Master of Science in Software Engineering
October 2017

Product Manager view on Practical Assumption Management Lifecycle about System Use

Guangyu Zhang

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona Sweden

Contact Information:
Author(s): Guangyu Zhang
E-mail: guzh15@student.bth.se

University advisor: Dr. Samuel A. Fricker
Department Software Engineering

ABSTRACT

Context. In practice, software projects frequently fail in many fields, which causes the huge loss for the human being. Assumption faults are recognized as a main reason for the software project failures. As the world is changing fast, environment assumptions of software can be easily wrong. The daily assumption-related activities show not enough effectiveness and efficiency to deal with assumption faults. For example, no documenting of key assumptions, inappropriate assumption validation, lack of knowledge. In research, there is no empirical research about assumption management practice. Two assumption management frameworks were outlined. They both support the assumption formulation and assumption management. The formal assumption management framework provides an assumption-component mapping function to analyze assumption failures.

Objectives. Our goal is figuring out how development team members handle environment assumptions today in practice and how they might handle them better tomorrow. To be specific, I test the applicability of the so far theoretical assumption management frameworks and investigate the assumption type, assumption formulation and assumption management in practical software development

Methods. An interview-based survey was implemented with 6 product managers from Chinese software companies. They have rich experiences on assumption management and software development. I used directed content analysis to analyze the qualitative data. The result of the research is intended to be a static validation of the assumption management frameworks.

Results. Interviewees consider that the assumption-component mapping function of the formal assumption management framework is useful in making decisions and analyzing the problems. However, using these frameworks takes too much effort. The functions of frameworks are covered by the development team members and the existing tools. Assumptions tend to be discovered when they frequently change and are important to the requirements. The main assumption types are user habit assumptions and quality attribute assumptions, which are both requirement assumptions. The user habit assumptions consist of name, description and value, while the quality attribute assumption formulation is name and value. The major assumption treatment activities are figuring out the value of assumptions, assumption monitoring, assumption validation and handling assumption failures. Assumption failures result in the loss of users and benefits. Assumption failures are always caused by the poor ability and experience of development team members.

Conclusion. I create an assumption management model based on my result, and find out the advantages and disadvantages of the formal assumption management framework and semi-formal assumption management framework. The research could help improve the efficiency and effectiveness of assumption management practice. Also. The research can be treated as the starting point to study assumption management practice deeper.

Keywords: assumption management, practice, improvement opportunity, development team members

CONTENTS

1 INTRODUCTION

An assumption is “a fact or statement taken for granted.” The assumption is a broad concept in the world, and I apply the concept in the software engineering domain. Software assumptions are special that need a different system to manage because of the following reasons.

Software assumptions concern almost all aspects of software engineering, from the stakeholders to software development lifecycles. Besides, they have intensive relation with the software quality [13].

Firstly, software assumptions are used in a way that software-specific problems are solved. The assumption is a bridge between software and the world. The software and world interact with each other through assumptions. On the one hand, assumptions are used to reflect the conditions on which the software could work without sufficient and available knowledge, information, experience, historical data and requirements [9]. Besides, assumptions make software development more efficient [9]. On the other hand, the execution of software could bring changes on its environment through assumptions [35].

Secondly, the software assumption has a special lifecycle, containing three stages: assumptions being discovered, assumption changing and assumption failing, which can be regarded as the base to establish assumption management systems [30].

Thirdly, software assumptions are always managed according to different types. For example, the today’s development teams figure out user habit assumptions through investigation or discussion with users, while quality attribute assumptions are made by using historical data. Besides, previous assumption management frameworks record the assumption type when documenting assumptions [9] [8] [34].

Fourthly, the complexity of software assumptions is high. An assumption could have connections with different-level artifacts in software development. It is difficult for development team members to manage assumptions that are involved in requirements, architectures, codes, design decisions and so on. Therefore, it is better to use a special system to support assumption management [13].

Fifthly, software assumptions run through and are scattered in the whole software lifecycle. In every stage of the software lifecycle, development team members could discover new assumptions [13]. Besides, after figuring out the value of the assumption, it could change until the end of assumption lifecycle in the software lifecycle because of the software uncertainty [9]. Today’s software executes in the dynamic environment, so, the volatility of software assumptions is high, resulting in the software project failures [13].

Sixthly, software assumptions become invalid in two ways: development team members do not notice the assumption or do not document it [13].

Seventhly, it is software-related stakeholders that handle assumptions, which mainly contain product manager, requirement engineer, architect, developer, reviewer and maintainer [9].

A web survey shows that 26%-34% percent software projects failed and resulted in the great loss [1].

One of the most critical reasons for software failures is the wrong assumption, especially environment assumption [13]. The rate of software failures is high in many fields, which results in a significant loss of resources and money [1].

As the software systems and their environment are changing fast, the poor assumption management activities are becoming serious [9]. That always leads to assumption failures.

So, it is necessary to improve the efficiency and effectiveness of assumption management practice. It is development teams that handle assumptions every day [9]. The best way to understand the assumption management practice and improvement opportunity is directly investigating development team members. Besides, there are few empirical studies of assumption management. I also found the advantages and disadvantages of the assumption management frameworks that are highly likely to support assumption management practice.

I used the interview research to achieve the research goal. I used directed content analysis to analyze the data. The interview was applied with six Chinese product managers. All interviewees have three years or more experience and bachelor or above educational background. Besides, this is a static validation [37] for the assumption management frameworks.

I expected to make an assumption management model, and understand the advantages and disadvantages of the assumption manage frameworks.

Firstly, I introduce the related work (Section 2). Secondly, I describe the methodology I used to conduct the research (Section 3). Thirdly, I present the result and analysis of the study (Section 4). Fourthly, I discuss the result (Section 5). Finally, I conclude the research (Section 6).

2 RELATED WORK

Here I present several previous works that concern assumptions.

2.1 Terminology

An assumption is "a fact or statement taken for granted" [30]. The concept highlights that the assumption is used when there is not enough related information and is required for software development.

Environment assumptions are the outside assumptions of the software that executes in the environment. Like user habit assumptions, assumptions about software execution contexts, quality attribute assumptions and so on.

Assumption impact indicates the role assumptions play in software engineering. Practitioners use assumptions to pursue the success of software development, especially in the dynamic software environment.

Assumption lifecycle is a set of stages in its lifetime. The lifecycle I use in the research contains three phases: assumption being discovered, assumption changing and assumption failure.

Assumption management is a set of activities implemented to control the stages of the assumption life cycle. The previous work [30] proposed four activities of assumption management: identifying assumptions, recording assumptions, monitoring assumptions and dealing with assumption failures.

Assumption type is subdivisions of assumptions to ease managing assumptions. Assumption types are not same in different assumption management frameworks.

Assumption formulation is "a combination of values of assumption properties that is frequently observed during assumption" [30].

I use Assumption value to identify the value for all types of assumptions, whether user habit assumptions or quality attribute assumptions. For example, the interface design is a kind of a user habit assumption value; the numerical value is the quality attribute assumption value.

Assumption state indicates whether the the assumption value established to software fit into the practical realities.

Assumption failure (or invalid assumption) is that the the assumption value established to software does not fit into the practical realities.

2.2 Assumptions, Constraints, Requirements, Design Rationale

It is hard to distinguish among assumptions, constraints, requirements, and design rationale, because they have overlap. The four terminologies are sometime used with no differences [48] and extended to beyond the original definition [49].

Requirements are how users expect about the software. Constraints indicate restrictions, limitations and regulations of the software. Generally, requirements and constraints are the base of software development that agreed by companies and customers [13].

Assumptions reflect the conditions of software. From the perspective. Constraints and Assumptions are overlap. The difference is that an assumption could be invalid at once they are created, while constraints can not [13].

Design rationales are the reasons for design decisions [13]. Assumptions and design rationales can be regarded as a part of design decisions [50] [52]. Assumptions and constraints can be used to extract design rationales [51].

2.3 Assumption in software and system development

2.3.1 Assumption in waterfall and agile

Assumption management is used to handle frequent assumption changes in the software environment, which is also regarded as a kind of risk management. I. Ostacchini and M. Wermelinger [30] approve that agile development is used when requirements are highly possible to change, while the requirements would not change once identified in waterfall development. So, assumption management fits more into agile than waterfall. Besides, assumption management is a lightweight method in the implementation and documentation, which is suitable for agile. Compared with agile, the waterfall is rigorous and process-oriented.

2.3.2 Main stakeholders

The section describes the assumption-related stakeholders.

2.3.2.1 **Product manager**

S. A. Fricker [44] find that today's product managers can manage or lead some products during their lifecycle to make the product succeeded and max the value of businesses. Lewis et al. [9] evaluate the influence of executing the software in different environment based on assumptions.

2.3.2.2 Requirement engineer

K. Pohl and C. Rupp [45] propose a definition: "a requirements engineer is expected to elicit needs and expectations from stakeholders, to model and analyze the impact of these inputs on the system together with the development team, and to check proposed implementations for acceptance by the stakeholders". Lewis et al. [9] advise using assumptions to find the vagueness in requirements and verify the validity of the requirements.

2.3.2.3 Architect

C. Yang et al. [22] describe a situation that architects identify assumptions from project documentation or architecture documentation through discussions. The assumptions are identified based on their experience and knowledge. M. R. McBride [38] a definition of architects: "An architect is responsible for the design and technological decisions in the software development process" [38]. Lewis et al. [9] suggest that architects evaluate assumptions to guarantee that they are consistent with the application.

2.3.2.4 Developer

Lewis et al. [9] think that the developer, also known as the programmer, write code to realize requirements. They communicate assumptions with the users to make sure whether assumptions are satisfied. Besides, the assumptions can be used to optimize the code.

2.3.2.5 Reviewer

Lewis et al. [9] describe that the reviewer checks the software project deliverables. They can be any stakeholder that is related to the software project. They always use assumptions to create test cases.

2.3.2.6 Maintainer

Lewis et al. [9] state that they make the decisions about the software update based on the assumptions. For example, a maintainer evaluates the degree to which an update is based on the existing technology and determines the new environment that the software fit into.

2.3.2.7 Vendors

For vendors, they provide the outside technology to meet the value of the assumption and notify the team if the technology is updated.

2.3.3 Assumption and requirement

A. van Lamsweerde [39] approves that assumptions underline requirements to form a complete software. So, assumption management is always treated to have intensive relation with requirement engineering. S. Fickas and M. S. Feather [40] think that the requirements need to be updated when the corresponding assumptions become invalid.

A relation between requirements and assumptions of the context of risk estimation is proposed in A. Miranskyy et al. [41]. The relation is expressed through a Boolean network.

Haley et al. [11] describe how requirement developers view on trust assumptions in the security context. They make assumptions of security requirements. The engineers think that the derivation of security requirements and functionality realization affected by security requirements.

2.3.4 Assumption and architecture

S. Uchitel and D. Yankelevich [42] treat Assumptions as the connections between the different components (the outside technology is a form of components) at the architecture level. Here the assumption is also a form of dependencies between components.

P. Kruchten et al. [43] approve that software architecture is constructed based on design decisions, which are also based on rationales. Assumptions are regarded in design decision and rationale management. Besides, it is difficult to distinguish between the assumption and design rationale.

Garlan et al. [23] reviewed the challenges of architectural mismatches. They also proposed three technologies that deal with architectural mismatches: mismatch repairing, mismatch detection and mismatch prevention.

2.3.5 Assumption and code

M. M. Lehman and J. F. Ramil [27] discuss code assumptions in software evolution. When software is updated to meet the new need, the assumptions of its code must be adjusted to fit into the new context. Here, M. A. A. Mamun and J. Hansson [13] highlight that hardware is the important context that needs to be defined in assumptions.

2.4 Assumption impact

R. Ali et al. [32] regard the assumption as a factor that causes the software evolution because the software requirements and context interact with each other via assumptions. A software system could reach a goal is based on assumptions. They inevitably become invalid, which means that the original aim of the software is inappropriate. As a result, the software must evolve.

X. Wang et al. [35] propose roles of assumptions: software can not only handle information in given constraints, but also facilitate changes in the world where the software executes. To execute the software, the necessary actions should be taken, which happen in the social environment. Besides, how world changes could influence the state of the software and software use.

Lewis et al. [9] find that using assumptions is helpful to develop software in the whole software lifecycle.

Burge et al. [36] discover that the reuse is becoming more prevalent in software development. The biggest problem is whether a well-designed component can fit into the new condition, which can be solved by making explicit and visible assumptions.

As described in Introduction Section, assumption failures could result in a huge loss of the resource and money.

2.5 Assumption management tools

In this section, I introduce four frameworks that help to manage assumptions.

2.5.1 Assumption management frameworks

I use the name “semi-formal assumption management framework” and “formal assumption management framework” to identify the frameworks proposed by Lewis et al. [9] and A. S. Tirumala [8]. The authors of the frameworks do not use the names, being quoted from M. A. Al-Mamum and J. Hansson [13].

2.5.1.1 semi-formal assumption management framework

Lewis et al. [9] developed a semi-formal assumption management system prototype for supporting assumption recording, assumption searching and assumption validating. Assumptions are described in XML. Assumptions are extracted from Java code and stored in a repository using an assumption extractor. The system is web-based, which offers to browse and search assumptions with given criteria. A validator can use the webpage to review and validate assumptions in the repository. The assumption extractor is a Java application with Java runtime environment that executes on users' computer. The Management system is realized using Java Server Pages (JSPs) that execute on a Tomcat servlet engine and JavaBeans. An Oracle 9i database is used to implement assumption repository.

2.5.1.2 formal assumption management framework

A. S. Tirumala [8] developed a formal assumption management system. They create a language to document assumptions with a machine-checkable format where assumptions and their guarantees are encoded in the components of the architecture. The system supports not only the static assumption validation but also the dynamic assumption validation. The biggest advantage of the framework is that it could validate invalid assumptions for complex, large-scale software with only the technical assumption scope. The framework is realized for assumptions written in Architecture Analysis and Design Language (AADL), which is a kind of Architecture Description Language (ADL).

2.5.2 Assumption monitoring model

R. Ali et al. [32] outline a model that could monitor assumptions in a requirement model and assess their validity. The requirement model is based on Contextual Goal Model [33]. The monitoring model traces the variables in the requirement model during software execution.

2.5.3 Architectural Assumptions management model

C. Yang and P. Liang [34] provide a model to identify and record architectural assumptions. The model contains two parts: architectural assumption library and architectural assumption card. The frame could improve the efficiency of identifying and recording architectural assumptions, facilitating architecture evolution, maintenance, and design in agile software development.

2.5.4 Summary

The four frameworks have their assumption formulation, and support identifying and recording assumptions. All of them can not support figuring out the values of assumptions and handling assumption failures. The first two frameworks are used to treat code-level assumptions. The third one is designed for requirement-level assumptions. The last framework can only deal with architecture-level assumptions.

2.6 Assumption type

Here I introduce the type of assumptions used in assumption management frameworks. The efficiency of assumption management practice suffers from the lack of the scope of assumption type [13].

2.6.1 Assumption type in the semi-formal framework

When development team members develop requirements, they make different assumptions, concerning requirement interpretation, resources availability and data type. The system defines five types of assumptions: control assumption, environment assumption, data assumption, usage assumption, and convention assumption [9].

