THE FACTORS THAT INFLUENCE, AND MECHANISM OF, ON-BOARD PASSENGER INFORMATION SYSTEM SOFTWARE FAILURE IN THE CHINESE RAIL TRANSIT SYSTEM

A thesis submitted to the University of Manchester for the degree of

Doctor of Business Administration

in the Faculty of Humanities

2021

LIANDONG GUO

ALLIANCE MANCHESTER BUSINESS SCHOOL/FACULTY OF HUMANITIES

Table of Contents

Word Count of the Thesis: 58,298

List of Tables

List of Figures

List of Abbreviations

AdaPIT: Adaptive and Pre-emptive IT

AERA: American Educational Research Association

AHP: Analytic Hierarchy Process

AI: Artificial Intelligence

APA: American Psychological Association

ATC: Automatic Train Control

AVE: Average Variance Extracted

BPR: Business Process Reengineering

CBTC: Communications-Based Train Control

CFA: Confirmatory Factor Analysis

CFF: Critical Failure Factor

CFI: Comparative Fit Index

CIO: Chief Information Official

CITC: Corrected Item–Total Correlation

CMMI: Capability Maturity Model Integration

CR: Composite Reliability

CRH: China High-Speed Railway

CRRC: China Railway Rolling Stock Corporation

CSC: Critical Success Chain

CSF: Critical Success Factor

DEMATEL: Decision-Making Trial and Evaluation Laboratory

DOC: Documentation Quality Problems

DPIS: Digital Passenger Information System

EFA: Exploratory Factor Analysis

EMU: Electric Multiple Units

FAI: First Article Inspection

FCM: Fuzzy Cognitive Map

GLS: Generalized Least Square

HWD: Hardware Development

IoT: Internet of Things

I-P-O: Input–Process–Output

IRIS: International Railway Industry Standard

IS: Information System

ISD: Information System Development

ISM: Interpretative Structural Modelling

IT: Information Technology

ITS: Intelligent Transportation System

ITTC: IT systems for Transport Companies

KMO: Kaiser–Meyer–Olkin

LCC: Life Cycle Cost

LRV: Light Rail Vehicle

ML: Maximum Likelihood

NCME: National Council on Measurement in Educational

NCR: Non-Conformity Report

PGM: Programming Quality Problem

PIS: Passenger Information System

PLS: Partial Least Square

QC: Quality Control

QoS: Quality of Service

RAMS: Reliability, Availability, Maintainability, Safety

RMR: Root Mean Square Residual

RTPI: Real-Time Passenger Information

SDLC: Software Development Life Cycle

SEI: Software Engineering Institute

SEM: Structural Equation Modelling

SMS: Short Message Service

SWD: Software Development

TMS: Top Management Support

TUTPIS: Tampere University of Technology Passenger Information System

URT: Urban Rail Transit

Abstract

Objective: The paper aims to explore the factors that cause the software failure of on-board PISs in China' rail transit and to determine how these factors affect the software outcome to construct a mechanism of influencing factors on software failure based on which empirical improvement solutions can be recommended.

Method: The study carries out mixed-method research by combining quantitative analysis and qualitative analysis. First the semi-structured interview contents are inductively processed with the aid of Nvivo to code the information and to identify the factors influencing software failure. The literature review and an industry expert questionnaire help to construct the influencing factor system, and then the DEMATEL method is used for quantitative analysis of the causal relationships between different factors. Finally, the questionnaire survey and structural equation analysis quantitatively define the action path of the factors that influence PIS software failure.

Result: The factors that influence, and the mechanism of, PIS software failure are studied from the perspective of external market factors and internal organizational factors. The results show that an uncertain external environment, inadequate project management and insufficient software testing are the direct factors affecting the failure of PIS software; customer requirement defects indirectly affect the failure of PIS software through employees' negative characteristics, inadequate project management and insufficient software testing; and employees' negative characteristic affects the failure of PIS software through inadequate project management and insufficient software testing.

Contributions: This research makes a threefold contribution. First, the study identifies the mechanism and action pathways that influence software failure. Second, this thesis puts forward a research model/framework based on the existing literature review and semi-structured interviews. Third, the process of the scale development, the design and selection of variables in the research also provide a good reference for future studies.

Keywords: on-board PIS, software failure, influencing factors, risk management, risk factors, rail transit

Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification from this or any other university or other institute of learning.

Copyright Statement

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given the University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or in electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements that the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain copyright, patents, designs, trademarks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and my be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialization of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see

http://documents.manchester.ac.uk//DocuInfo.aspx?DocID=24420), in any relevant thesis restriction declarations deposited in the University Library, in the University Library's regulations (see http://www.library.manchester.ac.uk/about/regulations/) and in the University's policy on the presentation of theses.

Dedication

The thesis is dedicated to my daughter Suttie and my son Bowen. The journey to the summit of my academic research took a lot of time away from them, and they always showed their understanding. I would also like to share my experience with them and light a lamp for their progress.

Acknowledgements

It is my great honour to have had Professor Matthew Allen and Professor Laszlo Czaban as my supervisors in Manchester, guiding me through my DBA research journey of over 6 years. Their professional instruction and rigorous academic attitude impressed me immensely, and they taught me not only how to perform academic study but also how to bridge the gap between Chinese and Western culture and education to transform my thinking. I would also like to thank Professor Jiangwei, my co-supervisor from SJTU, for his continuous support and assistance, especially in mid-year reviews and annual reviews. My special thanks and appreciation are due to Professor Shilong Ge, Professor Liangjian, Professor Suyong, Professor Wangquan, Professor Yongtai Chen and Professor Mingwei Tang, among others, all of whom gave me suggestions and assistance during my research process.

I would also like to thank my colleagues and friends who always provided support with great enthusiasm whenever I needed help, especially during the processes of conducting empirical interviews and questionnaire surveys.

I would also love to share my success with my family. Without their understanding and support, it would have been impossible for me to fulfill my DBA dream successfully.

