necessary and Neglected?: An Empirical Study of Internal Documentation in Agile Software Development Teams. In *Proceedings of the 29th ACM International Conference on Design of Communication (SIGDOC '11)*. ACM, New York, NY, USA, 159–166. https://doi.org/10.1145/2038476.2038509

[23] György Szarvas, Veronika Vincze, Richárd Farkas, György Móra, and Iryna Gurevych. 2012. Cross-genre and Cross-domain Detection of Semantic Uncertainty. *Comput. Linguist.* 38, 2 (June 2012), 335–367.

[24] D. A. Tamburri, R. Kazman, and H. Fahimi. 2016. The Architect's Role in Community Shepherding. *IEEE Software* 33, 6 (Nov 2016), 70–79. https://doi.org/10.1109/MS.2016.144

[25] Dan Tofan, Matthias Galster, and Paris Avgeriou. 2013. Difficulty of Architectural Decisions – A Survey with Professional Architects. In *Software Architecture*, Khalil Drira (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 192–199.

[26] Smrithi Rekha V and Henry Muccini. 2018. Group decision-making in software architecture: A study on industrial practices. *Information and Software Technology* 101 (2018), 51 – 63. https://doi.org/10.1016/j.infsof.2018.04.009

[27] Hans van Vliet and Antony Tang. 2016. Decision making in software architecture. *Journal of Systems and Software* 117 (2016), 638–644.

[28] Andrew H Van De Ven and Andre L Delbecq. 1974. The effectiveness of nominal, Delphi, and interacting group decision making processes. *Academy of management Journal* 17, 4 (1974), 605–621.

[29] Chen Yang, Peng Liang, Paris Avgeriou, Ulf Eliasson, Rogardt Heldal, and Patrizio Pelliccione. 2017. *Architectural Assumptions and Their Management in Industry – An Exploratory Study*. Springer International Publishing, Cham, 191–207. https://doi.org/10.1007/978-3-319-65831-5_14

[30] Hui Yang, Anne De Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh. 2012. Speculative requirements: Automatic detection of uncertainty in natural language requirements. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*. IEEE, 11–20.

[31] Robert K Yin. 1994. Case study research: Design and Methods, Applied social research methods series, 5. *Biography, Sage Publications, London* (1994).

[32] Lisl Zach. 2006. Using a multiple-case studies design to investigate the information-seeking behavior of arts administrators. *Library trends* 55, 1 (2006), 4–21.

[33] Carmen Zannier and Frank Maurer. 2007. Social factors relevant to capturing design decisions. In *Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent*. IEEE Computer Society, 1.