Figure 2-1 assumption type in semi-formal framework

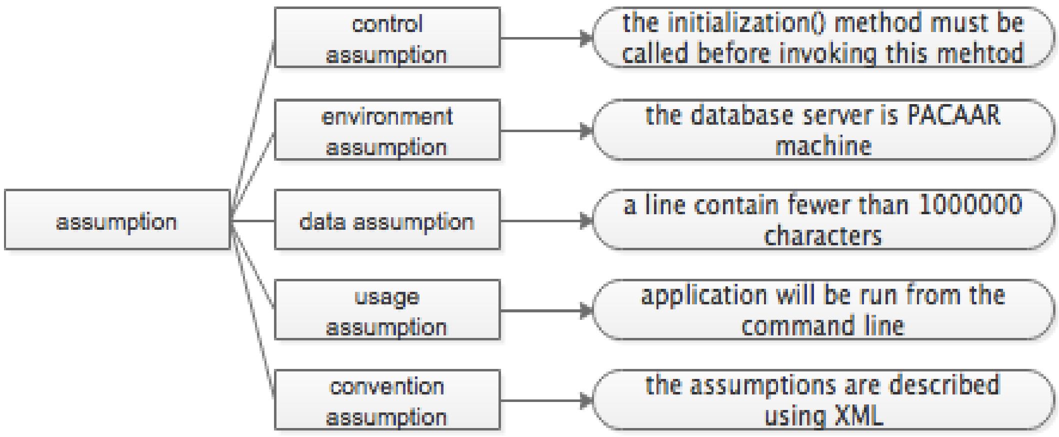
assumption
control assumption
environment assumption
data assumption
usage assumption
convention assumption
the initialization() method must be called before invoking this mehtod
the database server is PACAAR machine
a line contain fewer than 1000000 characters
application will be run from the command line
the assumptions are described using XML

Figure2-1 presents the types of assumption with their representative example. The following is the detailed description.

Control assumptions are about expected control flows. Development team members always assume the sequence in which methods are invoked. Control assumptions are used to 1) make sure the application flow and 2) be as conditions, based on which the development team members can change the software. Environment assumptions represent what is predicted of the environment that the software will be executed in. For example, the external database. Development team members often communicate environment assumptions with end users. Change analysts assess the degree to which the implementation depends on outside components to verify the modification cost using it. System integrators could fix bugs more easily through environment assumptions. Maintainers use them to look for the environment. Product managers evaluate the performance of software running in the different environment. Data assumptions concern the format of the input and output data. This type of assumption can be regarded as a condition to guarantee the application runs as expected. Also, it is easy to find how data is used through the assumption. Data assumptions are often used to locate vagueness in requirements, build test cases, optimize the code and reduce the complexity. Usage assumptions are about how the software is expected to be used. For example, the software will be used from graphic interface. Convention assumptions are standards or conventions that development team members follow [9].

2.6.2 Assumption type in the formal framework

Figure 2-2 assumption dimension

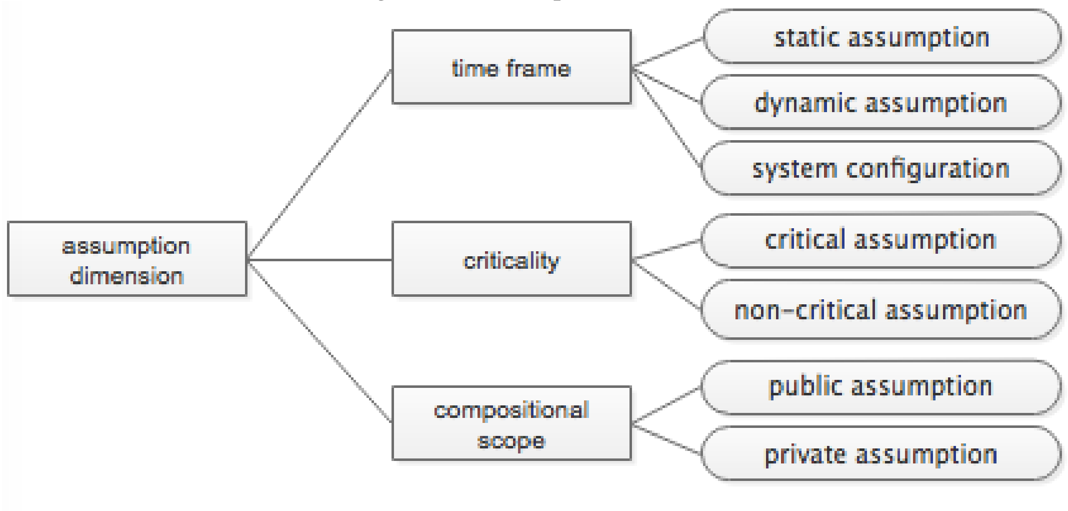

Figure 2-2 states three dimensions of the classification of assumptions established in the Framework: time-frame, criticality and compositional scope.

The first classification dimension of assumptions is its time-frame of validity. Static assumptions are assumptions that present the same validity in the whole software lifecycle. Dynamic assumptions represent the kind of assumptions with validity that can change during the software lifecycle. System configurations are those assumptions can not be changed during the software execution. The second dimension of classification is the criticality of assumptions. Critical assumptions are those assumptions, whose failure will result in major function failures or even the whole system down. Non-critical assumptions are those whose failure just degrade the software performance or damage non-core function. The third dimension of classification concerns abstracting relevant assumptions when the system becomes larger. When the system becomes larger, the original system becomes a component of the new system. Public assumptions are those assumptions exposed as a part of the super-component or needed to be satisfied by guarantees of outside components. Private assumptions are not exposed to the larger component and satisfied by the inside sub-components. Guarantees provide values for evaluating a specific assumption. They are classified according to their source that the values is generated from. Human-entered guarantees are entered by the human. Machine-generated guarantees are created by running a routine. Hybrid guarantees have both human-entered values and machine-generated values [8].

2.6.3 summary

The two ways of assumption categorization are different. The first one is based on the content of assumptions, while the second one is according to the criticality and validity of assumptions.

2.7 Assumption formulation

I. Ostacchini and M. Wermelinger [30] regard assumption formulation as "a combination of values of assumption properties that is frequently observed during assumption" [30].

Here I introduce assumption formulations in three frameworks.

2.7.1 Assumption formulation in the semi-formal system

Lewis et al. [9] establish the assumption formulation in their framework. Assumptions defined in the system consist of assumption type and assumption description. Besides, to make a convenience for development team members, the system also stores the name of the package, source file that the assumptions are extracted from and the validation state.

2.7.2 Assumption formulation in the formal system

A. S. Tirumala [8] uses name, input parameters, assumption body, Boolean-valued function and classification information to express an assumption. The formulation express the state and how they handled in the framework.

2.7.3 Assumption formulation in Architectural Assumption management model

C. Yang and P. Liang [34] approve that an assumption is made up of assumption name, stakeholders, the description, Pros, Cons and relationships (between assumptions). The formulation of architectural assumption fit better into complex software.

2.7.4 summary

No previous work independently studies the assumption formulation. The assumption formulations are designed to satisfy the purpose of assumption management activities in the software development.

2.8 Assumption lifecycle

I. Ostacchini and M. Wermelinger [30] provide an assumption lifecycle model, which contains three stages: assumption being made, assumption changing and assumption failing. Under the control of assumption management, assumptions can transition to the next stage or stay in the second stage.

Running the assumption lifecycle means that assumptions are transited through all stages of assumption lifecycle model.

2.8.1 Assumption discovery

Assumption discovery refers to that assumptions are created by development teams, which happens in the first stage of the assumption lifecycle model. Here I mainly care about the situation that assumptions are discovered.

A. S. Tirumala [8] agrees that assumptions are frequently discovered when the software contains a large number of components. It is hard for development team members to manage the great quantity of dependencies among the components.

M. M. Lehman [6] mentions that uncertainty is the nature of the world, which also influences computer use in real world. As the result of uncertainty, practitioners and researchers start to pay attention to assumptions, based on which software can be designed.

2.8.2 Assumption changing

M. M. Lehman [6] states that the practical environment of software is dynamic and the designer could not predict the changes and scenarios precisely. So, assumptions show continual changes. The assumption changes reflect the process of software maintenance and evolution.

2.8.3 Assumption failure

The impact of the assumption failures is the negative outcome of assumption failures.

Ö. Albayrak et al. [31] discover that poor experience and computer background of development team members frequently result in assumption failures. the result is generated from the survey.

C. Yang and P. Liang [34] agree that the poor interaction among the complex stakeholders could result in invalid assumptions, because it lead to the inconsistency between software components.

2.9 Assumption treatment

I. Ostacchini and M. Wermelinger [30] summarize four key assumption treatment activities: identifying assumptions, recording assumptions, monitoring assumptions and handling assumption failures.

The following is the necessary activities when managing assumptions with frameworks.

2.9.1 Management activity with the semi-formal framework

Firstly, the system administrator set the users, roles, assumption types and project for system. The project manager assigns users to the roles, assigns users to the projects and assigns roles to the assumption creation and validation. Secondly, development team members record assumptions as structured comments in the source code written in Java (the system also extract assumptions from other language source code) using XML. Thirdly, the development team members use an assumption extractor to search the assumptions in the source code and import them into the assumption repository, which facilitates an email to notify the validator that the assumptions are waiting for the validation. Finally, the validators (system designer, tester, system architect or domain expert) use the assumption management system to search, review, validate assumptions and mark them as valid or invalid. The team members with the right authorization could search and browse assumptions [9].

2.9.2 Management activity with the formal framework

The framework supports development team members directly specifying assumptions and extracting assumptions from the source code. Development team members could input assumptions through the Text Based Input (based on ANTLR Parser Generator) or GUI Based Input (generated by EMF). Using the two methods users do not need to take extra effort to learn the grammar. The framework also supports specifying assumptions directly from Java source code with a source code integration. Looking for a set of properties in the source code that is supposed to be provided by guarantees or monitored by assumptions. Next, all properties are exported and expressed as a return value in a public function. Followed by encoding the assumptions or guarantees through the public functions. (The format of assumptions is described in Section 2.2.4.1). AMF provides a composition algorithm for assumptions (described in 2.2.4.2), which could figure out matched assumptions and guarantees, and mismatched assumptions and guarantees. Users could search and review these assumptions with the component name, dependent component name and assumption/guarantee name. Only matched assumptions and guarantees can be validated. The framework transforms the whole assumption body into Java code and creates a ".class" file for each component. The framework reports all assumption failures in Eclipse. User clicks the invalid assumption and then the framework lists guarantees that cause the assumption failure. DMF [16] is used to analyze error influences in the system with a set of error propagation rules among components that have interaction relations with each other. Assumption violations (used to record immediate influence of an assumption failure) is the input of DMF. DMF analyze the violations and delivery violation impact as the output [8].

2.9.3 Summary

The two frameworks show a strong ability in supporting identifying assumption, recoding assumption and searching assumption. Development team members use the framework to organize and modify assumptions.

2.10 Knowledge gap

Assumption faults are recognized as a primary cause of software project and system failures [13]. A web survey is conducted to investigate the success rate of IT software projects. The fields covered in the investigation contain financial services, computer consulting and system integration, computer software publisher, government, telecommunications, web service, medical and health services and colleges [1]. As shown in the result, about 26% - 34% IT software projects failed. The software project failures cost huge resources and money to companies. For example, London ambulance system took more than ten million pounds and only worked three days before its scrapping, which was mainly due to that the designers falsely assumed that the screen is the most efficient mean to communicate for drivers [6]. The enterprise resource planning project failed in Jordan because the assumptions (about the usage context the development team is responsible for) of the ERP system did not fit into the practical realities of the client organization in the developing country [5].

Assumptions may be made explicit and documented or remain implicit without a conscious act of reflection. Whether implicit or explicit, assumptions may turn into faults when discovered to be wrong. If an invalid assumption can not be modified in time, it must cause the function of the software down [10].

Development teams make assumptions every day [9]. The software system and its operational domains are changing fast [2], leading to the appearance of problems of assumption treatments, like no documenting of significant assumptions, inappropriate validation of assumptions [6], lack of necessary knowledge [13].

The role of assumptions in software engineering has been studied from various perspectives, containing assumption management, assumption and requirement, and software security. An assumption management framework is created to help assumption management practice. The framework does not divide the assumptions into inside assumptions and outside assumptions. They are just named as component assumptions, which contain context assumptions, target user habit assumptions and so on. The framework is based on an assumption documenting language that could express assumptions in the machine-checkable format and support the dynamic assumption validation against software architecture and implementation [8]. There is also a lightweight assumption management framework that could extract assumptions from JAVA programs and import them into the repository [9]. The assumptions in this framework contain control assumptions, environment assumptions, data assumptions, usage assumptions and convention assumptions. There is a temporal mathematical model that describes the relation between assumptions and requirements in the risk prediction context. The risks here are mainly due to that the outside assumptions become invalid [3]. Trust assumptions are mentioned as the basis of security requirements [11].

As today's software tend to be executed in a complex dynamic environment, there are still many implicit and invalid assumptions that result in software project failures [13]. So, it is necessary to improve the effectiveness and efficiency of the assumption management practice. According to the viewpoint of the literature [2] [6] [9] [13], how software development team members handle assumptions is the key factor that influences assumption management. Besides, the previous works can not solve the assumption problems because the solutions are not proposed based on the practical need and purpose of the assumption management.

Therefore, it is needed to solve assumption problems by directly inquiring development teams, since there is little empirical evaluation in the literature (only for architectural assumptions [22]). I also found two frameworks that are highly likely to improve the assumption management practice. An understanding of that perspective would allow proactive discovery of assumptions and ease the work with assumptions – beyond the support of clerical tasks [8] [9]. So, I am going to study how development teams manage environment assumptions in practice and how the teams expect to manage assumptions better tomorrow.

3 METHODOLOGY

The section shows how I conduct the interview research.

3.1 Research goal

Our research goal is finding how software development teams manage environment assumptions in practice and how they might manage environment assumptions better tomorrow. Since today's assumption treatment effectiveness is low and the previous research can not solve the problem, it is effective to use empirical evaluation to reach the research purpose. To be specific, our work is testing the applicability of the frameworks and exploring the assumption type, assumption formulation and assumption treatment in practice. I particularly pay attention to the environment assumption because the it is the major source of the invalid assumptions [2].

3.2 Conceptual framework

Lewis et al. [9] put forward a framework that could improve the efficiency of assumption management activities: recording assumptions, validating assumptions and searching assumptions. Tirumala [8] proposes an assumption management framework that supports the assumption classification (based on the validity of time-frame and criticality), assumption structure (language), assumption selection, assumption validation and assumption-component mapping. I select these two frameworks because these two frameworks are the only two assumption management frameworks I found that are highly likely to improve today's practice. However, the papers about the frameworks do not provide the concrete evaluation from the development team perspective [8] [9].

3.3 Research question

RQ 1 How do development team members consider the advantages and disadvantages of the frameworks?
RQ 2 What main assumption types do development teams value?
RQ 3 How do development teams formulate assumptions?
RQ 4 How do development teams manage assumption lifecycles?

The research questions are answered from the view of product managers. I use interview-based survey to answer the research questions. RQ 4 directly reaches the research goal: assumption management practice. Besides, the good assumption management practice, reason of assumption failures and assumption management frameworks are the potential improvement chance of assumption management practice. Assumption type and assumption formulation have intensive relation with assumption management practice.

I use RQ1 to test the validity of the formal assumption management framework and semi-formal assumption management framework. They are highly possible to support assumption management practice, which is lack of the empirical evaluation [8] [9]. Here I can not give an explicit answer about whether the frameworks are applicable because the practical situation is too complicated. However, I can propose the advantages and disadvantages, which can be combined with the company state that the development team could judge whether to use the frameworks to max their benefits.

Through RQ2, I want to find main assumption types handled in practice because the assumption type has intensive relation with assumption management [13].

RQ3 focus on the assumption formulation, that is, how are assumptions documented or stored. The question has an intensive relationship with the assumption management practice [8] [9].

RQ 4 helps to explore the roles of assumptions in software development. To explore how the assumption lifecycle is controlled, I separately study its three stages: assumption being discovered, assumption changing and assumption failure [30]. Therefore, sub-questions concern the situations that development teams discover assumptions, the activities taken to treat assumptions and the implications of assumption failures. Besides, it is needed to investigate the reasons of assumption failures, which can be used to guide assumption management practice.