Chapter 1. Introduction

1.1 Research Background

In the past two decades, China's rail transit industry has experienced booming development, starting with market exchange for technology, progressing to localization and ending with in-house innovation. During this period, China's government (CRRC) issued substantial national policy support, which provided broad development space for the relevant equipment manufacturing industry. At the same time, though, it imposed higher requirements for product function and stability. However, due to the relatively low technical threshold, the market competition of on-board PIS products is extremely fierce. With the rapid iteration of new technology, especially the development of digital technology represented by AI, the complexity of PIS software results in its failure which seriously affects the user experience by lowering the reliability and availability of the PIS service and, even worst, potentially affecting the operational safety of China's railway system. Therefore, it is worthwhile exploring the influencing factors and the action mechanism to raise the success rate of software development projects.

1.1.1 Application Background

(1) China's rail transit undertakes a large amount of domestic long-distance goods transportation and long-, medium-, and short-distance passenger transportation. With a land area of about 9.6 million square kilometres, China has a vast territory, a deep and wide inland, a large population, unbalanced resource distribution and industrial layout, and frequent regional exchanges (Yan, 2020). According to the *Medium- and Long-Term Railway Network Plan (Commission, 2004),* the total railway mileage will reach 175,000 km by 2025 including about 38,000 km of high-speed railway. (Guo et al., 2020) In the long term, the total railway mileage will reach 200,000 km by 2030.

The mileage of new lines put into operation between 2010 and 2020 fluctuated (as shown in Figure 1-1). In 2020, 4,933 km of new lines became operational, of which 2,521 km are high-speed rail lines, accounting for 51%.

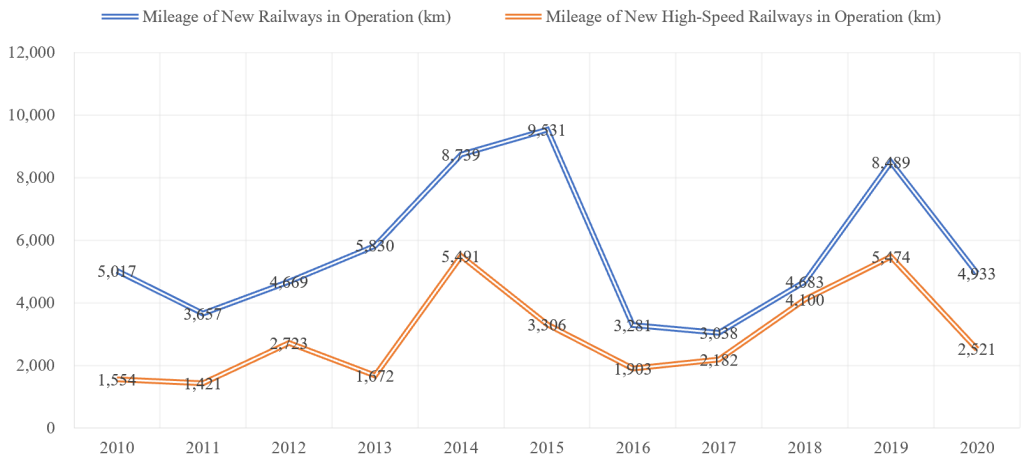


Figure 1-1 Mileage of New Railways commencing Operation in China from 2010 to 2020

Source: Prospective Industry Research Institute of State Railway Administration (eastmoney.com, 2021)

The current data of national railway network construction is that the national railway operation mileage reached 146,300 km by 2020, including 37,900 km of high-speed railway, double the length at the end of the 12th Five-Year Plan (as shown in Figure 1-2), accounting for more than two-thirds of the world's amount and leading the world. The national railway network density is 152.3 km/10,000 square kilometres, an increase of 6.8 km/10,000 square kilometres.

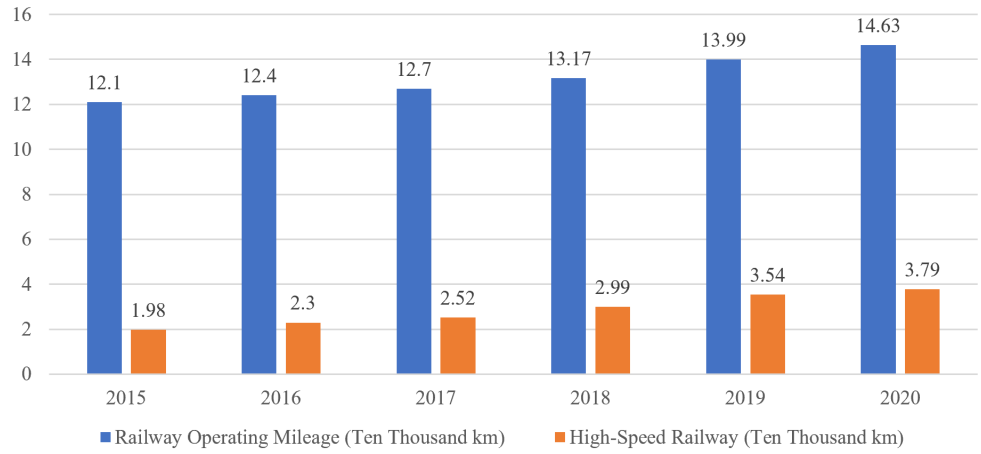


Figure 1-2 China's Railway Operating Mileage from 2015 to 2020

Source: Prospective Industry Research Institute of State Railway Administration (eastmoney.com, 2021)

China's investment in the rail transit industry has increased year on year since 2004 and

had stabilized at about 800 billion by 2009. In 2016, the State Council issued *the new Medium- and Long-Term Railway Network Planning (Commission, 2008),* and the national railway planning jumped from "four vertical and four horizontal" to "eight vertical and eight horizontal" (Guo et al., 2020). China's railway construction continued to advance, and the railway fixed assets remained high. From 2015 to 2020, the number of EMUs in China increased year by year. In 2019, the population broke through 3,600 standard units and reached 3,665 standard units, an increase of 443 standard units compared with the number in 2018, representing a year-on-year increase of 13.7%. In 2020, the forecast number of railway passenger carriages in China was 76,000, among which 3,828 units were expected to be standard EMUs (as shown in Figure 1-3).

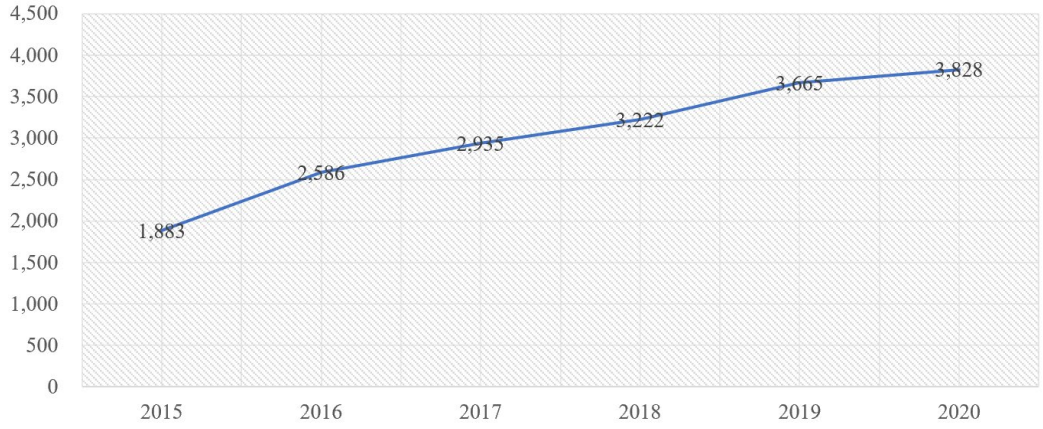


Figure 1-3 Number of EMUs in China from 2015 to 2020 (Unit: Standard Units)

Source: Prospective Industry Research Institute of State Railway Administration (chyxx.com, 2021)

In recent years, the output of EMUs in China has tended to be stable and has developed towards intelligence and lightweight. With the continuous expansion of the scale of China's high-speed railway network and the popularity of China's high-speed railway, it is expected that the demand for EMUs in China will continue to grow. The safety and environmental protection performance are continuously being optimized, and the flexible configuration, combined with artistic and aesthetic design, provides diversified passenger transportation services.

(2) URT has become the only way to protect urban development and the environment. The increase in motor vehicles with urban expansion has led to increasingly serious

traffic congestion and environmental pollution. The demand for a green transportation mode with safe, fast, low-carbon, low pollution and other travel characteristics of rail transit is becoming more and more urgent.

By the end of 2020, 45 cities on the Chinese mainland had opened 244 operation lines of URT with a total length of 7,969.7 km. (Meng et al., 2020) Among them, the metro operation line is 6,280.8 km long, accounting for 78.8%; other types of URT lines (such as monorail, LRV and tram lines) are 1,688.9 km long, accounting for 21.2%. The length of newly added operation lines in 2020 was 1,233.5 km.

During the period from 1 January 2016 to 31 December 2020, the so-called 13th Five-Year Plan period, the cumulative length of newly added operating lines was 4,351.7 km (as shown in Figure 1-4), with an average annual length of 870.3 km, which accounts for an average annual growth rate of 17.1% and is record breaking, being more than double the average annual length of 403.8 km of newly added operation during the 12th Five-Year Plan. The length of newly added operation lines during the 13th Five-Year Plan period exceeds the preceding cumulative total.

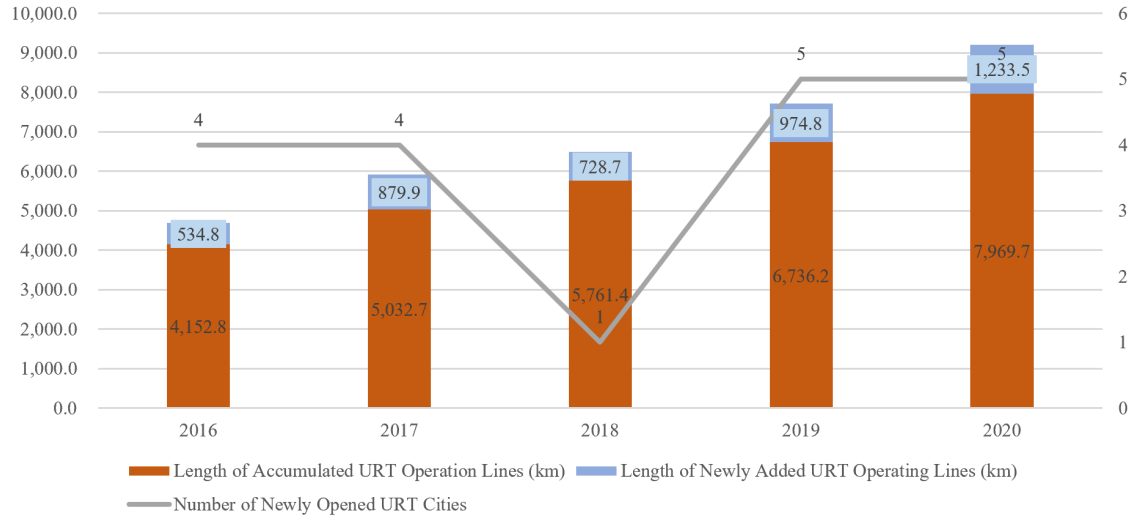


Figure 1-4 Mileage of Newly Added and Accumulated URT Operation Lines during the 13th Five-Year Plan Period

Source: Statistics and Analysis Report of 2020 URT by the China Urban Rail Transit Association (2021)

During the 13th Five-Year Plan period, the operating mileage of URT in China rose steadily, far exceeding that of developed countries, such as Germany, Russia and the United States. The total mileage in China even exceeded their total mileage, and China became the first country in the world in terms of URT operating mileage (see Figure 1-5).