3.4 Expected outcome

I expected to form an assumption management model from the view of the product manager. It contains all assumption management activities instead of just product manager part. The model is useful in effectively controlling assumption lifecycle. The model can be used in dealing with environment assumptions. It can be directly merged into software development model by development teams. I also want to explore the advantages and disadvantages of the frameworks [8] [9].

3.5 Research design

3.5.1 Method selection

Interview-based survey research is used to answer the research questions. Survey research is a comprehensive study method with the purpose of comparing, explaining and describing attitude, behavior and knowledge. I have little knowledge about the assumption management practice, so I can only use the interview-based survey to gather preliminary knowledge directly based on the development teams' insights and experiences [12]. The research method has an absolute advantage that digging deep insights when the previous knowledge is little. The interviewees' team does not systematically implement the assumption management, and, the assumption management activities are scattered in the software lifecycle. They are highly likely can not accurately construct their assumption management procedure. Therefore, I need to use discussion to refine interviewees' and my knowledge, and get profound insights of assumption management practice. I use the interview to collect qualitative data for the research questions. At the same time, I use directed content analysis [4] to qualitatively analyze the interview answers. Each interviewee answers the interview questions of the research questions, based on which the data is separately analyzed.

Research method meets our research objective and research feasibility. Easterbrook et al. [14] provide several empirical research methods for software engineering, containing controlled experiments, case study, survey research, ethnographies, action research. Controlled experiments are used to test the relations between the explicit variables, being inconsistent with our research objective. Besides, I can not set the software development condition as particular values, like team size, project scale and so on. Ethnographies requests researcher to participate in the daily development activities. Since I need to look into a typical situation, the only one software project can not represent the board population. Also, it is time-consuming to participate in several projects for me. Action research is to explore scientific knowledge as well as solve real-world issues, which is not consistent with our research goal. Comparing survey research with case study, survey research is better in extracting the knowledge for a common phenomenon of development teams rather than a special case of one development team.

To collect data effectively and efficiently, I use the interview. Easterbrook et al. [14] suggest two first-degree alternative technologies of survey research: interview and questionnaire. Comparing the two approaches. Since I want to be open and guide the development teams for understanding their approach for assumption management with the condition of limited research support, I use interviews as the data collection method. In comparison to the use of questionnaires, interview allows me to have a rich discussion to validate and refine our currently limited understanding of assumption management.

I use directed content analysis for our data. Directed content analysis is used to validate and extend a theory or framework. It also could help to support predicting variables [4]. The interviewees have little knowledge of assumption management, although having been doing it all the time, since they have never systematically implement assumption management. Savin-Baden et al. [7] collect several primary qualitative data analysis methods: keyword analysis, constant comparison, content analysis, domain analysis and thematic analysis. Firstly, considering our research goal, I exclude constant comparison, domain analysis and keyword analysis. Constant comparison is to generate theories; domain analysis mainly focuses on the relations among the categories; keyword analysis is to extract the main meaning from a large number of the data. Secondly, I exclude thematic analysis, because I want to test the applicability of two assumption management frameworks and I have little knowledge of the assumption management practice. Hsieh et al. [4] mention three types of content analysis: conventional content analysis, directed content analysis and summative content analysis. Conventional content analysis and summative content analysis have nothing to do with the framework.

3.5.2 Sampling

I look for the product manager with rich experience in wrong assumptions about the environments of their software. Besides, their answers should be meaning and consistent. Product managers have the profound knowledge about software environment, like market, user, and are responsible for software product strategy [44]. Therefore, they are familiar with environment assumptions. Product managers also guide the development team [44], which means that they have a comprehensive understanding of the assumption management activities. Besides, the interviewees with three years or more experience and bachelor or above educational background could describe their assumption management activities on environment assumptions. For the sample size, I do interviews until the last interviewee can not provide any new idea. To achieve this, I use non-probabilistic sampling: convenience sampling (researchers directly inviting candidates) and snowballing (researchers asking candidates to invite others) because I just want to interview the product managers with rich experience in wrong assumptions of environments.

3.5.3 Data collection

3.5.3.1 Interview question design

Table 3-1. Matching interview question with research question

Research question	Interview question	Sub question
RQ2. What main assumption types do development teams value?	I1. Could you please describe assumptions important in your software development (it is better to mention invalid assumptions)?	
RQ3. How do development teams formulate assumptions?	I2. How do your team documents these assumptions?	
	I3. What are assumptions made by?	
RQ4. How do development teams manage assumption lifecycles?	I4. In which situations do your team discovers assumptions?	
	I5. How do your team treats assumptions?	S1. How do your team identify the assumptions?
		S2. How do your team figure out the values of assumptions?
		S3. How do your team record assumptions?
		S4. How do your team monitor assumptions?
		S5. How do your team validate assumptions?

		S6. How do your team handle assumption failures?
		S7. What tools do your team uses?
	I6. How do you think of the influence of assumption failures?	
	I7. What reasons do you think cause assumption failures?	
RQ.1 How do development team members consider the advantages and disadvantages of the frameworks?	I8. How do you think of advantages of the frameworks?	
	I9. How do you think of disadvantages of the frameworks?	

RQ 2 is directly reflected in the interview questions.

For RQ3, the way assumptions are stored has the significant influence on the assumption formulation.

For RQ4.2 there are three important elements that concern the assumption management: workflow, staff and tool. Most of the companies do not systematically implement assumption management, which means that the interviewees could not construct their work flow easily. To effectively understand how assumptions are treated, I provide some treatment activities (found in the literature) to guide interviewees.

I tested the framework applicability after the assumption lifecycle because it is better to consider the assumption lifecycle and evaluate the frameworks deeply.

3.5.4 Research implementation

After establishing the research question and research method described in the previous section, I, firstly, created the interview questions based on the research questions and prepared the introduction of the frameworks (presented in appendix A).

Secondly, I executed a pilot study by interviewing an experienced product manager because I do not know whether the interviewee could understand the interview questions.

Thirdly, I started collecting and analyzing the data. I implemented Data collection and data analysis in parallel. An advantage of non-probabilistic sampling is that I can control the process of the data collection and the data analysis. I tested the quality of the questions and analyzed the data before starting the next interview. When the new data come, I compared new data with the emergence categories. Before the interview, I emailed interviewees the introduction of the interview (containing the introduction of the frameworks). I used directed content analysis to analyze the data. Hsieh et al. [4] provide a procedure to implement directed content analysis. Firstly, identifying main concepts as initial categories. Secondly, matching operational definitions for each category. Thirdly, reading the transcription and marking all text that concern the key categories. Fourthly, coding all marked text using the initial categories. Finally, identifying all marked text that not be categorized as a new code. Here I asked interviewees to check the transcription to make sure it reflects their real ideas.

3.6 Validity

Here I provide the evaluation of the validity from the below aspects, which are selected to cover the validity of both the research process and research product.

3.6.1 Meaningful coherence

This aspect indicates the extent that the research design and application fit into the research goal and research context.

I considered two approaches to implement the survey. The first one is setting a series of general questions and the research data is completely based on the respondents who answer the questions without guides and limitations. I use what the interviewees express. I use directed content analysis to analyze the text data. The second one is setting a series of specific questions containing the workflow, methods, tools and evaluations. Respondents just need to select which one they agree. I quantitatively analyzed the numerical data. Before deciding the research direction, I reviewed the literature. I approved that the previous works could not present common-used tasks of assumption management. However, they can only provide possible activities. Therefore, selecting the first research direction is reasonable.

3.6.2 Internal validity

This aspect reflects the degree that the data can represent the interviewees' idea and be confirmed by the them.

Since I do not have the assumption management experience with the interviewees, it is highly likely that I misunderstand what the interviewees express. As a result of it, there can be the bias in the transcription and data analysis. To avoid the threat, I asked interviewees to verify the data.

3.6.3 Construct validity

This aspect indicates the extent that the operational measure could represent the real research purpose.

The threat is that our materials delivered to the interviewees could limit their thoughts. To make sure the interviewees have the correct knowledge about the assumptions, I held examples of common assumptions quoted from the literature. At the same time, the interviewees have little knowledge about assumptions and can only construct the concept of the assumption based on our example, which inevitably leads to the bias to the main assumption type and assumption management activities.

To reduce the threat, I asked interviewees to consider "assumption" by themselves first. If they can not do this, I directly tell them its definition from the literature.

The activities provided by product managers may not fit into other requirement assumptions because they mainly treat user habit assumptions and quality attribute assumptions for servers. Besides, the different types of assumptions managed differently. So, development teams should be careful when implementing the assumption management process.

3.6.4 External validity

This aspect reflects the extent that the result can be generalized in other population.

My interviewees are all from Chinese internet companies. Under the environment of software development, China software industry has grown quickly, especially internet and smart phone [46]. China software industries are more and more involved in global market and play an important role in it [47].

However, different regions in the world may have different traditions and practices of how software is built in general and how assumptions are managed in particular. Thus, there may be a need to replicate your study results in other geographical areas, like Europe, Northern America.

3.6.5 Reliability

This aspect reflects the degree that the result is dependent on the researcher.

There is a threat related to the qualitative data analysis. Our data is analyzed by only one researcher, which could lead to the bias in the categorization and synthesis of the data. Different people have different understandings to the same answer of an interviewee. I analyzed the data two times to reduce the bias. However, it is hard to avoid the personal way of thinking. To reduce the impact of the threat, I constantly compared the data of each interview.

The terms I use in the model, like "figuring out the values of assumptions", are just what I create based on the interview answer. There is a possibility that the practical development teams understand my model in an incorrect way. It is better to assign experienced software development team members to implement the process.

3.6.6 Summary

The part presents how I evaluate the research process and research product. They mainly concern method selection, sampling, data collection, and data analysis.

3.7 Summary

I used qualitative interview research to explore the current situation and improvement chance of the assumption management. I mainly balance the validity and feasibility of the research. The interview research is good enough to study a phenomenon with little previous knowledge. The result can be used to guide assumption management practice and support further assumption management studies.

4 RESULTS

This section shows the interview results and their context. The transcription is shown in the appendix B.

4.1 demographic

the background information is listed in Table 4-1.

Table 4-1. background information

	Education background	Experience (year)	Assumption example	Product example	nationality
Interviewee 1	bachelor	3	Assumption about user concurrency for the server ; Assumption about the user habit for the interface	online map application	China
Interviewee 2	master	5	Assumption about the dependencies between components; Assumption about the load for the server	Lenovo Online shopping website	China
Interviewee 3	master	3	Assumption about user habit for function; Assumptions about the outside technology; Assumption about user habit for the interface	Mobile management application	China
Interviewee 4	master	5	Assumption about user concurrency for the server; Assumption about user habit for the interface	Online shopping website	China
Interviewee 5	Bachelor	5	Assumption about user habit for functions	Online game management application	China
Interviewee 6	Bachelor	3	Assumption about the usage context for functions; Assumption about user habit for the interface	Bicycle-sharing	China

In order to get as validate results as possible, it is needed to limit the educational background and experience of the interviewees. I approve that an interviewee with a bachelor degree or above educational background has enough ability to understand the interview questions, and describe and evaluate their work. An interviewee with three years or more experience could have the comprehensive understanding of their assumption management practice.

I select product managers because they are responsible for investigating, selecting, driving the development of products for an organization and performing the activities of product management [15]. Assumption management activities happen in the whole lifecycle of the software, which is fully understood by the product managers. Also, product managers have a comprehensive understanding of environment assumptions.

4.2 overview of the interview result

4.2.1 interview 1

The first interviewee is a product manager worked in an internet company. The company specializes online map applications and involves several to-Business or to-Customer services.

This company has a mature protocol in developing the requirement and architecture. The company has encountered some assumption-related problems. Considering the problems being not serious, they do not pay much attention to the assumption management.

The interview approves that the frameworks' functions are covered and too much effort needed to implement them.

They are familiar with user habit assumptions and concurrency assumptions for servers.

The user habit assumptions consist of name, description, values and image, while the quality attribute assumptions are composed of name and values.

He mentioned that the assumption is discovered because of its frequent change or the high benefit to the company. In the interviewee's company, assumptions are investigated in the software design. They are distributed stored in the software documents. In this phase, the development team also prepares the solution of assumption failures and implement real-time monitoring on the state of the assumption during the application running. To validate assumptions, they use online data statistics and software tests. In order to create more benefits, assumptions are set to meet the most of the users. If the invalid assumptions can not be modified in time, it will cause the decline of the user access, product sales and benefit. The failures of assumption management are caused by the following factors: poor development team members' understanding of assumption, unclear production orientation, product strategy, low operation standardization and low experience of product manager. The root factor is the team ability and experience.

4.2.2 Interview 2

The second interviewee is a product manager that has joined in a project of developing and maintaining a module of an online shopping website. This application has been run for many years. Some environment assumptions, like system load, execution environment, target user, are fixed. At the same time, their assumptions expressing the dependencies between components are very complex

The interviewee points out the assumption-component mapping of the formal assumption management framework is helpful in finding out the resource of the software problem. Besides, the two frameworks could improve assumption management practice. In practice, Cost-benefit is the starting point that evaluates a framework before using it. It must be analyzed and trialed.

She agreed that her company mainly pays attention to assumptions about the load for servers and the dependency for components.

The load assumption formulation contains name, description, value and version. The dependency assumption formulation includes name, description, value and mapping of dependent component-dependency relation.

Assumptions' frequent changes cause development team members to discover them. Her company implements online real-time monitoring and software test to avoid the negative influence of invalid assumptions. Before it, they just do a simple adjustment based on the historical data when figuring out the value of assumptions. Assumption failure could cause the loss of the sales and money. Reasons that lead to the assumption-related problem are not comprehensive understanding of the component relation in complex software projects, poor software testing, not enough communications between development team members, poor development team member experience and ability. The development team member is the most important factor.

4.2.3 Interview 3

The third interviewee is a product manager. He has taken part in more than one development teams. The company benefit is their final goal when setting the values of the assumptions.

The interviewee agrees that the assumption-component mapping function in the formal framework could help to analyze the problem of the software in time. However, it takes much effort before realizing the framework.

His team mainly focus on assumptions about users and outside technologies.

The outside technology assumptions are composed of name, description, value and vendor. The user habit assumption formulation contains name, description and value.

The interviewee mentioned that the assumption could be discovered when it is required to develop a new module. In his company, the development team decides handling assumptions based on data statistics, company profit and user need. They use data statistics, software test, manual reviewing to validate assumptions. The vendors notify the update of the outside technology and interface in advance. It is development team members who deal with the problems in detail and create the solutions. He also pointed out the influence of the invalid assumption: loss of profit, delay of development schedule, change of development activity, loss of user and increase of human resource and time. Poor development team members' ability, poor development team members' experience and incomplete software test always give birth to assumption-related problems, which are not serious. Besides, human resource, time, cost and benefit are important when a company starts to use a new framework.

4.2.4 Interview 4

The fourth interviewee is a product manager. She has taken part in website development. Her products are pushed to different countries with a great number of population. So, she often treats user habit assumptions.

He thinks of the frameworks can be used to process a large number of assumptions, though functions of the frameworks are covered. Also, the effort of realizing the frameworks is high. The assumption-component mapping function of the formal assumption management framework could improve the effectiveness of decision making.

He mentions two main assumptions handled in the software development: assumptions about user habits and concurrency for servers.

All assumptions consist of name and value.