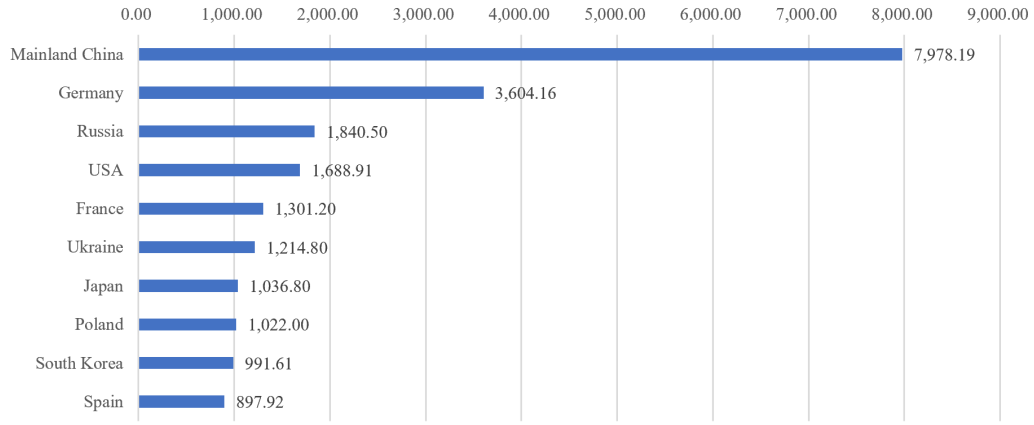


Figure 1-5 Top 10 Countries of Global URT Operation Mileage at the End of 2020 (Unit: km)

Source: *Urban Rapid Rail Transit* compiled by the Prospective Industry Research Institute (2021)

Shanghai, Beijing, Chengdu, Guangzhou, Shenzhen and Nanjing were among the top 10 cities in the world in terms of URT operating mileage in 2020, and the operating mileages of Shanghai, Beijing and Chengdu are listed as the top three in the world. The operating mileage of Shanghai URT is even about twice that of New York (as shown in Figure 1-6).

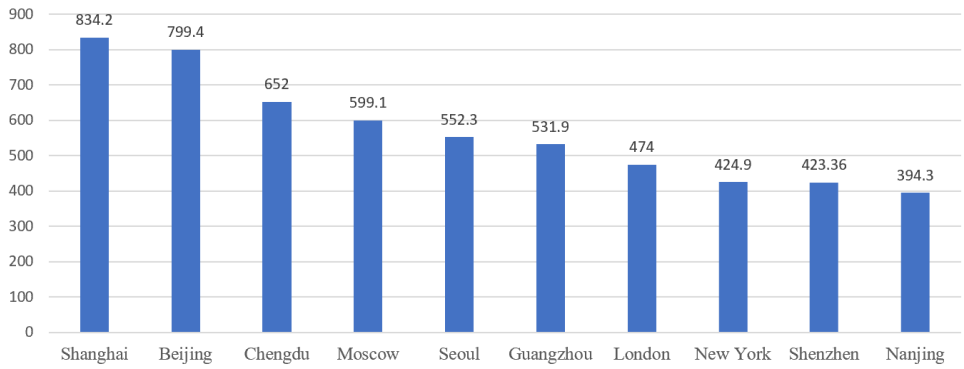


Figure 1-6 Top 10 Cities of Global URT Operation Mileage at the End of 2020 (Unit: km)

Source: *Urban Rapid Rail Transit* compiled by the Prospective Industry Research Institute (2021)

In 2021, the China Urban Rail Transit Association put forward four key development points for domestic URT to promote the development of smart city planning. At the same time, *the Fourteenth Five-Year Plan and the Outline of Long-Term Objectives for 2035* also suggested that, during the Fourteenth Five-Year Plan period, the operating mileage of China's URT will increase by 3,000 kilometres. It is expected that the cumulative passenger volume during the 14th Five-Year Plan will exceed 100 billion, and the cumulative completed investment is expected to reach 1,818.8 billion yuan.

(3) The development of the rail transit industry provides a very broad market space for on-board PIS software. The PIS, with passenger service as its main aim, will also enter a new stage with the development of rail transit. It will provide passengers with safe, efficient, diversified information and humanized services in the process of taking the train. Through correct service information guidance, passengers can travel safely and conveniently by rail transit and receive value-added information services; at the same time, the on-board PIS can improve the organizational efficiency of URT, reduce the disorderly flow of passengers, decrease the comprehensive social cost, greatly improve the service level, operation efficiency and emergency handling capacity of rail transit, and achieve the purposes of asset development and added value. Driven by the technological upgrading of network technology, communication technology and cloud computing big data, with the development of rail transit informatization, on-board PIS software, as the main interface between the traffic organization and the passengers, has higher and higher requirements for stability, reliability, convenience and real-time information, with coexisting opportunities and risks.

However, due to the relatively low technical threshold the market competition of on-board PIS products is extremely fierce. With the system becoming increasingly complex, although successful PIS implementation is common in China's rail transit, even when PIS software seems to be executed correctly, it may still fail under a few conditions or when specific conditions are met, which will seriously affect the user experience. More seriously, the failure of PIS software reduces the reliability and

availability of the system and potentially affects the operational safety of China's rail transit. According to statistics, although the value of on-board PIS accounts for less than 1% of the total vehicle price, it ranks among the top three customer complaints in the rail transit vehicle system: the door system (Dinmohammadi et al., 2016), air conditioning system and PIS, respectively. Through the analysis of the open items of 10 projects' first article inspection (FAI), it has been found that most of the customer complaints about PISs are caused by software problems, accounting for 58.99 per cent of complaints. Therefore, it is necessary to prevent the occurrence of PIS software failure. The influencing factors of PIS software failure have become a research topic for scholars.

1.1.2 Theoretical Background

Software development projects have specific characteristics such as human–computer interaction, high complexity and product versatility (Sarigiannidis and Chatzoglou, 2014, Jorgensen, 1999, Subramanian et al., 2007). These characteristics will undoubtedly make the development and management of PIS software more complex. Generally, PIS software project management focuses on four key aspects: personnel, products, processes and projects (Pressman, 2005); that is, software engineering involves human factors, project delivery, process control, a software development plan, software development methods and tools, and so on. From this perspective, PIS software project development is essentially a risk investment(Charette, 1992, Boehm, 1991), so it involves a high level of uncertainty.

According to Charette (2005), the most common factors leading to software project failure are unrealistic goals, wrong estimates, poorly defined system requirements, poor representation of the project status and unmanaged risks, so, even though it is widely performed in different fields, software development has a reputation for failure (Savolainen et al., 2012).

The impact of requirement management on software failure has been widely investigated by scholars. The requirements defined initially will almost certainly change,

which will affect the progress, cost (de Bakker et al., 2010, Glass, 2001) and realization of objectives (Wysocki et al., 2014, Kerzner, 2017). In particular, a lack of effective requirement management is reported as the biggest cause of software failure, leading to more than 70% of software failure problems (Kumar and Kumar, 2011, Muhammad Naeem Ahmed et al., 2013).

In addition, a direct relationship exists between risk management and the success or improvement of software development project performance (Jiang and Klein, 2000, Jiang et al., 2001, Raz et al., 2002, Wallace and Keil, 2004, Wallace et al., 2004b, de Bakker et al., 2010, Han and Huang, 2007). These studies showed that, to ensure that a project achieves its objectives, the risk factors should at least be identified and controlled (Menezes Jr et al., 2019). Therefore, the identification of risks or risk factors is regarded as the most influential activity in risk management (de Bakker et al., 2010, López and Salmeron, 2012, Neves et al., 2014). Risk factors are defined as conditions that pose a serious threat as "a danger or hazard" to the successful completion of software development projects (March and Shapira, 1987). The "inefficiency" of the risk identification process in complex system development is considered to be one of the main reasons for project failure (Reeves et al., 2013). Obviously, the identification of risk factors plays a vital role in the success and failure of on-board PIS software in rail transit.

Therefore, as PIS software is a special software product, identifying and monitoring the relevant factors that may lead to its failure is the core of PIS software project management and is essential to the realization of the PIS software's goal. This requires a comprehensive analysis and identification of the risk factors in the development project environment that may lead to PIS software failure, especially the types of problems that arise throughout the whole life cycle of PIS software (e.g., demand failure, design failure, coding failure and integration and interface failure), management activities (e.g., inspection, analysis and testing) and the severity of software failure (i.e., safety-critical and non-safety-critical failure). On this basis, more detailed empirical research needs to

be carried out to reveal and quantify the key factors affecting PIS failure as well as the structure of and the correlation between factors and further to study systematically the action mechanism of these influencing factors on software failure. Specifically, based on interviews with and analysis of PIS software providers in China's rail transit, this paper intends to focus on the following questions:

(1) What are the key factors affecting the on-board PIS software failure in China's rail transit?

(2) What is the influencing factor system of PIS software failure based on software risk?

(3) What is the mechanism through which influencing factors cause PIS software failure?

The main body of the thesis, composed of Chapter 4, 5 and 6, focuses on answering these three questions. Chapter 4 sets out to identify the factors that influence software development in China's rail transit system and construct the influencing factor system by combining semi-structured interview and literature review. While, Chapter 5 & 6 draws on structural equation modelling to study the action mechanism of influencing factors on PIS software failure.

1.2 Definition of the Research Object

1.2.1 PIS Software System

The on-board PIS of rail transit is a service system making rail travel more pleasant and efficient by providing passengers with all kinds of dynamic audio and video information in some rail transit trains, such as high-speed trains, EMUs and metros (a typical configuration is shown in Figure 1-7). The system comprehensively uses different kinds of technologies, such as network, communication, computer, display and control technologies, to provide passengers with information services. It is a comprehensive service platform integrating a vehicle (1998) operation information service, multimedia real-time information releases, radio and television programme production and broadcasting, television monitoring and equipment monitoring. A rail transit PIS mainly provides passengers with operation management information, such as train departure

times, arrival times and passenger instructions, as well as public information releases, such as news reports, weather forecasts, entertainment highlights, advertisements and live events. In the case of natural disasters, terrorist attacks and other emergencies, a PIS can provide dynamic emergency evacuation tips, information guidance and other contents so that passengers can take rail transit trains safely and conveniently.

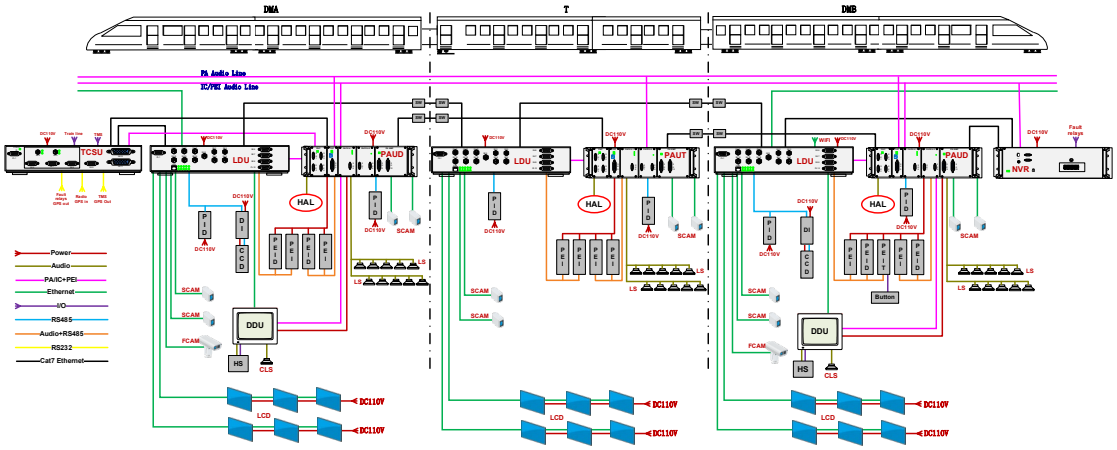


Figure 1-7 Configuration of a Typical On-Board PIS

(1) System Architecture

1) Traditional 485 System Architecture

The traditional on-board PIS is only passenger information system, which provides passengers with carriage number information, station information broadcasting and emergency communication and broadcasting. In the CRH1 EMU scheme, the traditional on-board PIS framework is shown as Appendix 1-1.