They discover the assumptions when they frequently change. Four activities in his company that concern assumption management: figuring out the value of assumptions, assumption monitoring, assumption validation and handling assumption failures. First of all, she figure out the value of assumptions with their historical data, development team members' experience and investigation. Also, the backup assumption values are necessary because they can not guarantee the assumption is accurate. To make the assumption more reliable, they take into account resource, technical support and time, and benefit. Next, her company uses a website development platform to monitor and validate assumptions since they expertise web application. They review the web log files to deal with assumption failures. The fourth interviewee mentioned that the assumption failures result in the system function down, user number decline and law problems. The interviewee provides three reasons that lead to invalid assumptions: poor development team members' ability, experience and understanding of requirements.

4.2.5 Interview 5

The fifth interviewee is a product manager that is responsible for the mobile front-end development. He mainly works with function-related assumptions. Their key target is maximizing user amount.

This interviewee agrees that the two frameworks could help to manage assumptions and the assumption-component mapping function of the formal framework is useful in decision making.

About the main assumption they handle, they pay attention to assumptions about the user habits for functions and user context.

The assumptions are composed of name, description, value and risk point.

The interviewee's company discovers assumptions when they think of the assumption is necessary to the requirement. He provided four activities: figuring out the value of the assumption, assumption monitoring, assumption validation and assumption failure handling. His team mainly works on function-related assumptions, so, they use "5W3H" analysis method to construct usage context assumptions. Also, they figure out the value based on their development team members' idea and data statistics. They also make backup assumption values and analyze their risks. Next, the development team members always pre-validate function-related assumptions before the software execution. The interviewee’s company implements an online tool to monitor assumptions. If the assumption becomes invalid, the development team members could analyze risk points and select the backup assumption values according to their experience. The interviewee mentioned that the main outcome of assumption failures: user amount decrease. He mentions several reasons that result in assumption-related problems: poor understanding of risk, poor understanding of market context, and poor ability and experience of development team members.

4.2.6 Interview 6

The sixth interviewee is a product manager that has touched mobile application development (containing hardware development and software). He is experienced in software part. He has rich experience in function-related assumptions.

The interviewee agreed that the assumption frameworks are applicable when assumption number is high, although the functions of the frameworks are covered. Assumption mapping function is useful in analyzing the problems.

His company mainly pay attention to user habit assumptions and user context assumptions.

The user habit assumptions consist of name, description, value and image. The user context assumption formulations are name, description, value and corresponding requirement.

The interviewee discovers assumptions that are necessary to requirements. He provided four activities: figuring out the value of assumptions, assumption monitoring, assumption validation and handling assumption failure. Firstly, they figure out the value of the assumptions based on their experience and investigation. They always validate the assumptions before implementing the application. Online data statistics tool is used to monitor the assumptions. If the assumptions fail, the development team members handle the assumption failures according to their experience and data. Assumption failures directly lead to the decrease of the user amount. He provided three reasons that cause assumption failures: Poor ability and experience of development team members and improper product orientation.

5 ANALYSIS

This section presents the analysis the answers of the interviewees.

5.1 RQ 1 How do development team members consider the advantages and disadvantages of the frameworks?

The advantage is that assumption mapping function of the formal assumption management framework is useful. The disadvantages are that it takes too much effort to implement the frameworks and their functions are covered.

5.1.1 Synthesis

Table 5-1. data synthesis of RQ 1

	I1	I2	I3	I4	I5	I6
Taking too much effort	yes		yes	yes		
The functions are covered	yes			yes		yes
Improving the efficiency of assumption management		yes			yes	
Assumption mapping is useful		yes	yes	yes	yes	yes
They are used when the number of the assumption is large				yes		yes

As shown in the above table, the assumption-component mapping of the formal framework (I2, I3, I4, I5, I6) is useful in making decisions and analyzing the problem. The strength is mentioned five times. The function is achieved by an independent component in the formal framework, which uses an algorithm to map the assumption with the component. If an assumption becomes invalid, the function could find out which software components cause the invalid assumption and which software components are influenced by the assumption.

"For the frameworks, I think assumption-component mapping in the formal assumption framework is significantly helpful."

It takes too much effort to implement the two frameworks (I1, I3, I4) and the functions of the frameworks are achieved by the developers and existing tools (I1, I4, I6). The two weakness are separately mentioned three times. Normal companies have their own protocol of developing software. If a new tool does not have new functions and takes too much effort, no one will use it, because the benefit is the top priority for all companies. Functions of the frameworks are covered by the existing tools and development team members in the software development. Besides, to realize the frameworks, the development team members need to insert the assumption expressions into the source code and construct the platform, which is time-consuming and resource-consuming. Generally, they are not willing to find a new solution before there is a problem.

"To realize the frameworks, the development team should create the assumptions, establish the execution environment of the framework, which can not be achieved if the problem is an emergency or the development cycle is short."

"However, the two functions are not applicable in practice. For tracing assumptions, the product manager could document assumptions unsystematically with few problems happened".

Interviewee 2 and interviewee 4 agree that the two frameworks could improve the efficiency of assumption management practice, but did not explain their idea.

"The two frameworks integrated with the powerful software computing ability should be useful in assumption management practice."

Interviewee 4 and interviewee 6 agreed that the frameworks could only be used when the amount of the assumption is large.

"The notification of the assumption validation is not needed because assumption number is not large."

5.1.2 Taking too much effort

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-2. data analysis for RQ1 (italic: quoted from the answer of the interview)

Sentence	Interview number
I do not think a systematic framework is needed in practice if the framework is hardly realized.	1
We can not predict the loss, but it takes too much time to implement a new framework.	
To realize the frameworks, the development team should create the assumptions, establish the execution environment of the framework, which can not be achieved if the problem is an emergency or the development cycle is short.	3
Implementing the frameworks take too much effort and resources.	4

5.1.3 The functions are covered

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-3. data analysis for RQ1 (italic: quoted from the answer of the interview)

Sentence	Interview number
Besides, assumption management is an essential skill of product manager.	1
However, the two functions are not applicable in practice. For tracing assumptions, the product manager could document assumptions unsystematically with few problems happened.	
Our team has the clear division of the responsibility.	
The another reason that the frameworks are not applicable is that the existing project management practice can cover their functions, like software testing, software verification. There are few risks that could result in software problems	
Our platform has enough ability to support website development. It is not worth using a new framework since there are no serious problems happened.	4
Also, software test could help to decrease the assumption-related problems.	6

5.1.4 Improving the efficiency of assumption management

The transcription is presented in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-4. data analysis for RQ1 (italic: quoted from the answer of the interview)

Sentence	Interview number
The two frameworks integrated with the powerful software computing ability should be useful in assumption management practice.	2
The formal framework is more helpful in improving the efficiency of assumption management.	5

5.1.5 Assumption mapping is useful

The transcription is shown in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-5. data analysis for RQ1 (italic: quoted from the answer of the interview)

Sentence	Interview number
For the frameworks, I think assumption-component mapping in the formal assumption framework is significantly helpful.	2
The assumption-component mapping in the formal framework could help to find out the problem of the software.	3

The frameworks should be helpful in detecting the problems in time.	
However, the assumption mapping function is useful when making decisions.	4
The mapping function could provide additional information to help make the decision.	5
Maybe the assumption mapping function could help to analyze the reason for the problem.	6

5.1.6 They are used when the number of the assumption is large

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-6. data analysis for RQ1 (italic: quoted from the answer of the interview)

Sentence	Interview number
Also, I think our assumption problems are not due to improper management of a larger number of assumptions.	4
The notification of the assumption validation is not needed because assumption number is not large	6

5.2 RQ 2. What main assumption types do development teams value?

The main assumptions handled in practice are assumptions about user habit and assumptions about quality attributes for the server.

5.2.1 Synthesis

Table 5-7. data synthesis of QR 2

	I1	I2	I3	I4	I5	I6
Assumptions about user habit for the interface	Yes		yes	yes		yes
Assumptions about user concurrency for the server	yes			yes		
Assumptions about the load for the server		yes				
Assumptions about the dependency for components		yes				
Assumptions about the outside technology			yes			
Assumptions about user habit for functions			yes		yes	
Assumptions about the usage context for functions					yes	yes

From the above table, I can see that each product manager is worried about the limited number of assumptions.

Assumptions about user habit for the interfaces are discussed most (I1, I3, I4, I6) and assumptions about user habits for functions are mentioned by interviewee 3 and interviewee 5. They are both user habit assumptions. For most companies, the user habit assumption is inevitably involved in the software design or requirement analysis. Generally, the software with the good assumption of the user habit could attract users' attention. So, assumptions about the user habit are the most important in the software development.

"When we decide whether to use the round buttons or the square buttons in the user interface, we need to make the evaluation. If we approve that the round button can attract users' attention and the product manager does not do online verification (the data shows whether there are more users click the button after it becomes round), it can not bring serious problems."

"We developed a memory management application, one of the functions is cleaning applications in the memory. It is the development team that establishes the protocol about which application should be deleted. It is highly likely that the application is removed by mistake because the development team members can not expect all scenarios."

The assumption about the concurrency for servers is mentioned by interviewee 1 and interviewee 4. The assumption about the load for servers is discussed by interviewee 2. So, assumptions about quality attributes of servers are mentioned three times. These quality attribute assumptions could result in serious software problems if they become invalid, and the servers are the important part of many internet-related software, which means that the kind of assumptions can not be ignored in internet industries.

"The user concurrency has an intensive relation with architecture design. When I know an approximate amount of the user and approximate increment of the user (history data), based on which we can set how many users that the system needs to support. Our website provides service for the users from 120 countries. At the early stage of we assuming the user concurrency, the amount of the user is about 60000 in 120 countries, so, we assume the user concurrency is 5000. Since 120 countries are in different time zones and the time of the website implementation is short, we think there are limited users on the website. After less than half a year, the marketing department held a promotion activity. The user concurrency sharply increases to 10000, and the system is down in a short time."

"when we consider the system load for the next version of the application, we would make a certain percentage growth of the system load based on the original system architecture and use hardware to support it. We also monitor the real-time load during the system being operated. The growth percentage is made by comparing the system load in recent years, which could guarantee the system can run and not too much waste of the resource."

The other assumptions are also described in detail. However, they are not common in practice.

5.2.2 Assumption about user habit for the interface

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-8. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
when we decide whether to use the round buttons or the square buttons in the user interface, we need to make the evaluation. If we approve that the round button can attract users' attention and the product manager does not do online verification (the data shows whether there are more users click the button after it becomes round), it can not bring serious problems	1
like interface design, the product managers assume what design style could attract users' attention. In practice, it is highly likely that the users' scenarios are not what the designer expected, which lead to the users ignoring the functions.	3
The assumption about the user operation habits is the another key assumption in our website development. Our target user is Foreigners, while the designer is Chinese. We need to design workflow, for example, register workflow (containing "back" button, "continue" button and "cancel" button). We put the buttons on the right side of the page and put the questions on the left side because we assume it is convenient for the users. However, the users in different countries have different habits, some of the users want that the buttons and questions are on the same side. We, finally, highlight the color of the button to meet the need of all users. The reason for the assumption failure is that we do not realize that the users have different habits.	4
The assumption of user habit concerns the user interface design. One of the reasons that the users do not use a function is that they are not aware of the function. So we add the highlight or dynamic effect on the function.	6

5.2.3 Assumption about user concurrency for the server

The transcription is shown in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-9. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number

In other situations, the factors, like user concurrency, is changing every day, which deserves being studied because different software or the same software in different time need different environments.	1
The user concurrency has an intensive relation with architecture design. When I know an approximate amount of the user and approximate increment of the user (history data), based on which we can set how many users that the system needs to support. Our website provides service for the users from 120 countries. At the early stage of we assuming the user concurrency, the amount of the user is about 60000 in 120 countries, so, we assume the user concurrency is 5000. Since 120 countries are in different time zones and the time of the website implementation is short, we think there are limited users on the website. After less than half a year, the marketing department held a promotion activity. The user concurrency sharply increases to 10000, and the system is down in a short time.	4

5.2.4 Assumption about the load for the server

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-10. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
when we consider the system load for the next version of the application, we would make a certain percentage growth of the system load based on the original system architecture and use hardware to support it. We also monitor the real-time load during the system being operated. The growth percentage is made by comparing the system load in recent years, which could guarantee the system can run and not too much waste of the resource.	2

5.2.5 Assumption about the dependency for components

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-11. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
The most significant problem is that the development team members can not realize the relations between two modules. They always change one module with ignoring its influence on another module. For example, one of the requirements is that the global address format of the receiver is configurable. Allowing for there is the various address formats in the various countries, like the United States is divided into states and China is divided into Provinces, the address in registration must be configured according to the different countries. At the same time, another module, like payment check is also needed to be updated to be consistent with the registration module. If it is not found, the function can not be used	2

5.2.6 Assumption about the outside technology

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-12. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
If the outside technology or interface is needed to be updated, they would inform us in advance. We can adjust the corresponding module to fit the update of the outside technology.	3

5.2.7 Assumption about user habit for functions

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-13. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
we developed a memory management application, one of the functions is cleaning applications in the memory. It is the development team that establishes the protocol about which application should be deleted. It is highly likely that the application is removed by mistake because the development team members can not expect all scenarios.	3
in registration module, we add an extended information in the final step of the registration process to guide the users to share the application.	5
If we decide to push eight games， we will push four top games and four games that we expect the users to download. The four quotas are assigned to the advertisement. The four top games are the same category as the advertisement games are. We assume the top games could attract users' attention and promote the users to download advertisement games.	

5.2.8 Assumption about the usage context for functions

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-14. data analysis for RQ2 (italic: quoted from the answer of the interview)

Sentence	Interview number
I think the functions created based on using context. So, if the assumption is important, we will realize it.	5
We found a situation that people go from the home to the subway station. If the distance between the home and the subway station is about the 2-3 kilometer, it is too far to walk. At the same time, it is not worth taking a taxi. So, it is convenient to use a bicycle. However, customers can not bring the bicycle on the subway. So, these people need two bicycles, which is not practical. Because the cost of buying and maintaining two bicycles is too high for people that need to go to work by subway. We assume that there is a kind of bicycle that can take users from the home to the subway station and from the subway station to the workplace, and does not require maintenance from the users. We need to investigate whether there are many people have this kind of problems. If the number of the people is large enough, we will consider solving the problem from the software perspective, like developing an application to manage the bicycles. Here we could validate the assumption after designing the software.	6

5.3 RQ 3. How do development teams formulate assumptions?

User habit assumptions are composed of name, description and value, while quality attribute assumptions of the server consist of name and value. The number of assumptions is not large enough to document them independently, which are always treated with requirements. So, most development teams just store the basic information of assumptions. Assumption formulations are set based on the need of software development.

5.3.1 Assumption about the user habit

5.3.1.1 Synthesis

Table 5-15. data synthesis of user habit assumptions for RQ3

	I1	I3	I4	I5	I6
Name	yes	yes	yes	yes	yes
Description	yes	yes		yes	yes
Value	yes	yes	yes	yes	yes
Image	yes				yes
Risk point				yes	

As described in the above table, there are three elements required by user habit assumptions: name (mentioned in I1, I3, I4, I5, I6), description (mentioned in I1, I3, I5, I6) and value (mentioned in I1, I3, I4, I5, I6).

5.3.1.2 transcription

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-16. data analysis for RQ3 (italic: quoted from the answer of the interview)

Sentence	formulation	Interview number
The structure of the user habit assumption is more complex, containing the name, description, value and image. Using images help improving the efficiency of the assumption validation	Name, Description, Value, Image	1
For user habit assumptions, we simply need its name, description and value	Name, Description, Value	3
We store the name and value with the corresponding requirement for all type assumptions.	Name. Value	4
We store not only the name, description and value, but the also risk point because we need change the assumption value by analyzing the risk point of the assumption failures.	Name, Description, Value, Risk point	5
For the interface, we mainly record its name, description, value and image to be convenient.	Name, Description, Value, Image	6

5.3.2 Assumption about the quality attribute for the server

5.3.2.1 Synthesis

Table 5-17. data synthesis of quality attribute assumption of the server for RQ3

	I1	I2	I4
Name	yes	yes	yes
Value	yes	yes	yes
Description		yes	
Version		yes	

From the synthesis shown in the table above, name is mentioned three times (I1, I2, I4), value is mentioned three times (I1, I2, I4), and Description and version are separately mentioned one time.