The control of the whole system is realized based on the traditional 485 communication system. In this scheme, the low-speed digital signal control (RS485 and CAN are applied instead of the Ethernet) is adapted to transmit analogue audio signals, the software development is less difficult, and the waterfall development mode can basically cope with FAI.

2) Upgraded 485 Plus System Architecture

With the increase in users' requirements for entertainment, advertising and information and the strengthening of security information control, the functional requirements of these two parts are also integrated into the on-board PIS. However, from the overall

perspective of priority and technology iteration, the audio broadcasting part, video entertainment and surveillance should still be handled separately. The audit broadcasting part including public address, still uses the traditional 485 communication technology, while the video parts employ network technology to build a network platform on which to realize networked transmission and control.

Subsequently, to meet customers' space-saving requirements, the audio carriage controller and the video carriage controller are combined at the physical level, and their functions are realized through a pluggable modular design. The upgraded system framework is shown in Appendix 1-2.

In this scheme, high-speed Ethernet control is added to the first-generation products to control the video digital signal with large amounts of data, especially in CCTV storage, playback, real-time playback, linkage with the emergency alarm and other functions. This architecture continues to follow the traditional waterfall approach to software development of which the weakest factor is the time issue, and long delays will have a high cost a lot in the subsequent software operation and maintenance stages (Zhang et al., 2006, Thummadi et al., 2011).

3) Full-Network System Architecture

With the development and application of network technology, since 2019, customers have basically unified the requirements for full networking. All the equipment in the system is controlled through the network, and broadcast audio signals are transmitted through the network. Only the UIC568 bus is reserved for the redundancy function in the degraded mode. The system framework of a full network on-board PIS is shown in Appendix 1-3.

(2) System Function

The setting of the train operation status and various information is mainly intended to help passengers and provide the arrival time, ticketing, ticket price and other relevant information regarding the coming train. All the above information can be obtained by the automatic train control (ATC) system.

1) Transfer Information

To ensure that passengers can ride or leave safely and quickly, the system can support the transmission of arrival, departure, speed and other information to passengers through audio and video communication.

2) Government Announcements

The system is also a powerful public information release platform, which can broadcast some announcements from the company or the government in real time according to the relevant publicity plans of government departments.

3) TV Programmes, Public Welfare and Commercial Advertisements

These high-quality media programmes present passengers with a wonderful audio and video with the help of the PIS. The signals of radio and television stations are introduced directly into the system to realize all or partial programme broadcasting.

4) Weather Forecast and Other Life Information

Passengers often have a considerable amount of free time when waiting for and taking the bus or train. The PIS can use this time to release all kinds of information in real time, such as the weather forecast, timely provision of local weather conditions and the urban pollution index, which greatly facilitate passengers' travel.

This is a real digital, intelligent and highly integrated scheme. The Ethernet has become the "nerve and blood vessels" of the system and plays an important role in the functioning of the whole system. Therefore, it also poses a higher challenge to software development. Based on the experience of project implementation, the "ISO 26262 V-Cycle R&D approach" (Pintard et al., 2013, Shitao et al., 2006) is basically used for software development.

1.2.2 Software Failure

PIS software failure is a typical software failure problem. Software failure refers to the inconsistency between the actual operation results and the expected operation results of the software, mainly including the failure of the system or its components to perform their required functions within the specified requirements, an accident or an event. If

such a situation or event is encountered, it may lead to the failure of the system or system components to perform as required. To support software development (Savolainen et al., 2012), the ISO and IEC (2008b) have jointly formulated various standards. A project is defined as "formulating the start and end dates of products or services according to the specified resources and requirements" (ISO/IEC 12207, 2008, p. 5). Based on the standards and traditions in the field of software development, the most common combination of standards used to measure the success of projects involves meeting the requirements of time, cost, functional and quality objectives (Anda et al., 2009, Atkinson, 1999, Koru et al., 2008). Software failure can be introduced at any stage of the software life cycle and can be associated with any type of software engineering (e.g., requirements, design and source code). (Hamill and Goseva-Popstojanova, 2009) Generally, the following factors are the main causes of software failure:

- Requirement uncertainty, including incorrect requirements, requirement changes and missing requirements.
- Software system design problems, such as design failure caused by human errors in the process of software design (for example, missing necessary component interactions).
- Coding failure, including errors in source code, such as typing errors, logic errors, algorithm errors and missing code.
- Data problems resulting in failure to respond to certain data patterns. These mainly include the omission or incompleteness of shared data and/or interfaces between components (or sub-components in some cases).
- System integration failures, including failures related to sub-components, components and/or subsystem integration.
- Other types of software failure, including ones that do not belong to any of the above categories, such as process or program problems, simulation errors, test problems (e.g., incorrect test configuration) and other problems of

non-integration with the software under development.

There are various manifestations of software failure. Generally, software failure mainly involves the following (Cerpa and Verner, 2009):

1) PIS infrastructure crash: the software stops producing output or does not respond to input;

2) Operation speed mismatch: the speed of data acceptance (input) or output is inconsistent with the requirements of the system;

3) Insufficient calculation accuracy: the calculation accuracy of one or some output parameter values does not meet the requirements;

4) Lack of output items: some necessary output values are missing;

5) Excess output items: the software outputs data/instructions that the system does not expect.