5.3.2.2 transcription

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-18. data analysis for RQ3 (italic: quoted from the answer of the interview)

Sentence	formulation	Interview number

For the concurrency assumption, we only record its name and value.	Name, Value	1
For the load assumption of servers, we store its name, description, value, and version. The version is needed because we expect to predict the value of the new version software by calculating the trend of the past version.	Name, Description, Value, Version	2
We store the name and value with the corresponding requirement for all type assumptions.	Name, Value	4

5.3.3 Other assumptions

5.3.3.1 Synthesis

The three types of assumptions are not common and show different formulations, which means that different teams establish the assumption formulation based on their need of software development.

5.3.3.2 transcription

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-19. data analysis for RQ3 (italic: quoted from the answer of the interview)

Sentence	Assumption	formulation	Interview number
For the dependency assumptions, we document their name, description, value and component-relation mapping.	Dependency assumption	Name, Description, Value, component-relation mapping	2
while the name, description, value and vendor are required for outside technology assumptions because vendors could provide the technology according to our need and notify us if the technology is changed.	Outside technology assumption	Name, Description, value, vendor	3
For the user context assumption formulation, we record assumptions' name, description, value and corresponding requirement. We store the corresponding requirement because they assume the user context according to the purpose of the software before analyze the requirement in details.	User context assumption	Name, Description, Value, Corresponding requirement	6
We store not only the name, description and value, but the also risk point because we need change the assumption value by analyzing the risk point of the assumption failures.		Name, Description, Value, Risk point	5

5.4 RQ 4. How do development teams manage assumption lifecycles?

Assumptions are discovered when they frequently change and are necessary to the requirements. The major assumption treatment activities are figuring out the value of assumptions, assumption monitoring, assumption validating and handling assumption failures. Many companies use the development platform to support assumption management. The aim of assumption management is maxing the company benefits and user amount. Assumption management is achieved by all team members. The major outcome of assumption failures is the loss of users and benefits. The major reasons for the assumption failures are the poor ability and experience of development team members.

5.4.1 The situation that the assumption is discovered

5.4.1.1 Synthesis

Table 5-20. data synthesis of RQ 2.1

	I1	I2	I3	I4	I5	I6
Assumption changes	yes	yes		yes		
benefit of assumptions	yes					
necessity to requirements			yes		yes	yes

From the result of the interview, two main factors make developers to discover assumptions: assumption change (I1, I2, I4) and necessity to requirement (I3, I5, I6). Also, the benefit of the assumption needs to be considered because it is the final goal of all companies.

For assumption changes, some companies have many year experience in one domain. They do not need to take more effort to study these kinds of assumptions. Most of the kinds of assumptions are environment assumptions or usage assumptions.

"As I understand, the Chinese software companies do not manage assumptions systematically, which is mostly due to that it is not worth doing this. For example, some knowledge, like using AliCloud in our software and Java to code, is very fundamental for our software development, which can not be changed."

For the necessity to requirements, studying this criterion is more important to function-related assumptions. If the developers are not familiar with the requirement, they always need to assume using context before developing requirements.

"If we plan to develop a new function, we tend to do some pre-investigations and assume the technical support, resource, middleware or interface that concern the function. We take these assumptions as the basis of the new function."

5.4.1.2 Assumption changes

The transcription is shown in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-21. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
If I am responsible for the service that the software provides, assumptions about the target users, execution environment, server are not mentioned in our document because these factors are not changed in our software project.	1
As I understand, the Chinese software companies do not manage assumptions systematically, which is mostly due to that it is not worth doing this. For example, some knowledge, like using AliCloud in our software and Java to code, is very fundamental for our software development, which can not be changed.	
In other situations, the factors, like user concurrency, is changing every day, which deserve being studied because different software or the same software in different time need different environments.	
Generally, large software companies have a relatively accurate judgment of the outside assumption of the application.	2
If an assumption has not been changed for a long time, I will not pay attention to it.	4

5.4.1.3 The benefit of assumptions

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-22. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
To peruse benefit, Some Chinese companies just add services with no change of software environment.	1
The target user changes are largely because the project leader approves that other users could bring more profits. For to-Business, if the customer change, it results in a new project.	

5.4.1.4 The necessity to requirements

The transcription is presented in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-23. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
If we plan to develop a new function, we tend to do some pre-investigations and assume the technical support, resource, middleware or interface that concern the function. We take these assumptions as the basis of the new function.	3
I think the functions created based on using context. So, if the assumption is important, we will realize it.	5
For an assumption, if it has an intensive relation with a requirement, like the interface design, we are willing to pay attention to it.	6

5.4.2 Assumption treatment activity

5.4.2.1 Synthesis

Table 5-24 data synthesis of RQ3.1

	I1	I2	I3	I4	I5	I6
Figuring out the value of assumptions	yes	yes		yes	yes	yes
Assumption recording	yes					
Assumption monitoring	yes	yes		yes	yes	yes
Assumption validation	yes	yes	yes	yes	yes	yes
Handling assumption failure	yes		yes	yes	yes	yes
Decision making			yes			

As shown in the above table, main assumption treatment activities are figuring out the value of the assumption, assumption monitoring, assumption validation and handling assumption failure.

For figuring out the value of the assumptions (I1, I2, I4, I5, I6), Development teams always use investigations, data statistics, their own experience. If the company is familiar with the domain, they just do a simple modification of assumption based on the history data. They take into account resource, technical support and time duration to realize the assumptions. For some assumptions, some of development teams also need to prepare a backup value of the assumptions. Interviewee 5's company analyzes the risks of each assumption. They avoid the non-core risks through some pre-investigations and set a backup value to solve core risks.

"Assumption-related activities are important in software design that development team members need to do investigations about the software environment, like target user, execution environment, usage environment and so on."

Assumption monitoring (I1, I2, I4, I5, I6) is always supported by online tools. The development teams implement the real-time monitoring on assumptions because they can not accurately figure out the value of assumptions.

"Monitor the assumptions by online monitoring and user feedback."

Assumption validation (I1, I2, I3, I4, I5, I6) is achieved through software test, and developers checking the data. Only the platform in interviewee 4's company could validate the assumptions without developers. If the value is abnormal, the platform will inform the development team members. In interviewee5's and interviewee6's company, their function-related assumption could be validated before implementing the application. In interviewee 3's company, the developers need to review the code to validate the assumptions. Also, when using the technology from other company, they will make a notification when the technology is updated.

"the product manager does not do online verification (the data shows whether there are more users click the button after it becomes round)."

Assumption failures (I1, I3, I4, I5, I6) are solved based on developers' experience data and data statistics. It is developers that analyze the problem and decide how to deal with the failure. To solve the invalid assumption more efficiently, the backup assumption values are prepared in practice. In interviewee 5's company, they also match the reasons that cause the assumption failure with risk points of the backup plan.

"We could review the website state from the log file, and compare the website state with parameter we set, based on which, we could implement an emergency treatment for the problem. We do not always adjust the value we set, because it can cause a series of effects on the website. "

Interviewee 1 mentioned that assumptions are not systematically stored in one document.

Interviewee 3 mentioned decision making in the assumption treatment. The decision is made about figuring out the value of the assumption and handling assumption failure. When making decisions, they always balance user experience and company benefits.

5.4.2.2 Figuring out the value of assumptions

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-25. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
Assumption-related activities are important in software design that development team members need to do investigations about the software environment, like target user, execution environment, usage environment and so on	Assumptions are investigated in software design	1
I am working on an online shopping website. The application has run for a long time, so, some outside assumptions, like server load, response time or target user, can be estimated accurately. For example, when we consider the system load for the next version of the application, we would make a certain percentage growth of the system load based on the original system architecture and use hardware to support it. The growth percentage is made by comparing the system load in recent years, which could guarantee the system can run and not too much waste of the resource.	Using history data and do a simple increment.	2

When I know an approximate amount of the user and approximate increment of the user (history data), based on which we can set how many users that the system needs to support.	Historical data	4
If the same event is held, we take the value as a reference.		
we use a matrix to adjust the factors and manage the user concurrency. The matrix has not been mature. It can just help us to predict an approximate value of assumptions only.	matrix	
The main factors are the events that our company will hold, the amount of the user and so on. Since we do not know whether the value is accurate, we need to monitor the factors through the matrix. If the fluctuation the value is too sharp, the development team member will modify the corresponding parameter. We record the value adjusted.		
So, at the early phase of the website design, we make the assumptions based on our experience	Development team members' experience	
We expect to set assumptions to meet the most of the users and prepare the backup plan for the special case.	Backup value	
It is the user that let us know they want the questions and buttons are on the same side.	investigation	
In other situation, if the system could support too many scenarios, it must reduce the performance of the website. We must consider the resource, technical support, time that realize the assumption, and benefits brought from the assumptions.	Resource, technical support and benefit of assumption influence figuring out the value of assumption	
During the requirement analysis phase, we need to assume the scenario that users use the application by using "5W3H" analysis method.	5W3H	5
We always make assumptions based on our experience and suggestions from other designers.	Team members' experience	
Next, categorizing all ideas.		
So we need to change the function with the backup function. We always prepare backup functions in the design phase.	Backup value	
We also analyze the risks for each of them. Through the risk analysis, we decide which assumption plan will be used.	Risk analysis	
We predict risks and prioritize them before designing the application functions. We could avoid the non-core risks by doing some pre-investigations and set a backup design to solve core risks.		
We have many of games in the library. The top game is selected according to the download history of the user. The benefit is the key factor in choosing the advertisement games. The information considered to make a decision is generated from the data statistics. The data statistics is achieved by software.	Data statistics	

However, customers can not bring the bicycle on the subway. However, customers can not bring the bicycle on the subway. So, these people need two bicycles, which is not practical. Because the cost of buying and maintaining two bicycles is too high for people that need to go to work by subway. We assume that there is a kind of bicycle that can take users from the home to the subway station and from the subway station to the workplace, and does not require maintenance from the users.	Team members' experience	6
We need to investigate whether there are many people have this kind of problems. If the number of the people is large enough, we will consider solving the problem from the software perspective, like developing an application to manage the bicycles	investigation	

5.4.2.3 Assumption recording

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-26. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
being recorded in the software design document, which tends not to be achieved systematically.	Assumption is not systematically stored in document	1

5.4.2.4 Assumption monitoring

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-27. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
There is no serious problem happens in the internet company because we implement 24-hour stand by. Maybe there are some problems solved in a short time that I do not know.	Real-time monitoring of the assumption state	1
We also monitor the real-time load during the system being executed.	Online real-time monitoring	2
We use a website development platform for developing a website. The platform supports adjusting the website execution parameters to guarantee the normal running of the website. The platform is needed to be monitored continuously by development team members, which can not be achieved in practice. So, we set the value that slightly higher than the value estimated.	Website development platform	4
monitor the assumptions by online monitoring and user feedback.		
We have an online tool to monitor the state of the application, which could present the change of the users.	Online monitoring	5
The online tool could help to monitor the user feedback.	data statistics	6

5.4.2.5 Assumption validation

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-28. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
the product manager does not do online verification (the data shows whether there are more users click the button after it becomes round),	Online data statistics	1
The another reason that the frameworks are not applicable is that the existing project management practice can cover their functions, like software testing	Software test	
Not all of this kind of problems can be detected in the software testing	Software test	2
In our company, there is a data team that specializes the data of user behavior, like time on site, user access. Next, they can compare the user behavior data before and after the assumption being realized to validate the assumption. If there is a sharp change after the assumption is realized, we can judge the assumption state.	Data statistics	3
If the outside technology or interface is needed to be updated, they would inform us in advance. We can adjust the corresponding module to fit the update of the outside technology.	The update of the outside technology and interface is notified to us.	
The problem is mainly due to the poor experience of the development team members and incomplete test case	Software test	
we set a critical value of the assumption. If the assumption reaches the critical value, the platform will notify the development team members, who could adjust the value of the assumptions in a short time.	Website development platform	4
testing their validity with users before application execution.	Pre-test	5
If the number of the people is large enough, we will consider solving the problem from the software perspective, like developing an application to manage the bicycles. Here we could validate the assumption after designing the software.	Pre-validation	6

5.4.2.6 Handling assumption failure

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-29. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
For our software, if user concurrency sharply rises in a short time, the maintainers could solve this problem by increasing the number of the server in a few minutes, which is planned before implementing the product.	Planning handling assumption failure	1
The development team analyzes the problems based on the data statistics. Every sub-teams of the module also could analyze the problems themselves.	Development team members	3

	analyze the problem based on the data	
For example, we assume that it is database problem that influences response time, leading to the decline of the user access. If we want to find the reason that results in user access decrease, we can analyze the application state data. The data analysis platform could review the execution state of each module, if there is a large number of the data of users can not be accessed normally, there must be problems in the database.	Checking the state if the modules one by one	
The development team members also review the code (comparing the update between the two versions) to solve the problem,	Reviewing the code	
If the system has some problems, we need to invite the related personnel to analyze the problems.	Team discussion	4
We could review the website state from the log file, and compare the website state with parameter we set, based on which, we could implement an emergency treatment for the problem. We do not always adjust the value we set, because it can cause a series of effects on the website.	Reviewing the website log file	
We, firstly, analyze the the risk points (like game's icon, game's introduce information and context that user selects the games).	Risk analysis	5
Next, we use the backup design and test it.	Backup value	
We analyze the problem based on our experience.	Team members' experience	
We would break down the solution into small components and test them one by one. We will select to modify the function or delete the function. The analysis depends on our experience and data of the statistics	Data statistics and team members' experience	6

5.4.2.7 Decision making

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-30. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Method	Interview number
In our company, most of the decision making is based on data statistics.	data statistics	3

5.4.3 Tooling

All of the companies interviewed have the application development platform that supports monitoring and managing the application. Using these auxiliary tools helps developers to make decisions and analyze problems better.

"We use a website development platform for developing a website. The platform supports adjusting the website execution parameters to guarantee the normal running of the website. The platform is needed to be monitored continuously by development team members, which can not be achieved in practice. So, we set the value that slightly higher than the value estimated."

Interviewee 4's company has developed a matrix to figure out the value of the assumption. In most of the situations, the value of an assumption can be influenced by many factors. They use this matrix to balance these factors and predict the value.

"We use a matrix to adjust the factors and manage the user concurrency. The matrix has not been mature. It can just help us to predict an approximate value of assumptions only."

Interviewee 5 mentions that they use "5W3H" analysis method to assumption using context. This method is constructing a practical problem happened by figuring out the people involved, its content, its location, its time, its reason, the method to evaluate it, its severity and how users feel about it.

"During the requirement analysis phase, we need to assume the scenario that users use the application by using "5W3H" analysis method."

5.4.4 Aim of assumption management

The final goal of assumption management is maxing benefits of the company and user amount. Thought the idea is just mentioned by interviewee 1 and interviewee 3, all companies arrange their activities according to the aim.

"Our product solutions could meet the need of the most of the users after we balance the software features for different users."

"Two factors that mainly influence the development team making the decision: company profit and user need. The development teams peruse the best balance between the two factors."

5.4.5 Roles involved in assumption management

Assumption management activities concern all team members, which is mentioned by interviewee 3.

"The decision is made by the development team."

5.4.6 The implication for assumption failure

5.4.6.1 Synthesis

Table 5-31. data synthesis of RQ2.2

	I1	I2	I3	I4	I5	I6
Decline of use number	yes		yes	yes	yes	yes
Decline of sales	yes	yes				
Decline of benefit	yes	yes	yes			
Influencing development plan			yes			
Taking more resources			yes			
System function problem				yes		
Law problem				yes		

As shown in the table above, the decline of the user number (I1, I3, I4, I5, I6) and benefit (I1, I2, I3) are mentioned most, the others just mentioned one or two times. Development teams highlight the decline of user number and benefit because they use them to directly evaluate their assumption management. The result does not mean that assumption failures have no influence on other aspects.

"In general, we publish the new version twice a month. We predict that the user number can increase to 1000 after releasing the new version, the direct outcome of the assumption failure is no increase of the user amount."

"If the assumption failure can not be solved appropriately, it will finally result in the loss of the profit."

5.4.6.2 Decline of use number

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-32. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
it can not bring serious problems. Maybe it just results in the decline of the user access, also leading to the decrease of product sales and benefit.	1
In other situation, like interface design, the product managers assume what design style could attract users' attention. In practice, it is highly likely that the users' scenarios are not what the designer expected, which lead to the users ignoring the functions.	3
Security assumption, like privacy protection, is different in different countries, if users think their privacy is violated, they will not use the application anymore.	
The millions of users could not access the services	4
Other assumptions are about the interface design. If the interface design can not meet the user habit, some of them feel it is not important and continue to use it, others refuse to use it.	
In contrast, if the assumption is invalid, the number of the user can not reach the amount expected.	5
In general, we publish the new version twice a month. We predict that the user number can increase to 1000 after releasing the new version, the direct outcome of the assumption failure is no increase of the user amount.	
For our product, the user amount is major direct criteria that evaluate our design. So, the user experience is the top priority in the design of front-end of a mobile application.	
If the user amount is not as expected.	6

5.4.6.3 Decline of sales

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-33. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
it can not bring serious problems. Maybe it just results in the decline of the user access, also leading to the decrease of product sales and benefit.	1
If it happens, it will cause the loss of sales and money.	2

5.4.6.4 Decline of benefit

The transcription is presented in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-34. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
it can not bring serious problems. Maybe it just results in the decline of the user access, also leading to the decrease of product sales and benefit.	1
If it happens, it will cause the loss of sales and money.	2
If the assumption failure can not be solved appropriately, it will finally result in the loss of the profit.	3

5.4.6.5 Influencing development plan

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-35. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
Besides, it also has the negative influence on the product development schedule.	3
which will have the influence on the development of the other functions because a function involves software test, software design and software maintenance.	
The increased attention of one assumption gives rise to the decreased attention of other assumptions.	

5.4.6.6 Taking more resources

The transcription is presented in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-36. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
The development team members also review the code (comparing the update between the two versions) to solve the problem, which takes much time and human resource, but necessary.	3

5.4.6.7 System function problem

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-37. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
After less than half a year, the marketing department held a promotion activity. The user concurrency sharply increases to 10000 and the system is down in a short time.	4
If designers do not consider the key assumption, it will have the negative influence on the logic of the service, data storage.	

5.4.6.8 Law problem

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-38. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
The millions of users could not access the services and complain to us.	4

5.4.7 Reasons that cause assumption failure

5.4.7.1 Synthesis

Table 5-39. data synthesis of RQ4.2

	I1	I2	I3	I4	I5	I6
Poor understanding of requirement		yes		yes		
Poor understanding of assumption	yes					
Poor understanding of risk of assumption					yes	

Poor understanding of market context					yes	
Unclear product orientation	yes					yes
Unclear product strategy	yes					
Poor ability	yes	yes	yes	yes	yes	yes
Poor experience	yes	yes	yes	yes	yes	yes
Low operation standardization	yes					
Poor software testing		yes	yes			
Not enough communication among team members		yes				

As shown in the table above, two major reasons that result in assumption problems are the poor ability (I1, I2, I3, I4, I5, I6) and experience of development team members (I1, I2, I3, I4, I5, I6). Experience is defined as the knowledge of the previous situation for a domain. Ability is used to analyze problems and solve problems. Interviewee 5 approves that improving the ability and experience of the development team members can not completely solve assumption-related problems. The interviewees provide many reasons that cause assumption failures. the reasons vary a lot according to the different realities of assumption management. However, the development team members that implement the assumption management is the root factor of assumption failures.

"The criteria that the memory management application judge whether the application should be deleted is set by the development team members. It is highly likely that the application is removed by mistake because the development team members can not dig all scenarios. After all, the ability to analyze problems and solve problems is not enough."

"At the beginning of the website design, we can only estimate the user concurrency based on our experience. After an assumption failure, we make some improvement."

5.4.7.2 Poor understanding of requirement

The transcription is listed in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-40. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
There are complex dependencies between components in our application. The most significant problem is that the development team members can not realize the relations between two modules. They always change one module with ignoring its influence on another module	2
It is hard for development team members to have the comprehensive understanding of all dependency modules.	
The reason for the assumption failure is that we do not realize the users have different habits.	4
A designer can not comprehensively understand the goal of the software at the early stage of designing,	

5.4.7.3 Poor understanding of assumption

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-41. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number

Sometimes, there are also some mistakes of assumption realizations. For example, the designers design that when users press "confirm" button, the client sends a request to the server. If the client sends a request to the server just when the users log in the client, the load of the server will be exceeded. The team estimates that the number of people who log in the client is lower than the number of people who press" confirm" button.	1

5.4.7.4 Poor understanding of risk of assumption

The transcription is shown in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-42. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
Not comprehensive considering of risks of requirements could result in the invalid assumption	5

5.4.7.5 Poor understanding of market context

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-43. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
Also, a good design has an intensive relation with the market context.	5

5.4.7.6 Unclear product orientation

The transcription is presented in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-44. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
I have no idea what kind of products that always change, which means that the company forms an unclear production orientation or product strategy.	1
the problem will mainly be the improper product orientation.	6

5.4.7.7 Unclear product strategy

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-45. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
I have no idea what kind of products that always change, which means that the company forms an unclear production orientation or product strategy.	1

5.4.7.8 Poor ability

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-46. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number

The companies always have poor development team members' ability. My understanding is likely to be not precise because I have worked for the only one company.	1
If the team is excellent enough, this kind of problems can not happen.	
I think the root cause is the ability of the development team members.	2
For example, we developed a memory management application, one of the functions is cleaning applications in the memory. It is the development team that establishes the protocol about which application should be deleted. It is highly likely that the application is removed by mistake because the development team members can not dig all scenarios. After all, the ability to analyze problems and solve problems is not enough.	3
Development team members' experience and ability are the root reason.	4
If the development team members' ability and experience are enough, the problems can be decreased, but not be completely solved.	5
A development team member with rich experience and high ability could handle assumption failures in time.	6

5.4.7.9 Poor experience

The transcription is presented in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-47. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
However, it is a mature procedure in the company. It is likely not to happen if the product manager is experienced.	1
The second reason is that the test case is not enough, which can be immigrated if the tester is experienced.	2
The invalid data assumption results in problems of connection between the new module and the original module. The problem is mainly due to the poor experience of the development team members and incomplete test case.	3
At the beginning of the website design, we can only estimate the user concurrency based on our experience. After an assumption failure, we make some improvement.	4
Development team members' experience and ability are the root reason.	
If the development team members' ability and experience are enough, the problems can be decreased, but not be completely solved.	5
A development team member with rich experience and high ability could handle assumption failures in time.	6

5.4.7.10 Low operation standardization

The transcription is listed in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-48. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
The problem depends on the standardization of the management of the product manager.	1

5.4.7.11 Poor software testing

The transcription is shown in the table below. Each sentence is the positive evidence to the synthesis.

Table 5-49. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
Not all of this kind of problems can be detected in the software testing.	2
The invalid data assumption results in problems of connection between the new module and the original module. The problem is mainly due to the poor experience of the development team members and incomplete test case.	3

5.4.7.12 Not enough communication among team members

The transcription is shown in the below table. Each sentence is the positive evidence to the synthesis.

Table 5-50. data analysis for RQ4 (italic: quoted from the answer of the interview)

Sentence	Interview number
The first reason is that not enough communication among the development team members of the different modules, especially in the complicated and large-scale software project.	2

6 DISCUSSION

6.1 Contribution

I constructed an assumption management model from the result of my interviews. The model can be used to control the assumption lifecycle (assumption being discovered, assumption changing and assumption failuring).

The first part of the model is assumption type and assumption formulation. User habit assumptions consist of name, description and value, while quality attribute assumptions are composed of name and value.

The second part of the model is the necessary situation that assumptions are supposed to be noticed, which could help development team members in avoiding losing significant assumptions. The assumption is discovered when it frequently changes and is necessary to requirements. This part corresponds to the first stage of the assumption lifecycle.

The third part of the model is a set of assumption treatment activities, which is useful in helping development teams with arranging assumption treatment activities. Development teams figure out the value of the assumption based on their investigation, historical data and experience. Assumptions are monitored by online tools. Assumptions are validated by development team members and development platforms (a few teams). It is the development team member that handles assumption failures. This part corresponds to the second stage of the assumption lifecycle.

The fourth part is the reasons and implications of assumption failures. Assumption failures result in the decline of the benefits and user number. The root reasons of assumption failures are poor ability and experience of development team members. This part corresponds to the third stage of the assumption lifecycle. The model separately provides the suggestions in the different stages of the assumption lifecycle to realize the goal of software development.

My model fits better into agile development than waterfall development. The model is to handle frequent assumption changes (as a situation that assumptions should be discovered), which is not consistent with the protocol of waterfall development. The assumptions in waterfall development barely change, so, it is not worthy implementing the assumption management model.

The model is created based on the view of product managers. However, it is implemented by all team members. Product managers discover key assumptions and figure out their value. They are responsible for connecting the customers when figuring out the value of assumptions and validating the value. They also make descriptions about how to handle assumption failures. Project managers control the process of the implementation of the model, especially when assumptions failing, which could result in the delay of the software development. Requirement developers also take part in figuring out the value of the environment assumptions. Architects and developers should have a correct understanding of the assumption. Reviewers (can be the customer, product manager, project manager, requirement developer and so on) validate assumptions. Maintainers (can be the suitable team members) handling the assumption failures. Vendors should be involved in the assumption validation and handling assumption failures.

Because of the validity threats discussed in the Validity Section, care must be taken when implementing the model. There are no more details of realizing the model, and the practical situation is too complicated. So, it is likely that the model has bad interaction with users' original assumption management practice.

I also explored the advantages and disadvantages of the two assumption management frameworks: assumption mapping is useful, taking too much effort and function being covered.

6.2 Restatement of the result

Assumption failures are the critical cause of software failure. The current assumption management practice is not efficient and effective, which is mainly due to that development teams wrongly handle assumptions. The previous research can not support assumption management practice because they created solutions without taking into account what the development teams expect to do with the assumption management. The formal assumption management framework and semi-formal assumption management framework have not been tested effective in practice. To solve the problem, I studied the current situation of assumption management practice, and advantages and disadvantages of the two frameworks, based on which I propose improvement chance of assumption management practice. Firstly, since today's assumption management can not reach what development teams expect, I need to know how do development teams handle assumptions. Next, by analyzing the factors of assumption failures and good assumption management practice, I could suggest how development teams improve the assumption management practice. Finally, I explored the advantages and disadvantages of frameworks that helps development teams judge whether to use the frameworks.

According to the research purpose, I answered the following research questions: the positive influence of the frameworks is helping analyze problems through the assumption mapping function. However, the functions of the frameworks are covered and too much effort needed to implement them. The main assumption types are assumptions about the user habit and assumptions about the quality attributes of the servers. The user habit assumption is composed of name, description and value. The quality attribute assumption consists of name and value. The situation that development teams discover assumptions is that assumptions frequently change and are necessary to the requirements. The main assumption treatment activities are figuring out the value of assumptions, monitoring assumptions, validating assumptions and handling assumption failures. The key implications of assumption failures are the decline of the user amount and company benefits. The main reasons for the assumption failures are the low ability and experience of development team members. The tool used in the assumption management is the website development platform. Assumption management is implemented by all team members. The aim of assumption management is maxing the benefits and user number.

6.3 Comparison of results with related work

Interviewees feel the assumption-component mapping function of the formal framework is useful in problem analysis. It uses an algorithm to map assumptions with software components. The interviewees also agree that the functions of the frameworks are covered by development team members and their existing tools. Besides, it takes too much effort to implement the frameworks (inserting assumption into source code and constructing platform). At the same time, the literature does not provide the empirical evaluation [8] [9].

Assumptions in our study are mainly environment assumptions. User habit assumptions and assumptions about the quality attribute are the most important. Assumptions are divided into three levels: requirement-level assumption, architecture-level assumption and code-level assumption [22]. All interviewees only mentioned requirement-level assumptions, which is due to two reasons: (1) they are all product managers, who are familiar with requirement assumptions, and (2) the outside requirement assumption changes always influence the software requirements [2]. The assumptions in the semi-formal framework [9] are classified according to their content. The assumption classification of the formal framework [8] is based on assumptions' validity, criticality and compositional scope. Two environment assumptions in my result are categorized according to the content and handled differently. So, the assumption category in the semi-formal framework should be divided to be more accurate. The assumption classification of the formal framework [8] is also not applicable. From the answers of the interviewees, many assumptions need to be handled but are not serious. Also, they evaluate benefits instead of criticality to decide how to handle the assumption.

Assumption formulations are studied in the assumption management frameworks in the previous research, which means that assumptions are systematically treated [8] [9] [34]. The result of my interview shows the different situation. Todays' software development teams do not treat assumptions independent of requirements. So the current assumption formulation just contains the name and value for quality attribute assumptions, and name, description and value of user habit assumptions. All assumption formulations are established to satisfy the need of software development.

The literature shows that an assumption is discovered when it frequently changes [2] [6] [8] [9] [10] [13], which is consistent with my result. In another situation, papers [8] [13] present that the assumptions about the dependencies among a lot of components need to be noticed, while only interviewee 2 mentions assumptions about component dependencies. Interviewees also approve that assumptions are discovered when they are necessary to the requirements. In some situations, development teams need to assume scenarios for each requirement if necessary.

As shown in our result, the major assumption treatment activities are figuring out the value of assumptions, monitoring assumptions, validating assumptions and handling assumption failures. From the result of my interview, I can define the four activities as Figuring out the value of assumption means exploring the value of an existing assumption. Monitoring assumptions suggests real-time collecting the practical value of the assumption. Validating assumption indicates comparing the real assumption value with value established to software. Handling assumption failures represents a set of activities taken to correctly deal with assumption faults. My interview result is not consistent with the statement of the literature [30], which proposes four main activities: identifying assumptions, recording assumptions, monitoring assumptions and handling assumption failures. There are no identifying assumptions and recording assumptions in my result because the development team of interviewees documents assumptions with requirements and they only pay attention to assumptions about servers and user habits. They value how to set the value of assumptions. That the previous work does not mention the assumption validation does not mean that assumption validations are not implemented. Many tools (containing the frameworks) are designed to support the assumption monitoring and assumption validation, while they are useless in figuring out the value of assumptions and handling assumption failures. The formal framework could help making decisions [8]. There are several tools used on architecture assumptions [23] [24] [25] [26], which have not been implemented in practice. Website development platform is used to support assumption management in internet companies. Assumption prioritization is useful in documenting assumptions [13]. According to my result, the activities vary according to the different assumption types. For the user habit assumptions, development teams could figure out the value through survey and discussion, and validate assumptions through the online feedback and survey. For the assumption about quality attributes of servers, they could do the calculation based on the history data and validate assumptions using data statistics. Also, the first type of assumptions could be validated before the application implementation. The backup value of assumptions can not be used in quality-related assumptions.