In critical areas such as rail transit, even small software errors or software failures may cause a series of major accidents and significant economic losses. With the emergence and fast development of automatic control and the internet of things (IoT), the quantity and complexity of PIS software are increasing rapidly, and the risks faced by software are expanding synchronously. Usually, PIS software failure will cause:

- Significant financial losses for the rail transit system;
- Personal injury and/or pain of passengers, such as unstable carriage temperatures caused by the software failure of the air conditioning system;
- Fear of rail transit technology, especially high-speed rail transit technology failure, triggering further fear of general failure of technology;
- Suspicions that software failure conceals fraud and corruption;
- Accidental damage to passengers' personal property due to software failure.

A failure in the software system will have a certain impact on the reliability of the system, which will usually lead to expensive emergencies, damage to the brand reputation, harm to the safety of the system and sometimes even casualties.

1.2.3 Problems Arising in PIS Software Development

The technology of on-board PISs in China's rail transit is maturing after dozens of years of development. However, with the rapid development of rail transit information technology and the increasing number of rail transit lines, the structure and characteristics, such as bridges and tunnels, of the lines are becoming more and more complex, the interaction between various lines is constantly intertwined and the transmission mode of PISs and vehicle information is becoming increasingly convoluted. The timeliness and stability of PIS software information has increasingly become the core issue of PIS software in China's rail transit. These problems are mainly reflected in the following aspects:

(1) Lack of data consistency management between the on-board PIS and the ground platform PIS. During construction projects of on-board and ground PISs, most of them are implemented by different manufacturers, resulting in different systems for the corresponding PIS layout design, layout distribution and message sending, and there are defects in data consistency management. Therefore, there are deficiencies in the immediate distribution and unified management of operation information.

(2) The on-board PIS live video signal is discontinuous. When the train moves at a high speed, due to signal interference, data packet loss and other factors of the train–ground wireless WLAN network signal, the livestream video signal in the line centre is discontinuous when received by on-board terminal, affecting the passengers' viewing.

(3) The stability of on-board equipment in high-temperature, shock and vibration environments needs to be further improved. The on-board PIS terminal, such as host equipment, operates in the harsh working environment of high temperatures, shocks and vibration for a long time, requiring high, continuous and stable operation performance of the equipment. Due to the many customized interfaces of the existing on-board host equipment, the equipment stability needs to be further improved.

For the above reasons, technical complexity and numerous equipment interfaces, there are problems in the on-board PIS software of China's rail transit (typical examples are

shown in Appendix 1-1), which exert a considerable impact on either the passengers or the operations of China's rail transit.

It can be seen that, although successful PIS software implementation is common in China's rail transit, it is an indisputable fact that there are various problems in PIS software projects. There are no recognized survey data on PIS software in China's rail transit, but, from the level of project management and the development history of the software industry, development projects of PIS software for China's rail transit have a high failure rate. In fact, researchers have questioned whether we have learned enough to ensure the success of software development projects (Cerpa and Verner, 2009).

1.2.4 Examples of PIS Software Failure

In order to demonstrate how software failure can occur, this section sets out some empirical cases.

Case 1: On 1 February 2007, the Harmony CRH1 EMU was officially put into service on the Guangzhou–Shenzhen line and the first train was numbered T971, travelling from Guangzhou East railway station to Shenzhen. This was the first time that this type of train had operated in the form of a coupling of two eight-carriage standard trainsets. After coupling at Guangzhou East railway station, both the external LED displays and the internal LED displays gave the wrong carriage number, and passengers on the platform could not identify the correct carriage and find their seats. Everything was in a mess. The Depot Operation Department urgently printed the carriage number on A4 paper, pasted them on the display and arranged for the crew members to guide the passengers onto the train and take their seats. After investigation, the cause of the problem was found to be that the software had identified the wrong address of the coupled carriage. Due to the conflict caused by the wrong address, it triggered the carriage controller to send the incorrect carriage number.

Case 2: At 4:51 p.m. on 13 December 2019, the driver of Wuhan Dahanyang tram T16 reported an error in the PIS: the HMI screen had stuck at 4:55 p.m. and the communication of multiple systems on the vehicle had failed and still could not be

recovered after resetting. The traffic dispatching center ordered the vehicle off the line at the Desheng terminal. The subsequent investigation revealed that the problem was caused by the failure of a resistor on PIS CAN network card, resulting in the IO port always being at the high level, which was detected by the software and response information was sent repeatedly. As a result, a huge amount of useless communication information blocked the whole CAN network of the vehicle, resulting in the communication failure of each system on the vehicle.

The above two cases are typical examples of PIS software failure causing serious complaints from customers, including but not limited to passengers, carriage builders and end-users as well as even financial claims and drops of new order intakes. Statistical data, either from China's high-speed trains or from urban metro carriages, demonstrate that an on-board PIS has a pair of opposite characteristics with low value but high reliability priority. By this means, it is well known that NCRs from PISs rank in the top three among all on-board systems.

1.3 Research Purpose and Significance

1.3.1 Research Purpose

The purpose of this paper is as follows:

(1) Through semi-structured interviews and coding tool, to identify the problems behind the failure of Company A's on-board PIS software and their causes.

(2) Combining the results of semi-structured interviews and the conclusions of a literature review, to construct the influencing factor system of PIS software failure and explore the causes and results based on the DEMATEL method.

(3) Through scale design, to use a structural equation model to study the action mechanism of influencing factors on PIS software failure and to make suggestions for improvement.

1.3.2 Research Significance

The significance of this research consists mainly of the following two aspects:

(1) Theoretical significance

This paper takes China's rail transit industry as the research background and explores PIS software failure, enriching the literature on software failure with a case study. Based on the semi-structured interviews conducted in Company A, this paper analyses the problem of PIS software failure, constructs the influencing factor system by synthesizing the literature review's conclusions, studies the causal relationship between factors using the DEMATEL method and then develops a scale to study the influencing mechanism, expanding the research on the influencing factors of software failure.