For the implication for the assumption failures, two serious accidents described in the literature cause the great loss for the user and company [5] [6], which barely happens in the current software development. The most serious implication is canceling the project. For the non-critical outcome, what the literature claim [3] [13] is the same as my interview result. Both interviewees and the previous works do not differentiate the assumption-related problems cross the assumption types. Assumption failures could be regarded as a driving force for software evolution [6]. User amount and benefits are directly evaluated in companies when making assumptions. I can reason out the following relations between the implications: when an assumption fails, its direct outcome is software function problems, leading to the decline of the user amount, which results in the reduction of sales and benefits. From the another aspect, to fix the problem, the company must take more resources and adjust the development plan. In other situations, if the requirement is set in the contract, assumption failures also result in the law problems.

The reasons that cause assumption failures are poor experience and no computer-related educational background of developers [31]. On the one hand, the authors use the working year to express experience, which is inconsistent with our survey. According to the answer of the interviewees, I define experience as the knowledge of the previous situation for a domain. On the other hand, the previous research figures out that the development team members with no computer-related educational background can not join in assumption-related works. Poor ability of development team members in managing assumptions refers to the ability to analyze problems and solve problems. Our survey presents the poor ability and experience of the development team members are the primary reason for assumption failures. Besides, the answers also show that the increase of the members' ability and experience can not completely solve assumption-related problems.

6.4 Implication for research

The research digs assumption management practice and improvement opportunities. It has the following implications for the future research: firstly, the development team member is the leading factor for the success of the assumption management. The future research can study deeper on how to improve assumption management practice by solving the problem of the poor ability and experience of development team members. Secondly, my research proposes a bunch of assumption management activities and methods without explicit definition and explanation. it is better to find how to implement the activities in more details and the factors that influence assumption management activities, such as software type, project type, ability of development team members. Thirdly, the solutions in the previous work (like solutions described in the related work) could use some adjustments based on the result of my research to fit into the practical software development.

6.5 Implication for practice

Firstly, product managers and requirement developers should prepare backup values for user habit assumptions. They could prepare more than one designs of the interface and function in the software design phase and implement them when the first option does not work. Secondly, documenting assumptions with requirements if assumptions are not too many, which could improve the efficiency of assumption management. Thirdly, it is efficient for development teams to use platform or online data statistics tool to manage assumptions. Fourthly, project managers need to consider the developer factors. They should Assign experienced and high-ability developers to assumption management or train members before starting the software project. Fifthly, project managers could not always change team members to different domain software projects because they are likely not familiar with new software domains and partners. Sixthly, it is better to use assumption management frameworks if the original assumption management practice is not as expected, and the resources and state of the company support starting a new framework. Seventhly, the formal assumption management framework can be used by the development teams to analyze assumption failures.

These recommendations can be used in handling requirement assumptions. The lessons do not cover all parts of the assumption management. I can not give more details to use the lessons. The practitioners could adjust them based on their realities.

7 CONCLUSION

I implemented a qualitative interview research to reach two research goals: how do development teams handle assumptions and how do they improve the assumption management. The study was conducted with six interviewees from five companies.

I have found that the assumption mapping function of the formal assumption management framework is helpful in analyzing problems. Besides, their functions could be covered and too much effort needed to implement the frameworks. The main assumption types are assumptions about user habits and quality attributes for servers. User habit assumptions are composed of name, description and value, while quality attribute assumptions consist of name and value. The situations that assumptions are discovered are assumption frequently changing and being necessary to requirements. The main activities of the assumption treatment are figuring out the value of assumptions, monitoring assumptions, validating assumptions and handling assumption failures. The implications of assumption failures are the loss of users and benefits. The reasons for assumption failures are the poor ability and experience of development team members.

The significance of our research is digging the assumption management current situation and improvement chance. Previous solutions do not take into account the need of practical software development. I propose seven lessons to help improve the efficiency and effectiveness of assumption management practice based on the interview result. Besides, the paper can be treated as a starting point of assumption management research.

REFERENCES

[1] K. E. Emam and A. G. Koru, “A Replicated Survey of IT Software Project Failures,” IEEE Software, vol. 25, no. 5, pp. 84–90, 2008.
[2] D. L. Parnas, “Information distribution aspects of design methodology,” IFIP Congress, vol. 4, no. 5, pp. 339–344, 1971.
[3] N. Madhavji, A. Miranskyy, M. Davison, and M. Reesor, “Modelling Assumptions and Requirements in the Context of Project Risk,” in 13th IEEE International Requirements Engineering Conference (RE), Los Alamitos, CA, USA, 2005, pp. 471–472.
[4] H.-F. Hsieh and S. E. Shannon, “Three Approaches to Qualitative Content Analysis,” Qualitative Health Research, vol. 15, no. 9, pp. 1277–1288, 2005.
[5] S. Dalal, R. S. Chhillar, "Case studies of most common and severe types of software system failure", International Journal of Advanced Research in Computer Science and Softare Engineering, vol. 2, no. 8, pp. 341-347, 2012.
[6] Lehman, Meir M. "The role and impact of assumptions in software development, maintenance and evolution.", in IEEE International Workshop on Software Evolvability, Budapest, Hungary, 2005, pp. 3-14.
[7] M. Savin-Baden and C. H. Major, “Qualitative research: the essential guide to theory and practice,” London: Routledge, 2013.
[8] A. S. Tirumala, “An Assumptions Management Framework for Systems Software,” University of Illinois at Urbana-Champaign, Champaign, IL, USA, 2006.
[9] G. A. Lewis, T. Mahatham, and L. Wrage, “Assumptions Management in Software Development,” Carnegie Mellon University at Pittsburghers, Pennsylvania, USA, Aug. 2004.
[10] M. M. Lehman, “Uncertainty in computer application is certain-software engineering as a control,” in COMPEURO’90: Proceedings of the 1990 IEEE International Conference on Computer Systems and Software Engineering - Systems Engineering Aspects of Complex Computerized Systems, Tel-Aviv, Israel, 1990, pp. 468–474.
[11] C. B. Haley, R. C. Laney, J. D. Moffett, and B. Nuseibeh, “Using trust assumptions with security requirements,” Requirements Engineering, vol. 11, no. 2, pp. 138–151, 2006.
[12] K. Kelley, B. Clark, V. Brown, and J. Sitzia, “Good practice in the conduct and reporting of survey research,” International Journal for Quality in Health Care Journal of the International Society for Quality in Health Care, vol. 15, no. 3, pp. 261–266, 2003.
[13] M. A. Al-Mamum and J. Hansson, “Review and Challenges of Assumptions in Software Development,” in Proc. of the Second Analytic Virtual Integration of Cyber-Physical Systems Workshop, Vienna, Austria, 2011, pp. 53–60.
[14] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, “Selecting Empirical Methods for Software Engineering Research,” in Guide to Advanced Empirical Software Engineering, F. Shull, J. Singer and D. I. K. Sjoberg, Eds, Springer London, 2008, pp. 285–311.
[15] G. Geracie, “Take Charge Product Management: Time-tested Tips, Tactics, and Tools for the New or Improved Product Manager,” 2nd ed. Chicago, IL: Actuation Press, 2016.
[16] H. Ding and L. Sha, “Dependency algebra: A tool for designing robust real-time systems," in IEEE Real-Time Systems Symposium (RTSS) Conference, Miami, FL, USA, 2005, p. 11 pp-220.
[22] C. Yang, P. Liang, and P. Avgeriou, “A survey on software architectural assumptions,” Journal of Systems and Software, vol. 113, pp. 362–380, 2016.
[23] D. Garlan, R. Allen, and J. Ockerbloom, “Architectural mismatch: Why reuse is so hard,” IEEE Software, vol. 12, no. 6, pp. 17–26, 1995.
[24] L. R. Cai, J. S. Bradbury, and J. Dingel, “Discovering Architectural Mismatch in Distributed Event-based Systems using Software Model Checking,” Queen’s University at Kingston, Ontario, Canada. 2006.
[25] L. R. De, C. Gacek, and A. Romanovsky, “Architectural Mismatch Tolerance,” *Architecting Dependable Systems*, vol. 2677, pp. 175–194, 2003.
[26] S. Uchitel and D. Yankelevich, “Enhancing architectural mismatch detection with assumptions,” in 7th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems(ECBS), Edinburgh, Scotland, 2000, pp. 138-146.

[27] M. M. Lehman and J. F. Ramil, “Rules and Tools for Software Evolution Planning and Management,” Annals of Software Engineering, vol. 11, no. 1, pp. 15–44, Nov. 2001.
[28] R. Roeller, P. Lago, and H. van Vliet, “Recovering architectural assumptions,” Journal of Systems and Software, vol. 79, no. 4, pp. 552–573, 2006.
[30] I. Ostacchini and M. Wermelinger, “Managing assumptions during agile development,” in 2009 ICSE Workshop on Sharing and Reusing Architectural Knowledge, Vancouver, British Columbia, Canada, 2009, pp. 9–16.
[31] Ö. Albayrak, H. Kurtoglu, and M. Biçakçi, “Incomplete Software Requirements and Assumptions Made by Software Engineers,” in 16th Asia-Pacific Software Engineering Conference, Batu Ferringhi, Penang, Malaysia, 2009, pp. 333–339.
[32] R. Ali, F. Dalpiaz, P. Giorgini, and V. E. S. Souza, “Requirements Evolution: From Assumptions to Reality.,” in the 16th International Conference on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD 11), London, United Kingdom, 2011, pp. 372–382.
[33] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, “Tropos: An Agent-Oriented Software Development Methodology,” Autonomous Agents and Multi-Agent Systems, vol. 8, no. 3, pp. 203–236, 2004.
[34] C. Yang and P. Liang, “Identifying and Recording Software Architectural Assumptions in Agile Development,” in 26th International Conference on Software Engineering and Knowledge Engineering, Hyatt Regency, Vancouver, Canada, 2014, pp. 308-313.
[35] X. Wang, J. Mylopoulos, G. Guizzardi, and N. Guarino, “How software changes the world: The role of assumptions,” in 10th IEEE International Conference on Research Challenges in Information Science (RCIS), Grenoble, France, 2016, pp. 1–12.
[36] Burge, Janet E., and David C. Brown. "Rationale-based support for software maintenance." in Rationale management in software engineering, A. H. Dutoit, R. McCall, I, Mistrik, B. Paech, Eds, Springer Berlin Heidelberg, 2006, pp. 273-296.
[37] T. Gorschek, P. Garre, S. Larsson, and C. Wohlin, “A Model for Technology Transfer in Practice,” IEEE Software, vol. 23, no. 6, pp. 88–95, 2006.
[38] M. R. McBride, "The software architect," Communication of the ACM, vol. 50, no. 5, pp. 75-81, 2007.
[39] A. van Lamsweerde, “Requirements engineering in the year 00: a research perspective,” in 22nd International Conference on Software Engineering (ICSE), Limerick, Ireland, 2000, pp. 5-19.
[40] S. Fickas and M. S. Feather, “Requirements monitoring in dynamic environments,” in 2nd IEEE International Symposium Conference on Requirement Engineering (ICRE), York, UK,1995, pp. 140-147.
[41] A. Miranskyy, N. Madhavji, M. Davison, and M. Reesor, “Modelling assumptions and requirements in the context of project risk,” in 13th IEEE International Conference on Requirements Engineering (ICRE), Paris, France, 2005, pp. 471-472.
[42] S. Uchitel and D. Yankelevich, “Enhancing architectural mismatch detection with assumptions,” in 7th IEEE International Conference and Workshop on Engineering of Computer Based Systems(ECBS), Edinburgh, Scotland, 2000, pp. 138-146.
[43] P. Kruchten, P. Lago, and H. Vliet, “Building Up and Reasoning About Architectural Knowledge,” in International Conference on Quality of Software Architectures, Vasteras, Sweden, 2006, vol. 4214, pp. 43-58.
[44] S. A. Fricker, “Software Product Management,” in Software for People, A. Maedche, A. Botzenhardt, and L. Neer, Eds, Springer Berlin Heidelberg, 2012, pp. 53–81.
[45] K. Pohl and C. Rupp, Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant, 1 edition. Santa Barbara, CA: Sebastopol, CA: Rocky Nook, 2011.
[46] Z. Tang, M. Yang, J. Xiang, and J. Liu, “The Future of Chinese Software Development,” IEEE Software, vol. 33, no. 1, pp. 40–44, 2016.
[47] Y. Huang, “Understanding the software industry in China: Export performance and regional development,” Journal of Emerging Knowledge on Emerging Markets, vol. 3, no. 1, pp. 16, 2011.

[48] M. Spiegel, P. F. Reynolds, and D. C. Brogan, “A case study of model context for simulation composability and reusability,” in 37th Conference on Winter simulation, 2005, vol. 2, no, 4, pp. 437-444.
[49] J. A. Dewar, C. H. Builder, W. M. Hix, and M. H. Levin, “Assumption-Based Planning; A Planning Tool for Very Uncertain Times,” Rand Corporation at Santa Monica, CA, Jan. 1993.
[50] T. Dingsøyr and H. Vliet, “Introduction to Software Architecture and Knowledge Management,” in Software Architecture Knowledge Management, M. Ali Babar, T. Dingsøyr, P. Lago, and H. Vliet, Eds, Springer Berlin Heidelberg, 2009, pp. 1-17.
[51] J. Tyree and A. Akerman, “Architecture decisions: demystifying architecture,” IEEE Software, vol. 22, no. 2, pp. 19- 27, 2005.
[52] S, A. Fricker, T. Gorschek, C. Byman, A. Schmidle, “Handshaking with Implementation Proposals: Negotiating Requirements Understanding,” IEEE Software, vol. 27, no. 2, pp. 72-80, 2010.

APPENDIX A

Introduction of the frameworks in interviews

Table 0-1. introduction of the assumption management frameworks

	Semi-formal framework	Formal framework
Assumption extraction	The two frameworks support extracting the assumptions from the source code and development team members directly inputting assumptions.	
Assumption type	Control assumption (the sequence that methods being invoked), environment assumption (execution environment of software), data assumption (the format of inputting and outputting data), usage assumption (how the software is used) and convention assumption (the standard that developers follow).	Three dimensions: time-frame of validity, criticality and compositional scope. Time-frame of validity: Static assumptions do not change before software life end; system configuration assumptions do not change when software execution. Dynamic assumptions could change all time. Criticality: critical assumptions cause serious software problems if they fail; non-critical assumptions failures cause performance degradation or non-core functions compromise. Compositional scope: public assumptions need to be met by external components; private assumptions are just met by internal the component.
Assumption searching	Development team members with right authorization could search assumptions.	
Assumption validation	Notifying the team members when the assumption being created in the framework.	Supporting establishing the schedule about the time that assumptions should be validated, based on which the team member is notified.
Assumption mapping	no	Mapping of assumptions and components help to analyze assumption failures.

APPENDIX B

Transcription of interview 1

For our software, if user concurrency sharply rises in a short time, the maintainers could solve this problem by increasing the number of the server in a few minutes, which is planned before implementing the product.

In practice, the company would not Deliberately arrange the assumption treatments according to its possibility to change, though it influences the activity arrangement. The companies gradually form a solution for each assumption based on their practical problems themselves.

Sometimes, there are also some mistakes of assumption realizations. For example, the designers design that when users press "confirm" button, the client sends a request to the server. If the client sends a request to the server just when the users log in the client, the load of the server will be exceeded. The team estimates that the number of people who log in the client is lower than the number of people who press" confirm" button. Their assumption management is not a systematic procedure to implement. In the daily development practice, assumptions are just a factor that needed to be considered when designing software. If I am responsible for the service that the software provides, assumptions about target users, execution environment, servers are not mentioned in our document because these factors are not changed in our software project. So, assumption management is missing in practice. Assumption-related activities are important in software design that development team members need to do investigations about the software environment, like target user, execution environment, usage environment and so on, being recorded in the software design document, which tends not to be achieved systematically.

As I understand, the Chinese software companies do not manage assumptions systematically, which is mostly due to that it is not worth doing this. For example, some knowledge, like using AliCloud in our software and Java to code, is very fundamental for our software, which can not be changed.

In other situations, the factors, like user concurrency, is changing every day, which deserves being studied because different software or the same software in different time need different environments. To peruse interests, Some Chinese companies just add services with no change of software environment.

Our product solutions could meet the need of the most of the users after we balance the software features for different users.

There is no serious problem happens in the internet company because we implement 24-hour stand by. Maybe there are some problems solved in a short time that I do not know.

I am very confident that we have not encountered the problem of the change of the target user. Setting target user is the first step of requirement analysis. The target user changes are largely because the project leader approves that other users could bring more profits. For to-Business, if the customer change, it results in a new project.

I have no idea what kind of products that always change, which means that the company forms an unclear production orientation or product strategy. The companies always have poor development team members' ability. My understanding is likely to be not precise because I have worked for the only one company.

I can hold an example, when we decide whether to use the round buttons or the square buttons in the user interface, we need to make the evaluation. If we approve that the round button can attract users' attention and the product manager does not do online verification (the data shows whether there are more users click the button after it becomes round), it can not bring serious problems. Maybe it just results in the decline of the user access, also leading to the decrease of product sales and benefit. The problem depends on the standardization of the management of the product manager. However, it is a mature procedure in the company. It is likely not to happen if the product manager is experienced.

I do not think a systematic framework is needed in practice if the framework is hardly realized. Besides, assumption management is an essential skill of product manager.

Clearly, the frameworks support tracing assumptions and provide constraints on each step. However, the two functions are not applicable in practice. For tracing assumptions, the product manager could document assumptions unsystematically with few problems happened. We can not predict the loss, but it takes too much time to implement a new framework. Our team has the clear division of the responsibility. So, there is no serious problem happened because of the assumption. The only one reason that forcing a company to pay attention is that it is the top priority problem. It is clear that assumption-related problem is not significant in our software development. If the team is excellent enough, this kind of problems can not happen.

In our software project, the development of the client module takes about 100-200 development team members.

We have encountered some problems with assumptions of components. I have worked in different teams. Some of them have many this kind of problems and others do not experience the problems. So, it depends on the team ability. In practice, there is a verification phase, if an attribute of a component can not meet the assumptions, the component will not pass. My company does not categorize the assumptions as the frameworks describe.

The another reason that the frameworks are not applicable is that the existing project management practice can cover their functions, like software testing, software verification. There are few risks that could result in software problems. In this situation, the company always follow the cost-benefits ratio.

For the concurrency assumption, we only record its name and value. The structure of the user habit assumption is more complex, containing the name, description, value and image. Using images help improve the efficiency of the assumption validation.

Transcription of interview 2

I am working on an online shopping website. The application has run for a long time, so, some outside assumptions, like server load, response time or target user, can be estimated accurately. For example, when we consider the system load for the next version of the application, we would make a certain percentage growth of the system load based on the original system architecture and use hardware to support it. We also monitor the real-time load during the system being operated. The growth percentage is made by comparing the system load in recent years, which could guarantee the system can run and not too much waste of the resource.

Large software companies have a relatively accurate judgment of the outside assumption of the application.

There are complex dependencies between components in our application. The most significant problem is that the development team members can not realize the relations between two modules. They always change one module with ignoring its influence on another module. For example, one of the requirements is that the global address format of the receiver is configurable. Allowing for there is the various address formats in the various countries, like the United States is divided into states and China is divided into Provinces, the address in registration must be configured according to the different countries. At the same time, another module, like payment check is also needed to be updated to be consistent with the registration module. If it is not found, the function can not be used. Not all of this kind of problems can be detected in the software testing. If it happens, it will cause the loss of sales and money. The first reason is that not enough communication among the development team members of the different modules, especially in the complicated and large-scale software project. It is hard for development team members to have the comprehensive understanding of all dependency modules. The second reason is that the test case is not enough, which can be immigrated if the tester is experienced. I think the root cause is the ability of the development team members.

For the frameworks, I think assumption-component mapping in the formal assumption framework is significantly helpful. The two frameworks integrated with the powerful software computing ability should be useful in assumption management practice. Assumption classification in the two frameworks is not what we categorize of our assumptions. So, the pre-

definition of the classification can not cover the scenarios of all companies and it should be configurable. The cost-benefit is always a root factor that drives a company to use a new framework. In practice, before using a new framework, we need to analyze its performance, trial it and make a decision.

For the load assumption of servers, we store its name, description, value, and version. The version is needed because we expect to predict the value of the new version software by calculating the trend of the past version. For the dependency assumptions, we document their name, description, value and component-relation mapping.

Transcription of interview 3

If we plan to develop a new function, we tend to do some pre-investigations and assume the technical support, resource, middleware or interface that concern the function. We take these assumptions as the basis of the new function. If the outside technology or interface is needed to be updated, they would inform us in advance. We can adjust the corresponding module to fit the update of the outside technology. If the assumption failure can not be solved appropriately, it will finally result in the loss of the profit. Besides, it also has the negative influence on the product development schedule. If there are some problems in the new module and the development team approves it could bring profits, they will assign time and resource to solve the problem. It will have the influence on the development of the other functions because a function involves software test, software design and software maintenance. In other situation, like interface design, the product managers assume what design style could attract users' attention. In practice, it is highly likely that the users' scenarios are not what the designer expected, which lead to the users ignoring the functions. The problem also impacts the user access. Security assumption, like privacy protection, is different in different countries, if users think their privacy is violated, they will not use the application anymore.

In our company, there is a data team that specializes the data of user behavior, like time on site, user access. Next, they can compare the user behavior data before and after the assumption being realized to validate the assumption. If there is a sharp change after the assumption is realized, we can judge the assumption state. In our company, most of the decision making is based on data statistics. For example, we assume that it is database problem that influences response time, leading to the decline of the user access. If we want to find the reason that results in user access decrease, we can analyze the application state data. The data analysis platform could review the execution state of each module, if there is a large number of the data of users can not be accessed normally, there must be problems in the database. The platform also could figure out the problem by comparing the current data with historical data. The development team analyzes the problems based on the data statistics. Every sub-teams of the module also could analyze the problems themselves.

In our company, whether development team members take resource on an assumption depends on whether it could bring profits. The increased attention of one assumption gives rise to the decreased attention of other assumptions. The decision is made by the development team. Two factors that mainly influence the development team making the decision: company profit and user need. The development teams peruse the best balance between the two factors.

I heard about that there are teams that cancel the project because of the assumption failure, but it rarely occurs. This kind of problems has not raised serious problems. For example, we developed a memory management application, one of the functions is cleaning applications. It is the development team that establishes the protocol about which application should be deleted. It is highly likely that the application is removed by mistake because the development team members can not dig all scenarios. After all, the ability to analyze problems and solve problems is not enough. The invalid data assumption results in problems of connection between the new module and the original module. The problem is mainly due to the poor experience of the development team members and incomplete test case. The development team members also review the code (comparing the update between the two versions) to solve the problem, which takes much time and human resource, but necessary.

For the classification of the assumption, I think the companies are inclined to divide the assumptions by their content. However, we do not categorize the assumptions as described in the semi-formal assumption management framework. The assumption-component mapping in the formal framework could help to find out the problem of the software. To realize the frameworks, the development team should create the assumptions, establish the execution environment of the framework, which can not be achieved if the problem is an emergency or the development cycle is short. The frameworks should be helpful in detecting the problems in time. If this strength could be proved in practice, the development team will use it.

For user habit assumptions, we simply need its name, description and value, while the name, description, value and vendor are required for outside technology assumptions because vendors could provide the technology according to our need and notify us if the technology is changed.

Transcription of interview 4

I am mainly responsible for website development. At the early time of the website design, user concurrency is a key attribute that I need to consider. The user concurrency has an intensive relation with architecture design. When I know an approximate amount of the user and approximate increment of the user (history data), based on which we can set how many users that the system needs to support. Our website provides service for the users from 120 countries. At the early stage of we assuming the user concurrency, the amount of the user is about 60000 in 120 countries, so, we assume the user concurrency is 5000. Since 120 countries are in different time zones and the time of the website implementation is short, we think there are limited users on the website. After less than half a year, the marketing department held a promotion activity. The user concurrency sharply increases to 10000, and the system is down in a short time. We need to make the adjustment for the user concurrency. Many factors that influence the user concurrency. At the beginning of the website design, we can only estimate the user concurrency based on our experience. After an assumption failure, we make some improvement. For example, we use a matrix to adjust the factors and manage the user concurrency. The matrix has not been mature. It can just help us to predict an approximate value of assumptions only. The main factors are the events that our company will hold, the amount of the user and so on. Since we do not know whether the value is accurate, we need to monitor the factors matrix. If the fluctuation the value is too sharp, the development team member will modify the corresponding parameter. We record the value adjusted. If the same event is held, we take the value as a reference. We use a website development platform for developing a website. The platform supports adjusting the website execution parameters to guarantee the normal running of the website. The platform is needed to be monitored continuously by development team members, which can not be achieved in practice. So, we set the value that slightly higher than the value estimated.

The assumption about the user operation habits is the another key assumption in our website development. Our target user is Foreigners, while the designer is Chinese. We need to design workflow, for example, register workflow (containing "back" button, "continue" button and "cancel" button). We put the buttons on the right side of the page and put the questions on the left side because we assume it is convenient for the users. However, the users in different countries have different habits, some of the users want that the buttons and questions are on the same side. We, finally, highlight the color of the button to meet the need of all users. The reason for the assumption failure is that we do not realize that the users have different habits. So, at the early phase of the website design, we make the assumptions based on our experience and monitor the assumptions by online monitoring and user feedback. We expect to set assumptions to meet the most of the users and prepare the backup plan for the special case. It is the user that let us know they want the questions and buttons are on the same side. A designer can not comprehensively understand the goal of the software at the early stage of designing, which is likely to result in the system down. In other situation, if the system could support too many scenarios, it must reduce the performance of the website. We must *consider the resource, technical support, time that realize the assumption, and benefits brought from the assumptions.*

If an assumption has not been changed for a long time, I will not pay attention to it. We set a critical value of the assumption. If the assumption reaches the critical value, the platform will notify the development team members, who could adjust the value of the assumptions in a short time.

If the system has some problems, we need to invite the related personnel to analyze the problems. We could review the website state from the log file, and compare the website state with parameter we set, based on which, we could implement an emergency treatment for the problem. We do not always adjust the value we set, because it can cause a series of effects on the website.

The assumption failure, like user concurrency, results in the system down. The millions of users could not access the services and complain to us. Other are assumptions about the interface design. If the interface design can not meet the user habit, some of them feel it is not important and continue to use it, while others refuse to use it. If designers do not consider the key assumption, it will have the negative influence on the logic of the service, data storage.

Poor understanding of the requirement always causes assumption problems. The backup plan to deal with assumption failure is also important in website execution. Sometimes, the assumption can not be completely realized. Development team members' experience and ability are the root reason. For example, we are not aware of that users in different countries have different habits.

Our platform has enough ability to support website development. It is not worth using a new framework since there are no serious problems happened. Implementing the frameworks take too much effort and resources. Also, I think our assumption problems are not due to improper management of a larger number of assumptions. However, the assumption mapping function is useful when making decisions. The assumption classifications of the frameworks are not flexible, and only the criticality is reasonable.

We store the name and value with the corresponding requirement for all type assumptions.

Transcription of interview 5

I have worked on mobile front-end development. During the requirement analysis phase, we need to assume the scenario that users use the application by using "5W3H" analysis method. We always make assumptions based on our experience and suggestions from other designers. Next, categorizing all ideas and testing their validity with users before application execution. In our products, assumption failures barely happen. For example, in registration module, we add an extended information in the final step of the registration process to guide the users to share the application. However, the users do not operate it as expected. So we need to change the function with the backup function. We always prepare backup functions in the design phase. Our final goal is increasing the number of the user. In contrast, if the assumption is invalid, the number of the user can not reach the amount expected. In general, we publish the new version twice a month. We predict that the user number can increase to 1000 after releasing the new version, the direct outcome of the assumption failure is no increase of the user amount. We have an online tool to monitor the state of the application, which could present the change of the users. We could make the decision based on that information. When we make the several assumption plans, we also analyze the risks for each of them. Through the risk analysis, we decide which assumption plan will be used.

There is also an example of the mobile game recommendation. We tend to push more than one game at one time. If we decide to push eight games, we will push four top games and four games that we expect the users to download. The four quotas are assigned to the advertisement. The four top games are the same category as the advertisement games are. We assume the top games could attract users' attention and promote the users to download advertisement games. Apparently, the download rate is not increased if the assumption is invalid. We, firstly, analyze the the risk points (like game's icon, game's introduce information and context that user selects the games). Next, we use the backup design and test it. We analyze the problem based *on our experience. In general, we predict risks and prioritize them before designing the application functions. We could avoid the non-core risks by doing some pre-investigations and set a backup design to solve core risks.*

For our product, the user amount is major direct criteria that evaluate our design. So, the user experience is the top priority in the design of front-end of a mobile application. I think the functions created based on using context. So, if the assumption is important, we will realize it.

We have many of games in the library. The top game is selected according to the download history of the user. The benefit is the key factor in choosing the advertisement games. The information considered to make a decision is generated from the data statistics. The data statistics is achieved by software.

The assumption failures could not influence software publication plan. Not comprehensive considering of risks of requirements could result in the invalid assumption. Also, a good design has an intensive relation with the market context. If the development team members' ability and experience are enough, the problems can be decreased, but not be completely solved.

For assumption classification, we do not intentionally consider them. We handle assumptions based on requirements.

The formal framework is more helpful in improving the efficiency of assumption management. The mapping function could provide additional information to help make the decision.

I consider semi-structured framework as an information management system.

We store not only the name, description and value, but the also risk point because we need change the assumption value by analyzing the risk point of the assumption failures.

Transcription of interview 6

In software design, we always need to make function-related assumptions. These kind of assumptions are mainly about how users feel about the function and whether they are willing to use the functions. We found a situation that people go from the home to the subway station. If the distance between the home and the subway station is about the 2-3 kilometer, it is too far to walk. At the same time, it is not worth taking a taxi. So, it is convenient to use a bicycle. However, customers can not bring the bicycle on the subway. So, these people need two bicycles, which is not practical. Because the cost of buying and maintaining two bicycles is too high for people that need to go to work by subway. We assume that there is a kind of bicycle that can take users from the home to the subway station and from the subway station to the workplace, and does not require maintenance from the users. We need to investigate whether there are many people have this kind of problems. If the number of the people is large enough, we will consider solving the problem from the software perspective, like developing an application to manage the bicycles. Here we could validate the assumption after designing the software.

The assumption of user habit concerns the user interface design. One of the reasons that the users do not use a function is that they are not aware of the function. So we will add the highlight or dynamic effect on the function. We also need to validate the change through the online user feedback.

The online tool could help to monitor the user feedback. If the user amount is not as expected, the problem will mainly be the improper product orientation. We would break down the solution into small components and test them one by one. We will select to modify the function or delete the function. The analysis depends on our experience and data of the statistics.

For an assumption, if it has an intensive relation with a requirement (like the interface design), we are willing to pay attention to it.

I think the frameworks are not applicable. The notification of the assumption validation is not needed because assumption number is not large. A development team member with rich experience and high ability could handle assumption failures in time. Also, software test could help to decrease the assumption-related problems. The criticality of the assumption is a factor *that influences we arranging assumption treatment activities. Maybe the assumption mapping function could help to analyze the reason for the problem.*

For the user context assumption formulation, we record assumptions' name, description, value and corresponding requirement. We store the corresponding requirement because they assume the user context according to the purpose of the software before analyzing the requirement in details. For the user habit assumption for the interface, we mainly record its name, description, value, and image to be convenient.