(2) Practical significance

Taking Company A as the case background, based on the company's current situation and the problems of PIS software development, this paper analyses the causes of the problems through semi-structured interviews and puts forward practical and detailed improvement countermeasures, which are conducive to improving Company A's software development performance. The research on the impact mechanism of PIS software failure in the rail transit industry has reference significance for promoting the development of the rail transit industry and the project performance management of related software companies.

1.4 Research Method

The main research methods used in this paper are the following:

(1) Literature analysis method: Reading literature is the way to find problems. This study obtains literature materials in relevant fields, such as software project success criteria, software project failure factors, risk factors and relevant research methods, through network resources, articles and books collected from prominent university libraries at home and abroad. The selected literature becomes the main material and theoretical basis of this study. Through reading and studying a large amount of documents, the author enriches the theoretical knowledge, absorbs the essence of research from experts at home and abroad and forms the theoretical framework of this study.

(2) Interview method: Through semi-structured interviews with the software

development project directors, managers, developers, maintenance personnel and quality management personnel of Company A, this study obtains relevant information and data, constructs the influencing factors of on-board PIS software failure and prepares for the subsequent empirical research. In the questionnaire design, middle and senior managers with rich development experience from industry-leading software enterprises are consulted and exchange their understanding and ideas on the relevant issues in this study many times. They put forward various suggestions for software projects and structural equation modelling, which provide timely solutions to some theoretical and practical problems encountered in this research and ensure the scientific nature, reality and operability of the research.

(3) Questionnaire survey and statistical analysis: The structured questionnaire is mainly used to obtain data. The sample selection is carried out appropriately, according to the concept definition of the research objects to which the questionnaire is distributed, which can control its effectiveness. The returned questionnaires are analysed using professional statistical software and the structural equation modelling method. The main analysis methods used are descriptive statistics, regression analysis, partial least squares and other statistical methods, reflecting the scientific nature and preciseness of the research.

The research involved collecting qualitative and quantitative data. The former relied on semi-structured interview and a three-stage coding process that involved 1) open coding, 2) axial coding and 3) theoretical, aggregate dimensions. In the quantitative research, the study used the DEMATEL method, drawing on questionnaire responses that were analysed using structural equation modelling.

1.5 Research Approach Design

1.5.1 Research Contents

The paper is organized in seven chapters. The first chapter introduces the research background, defines the research object, research purpose and significance, and research methods and presents the research ideas and framework of this paper. In the second

chapter, a systematic literature review is carried out, covering key concepts of software failure, success/failure criteria, software development risk, influencing factors, related research methodology and so on, through which the research gap is identified and the research questions are defined regarding the influencing factors that cause on-board PIS software failure and the mechanism through which they cause it. The third chapter presents the research design and a detailed summary of the research methods, including the semi-structured interviews, the DEMATEL method, the questionnaire survey and the structural equation model. The fourth chapter identifies the factors that influence PIS software failure by combining the qualitative analysis results of semi-structured interviews conducted in Company A and the literature review outcomes of software development projects' failure and risk; then, after two rounds of expert consultation, this paper constructs the influencing factor system of software failure and finally performs a quantitative analysis of the factors' causal relationship using the DEMATEL method. The fifth chapter contains the theoretical framework construction and questionnaire design regarding the factors that influence on-board PIS software failure. Combining the causal relationship of factors with the research results reported in the literature, the theoretical framework of this study in constructed. Based on the study of an existing mature scale, in conjunction with the interview research, the paper modifies the measurement items of the questionnaire, then conducts a small-sample pilot test, modifies the scale accordingly, finally, conducts a questionnaire survey and describes the data collection and questionnaire situation. The sixth chapter contains the empirical analysis and discussion of the results. First, the research model is selected and discussed, then the sample data are analysed descriptively, the correlations and structural model are verified and the theoretical model and relevant assumptions are tested through analysis. The seventh chapter presents the conclusion, contributions, research limitations and future research directions of this research.

1.5.2 Main Contributions

The main contributions of the research are primarily reflected in the following aspects:

(1) The research identifies the factors that influence PIS software failure, and clarifies the causal relationship between the influencing factors systematically through a method combining qualitative analysis and quantitative evaluation.

(2) Based on this causal relationship, this study has put forward a research model/framework and found out the impact mechanism and action path of factors through empirical analysis of the influence mechanism of PIS software failure for a more in-depth and comprehensive analysis.

(3) Some efforts have been done on the development of the scale including the design and selection of the variables in the research; hence, this research provides a reference for subsequent studies in the area.

Chapter 2. Literature Review

As early as 1975, researchers tried to extract the determinants of project success and failure to increase the likelihood of achieving successful results (Lucas, 1975). Since the mid-1980s, scholars have had a more detailed understanding of the reasons for and methods of system development. However, ensuring software success is still in a sense a major challenge (McLeod and MacDonell, 2011), and there are cases of software development failure in different fields (Savolainen et al., 2012). For a long time, the success rate of information technology application was not high. The highly popular failure data was provided by Standish Group, an international IT development and consulting company, based over 50,000 IT cases, as an approximate trisection in the categories (Kunert and von der Weth, 2018, Group, 2013); other scholars' research has indicated that their success rate hovers around 30 per cent (Ward et al., 2008). In the field of information technology project management, there are many typical failure cases (Nelson, 2007). Lu Xinyuan et al.(2006) stated that the success rate of domestic information technology projects is about 10–20% and that 80–90% of enterprises do not achieve the expected effect or even fail completely. This paper analyses the failure of on-board PIS software in rail transit, and explores its influencing factors and mechanism. This chapter starts with a general introduction to the PIS research literature, with some failure cases as a supplement to the insufficient research literature available in the specific field of rail transit PISs, then studies the characteristics of PIS software projects and subsequently summarizes its influencing factors, research methods and interactions in combination with the literature related to the failure of IS/IT projects to provide a theoretical basis for the following study of PIS software failure.

2.1 Research on PIS Software

The research corresponding to PIS software from the existing application literature is discussed, providing practical examples of PIS software failure and further characteristics of PIS software. Taking into consideration the limited literature available

on software failure specifically in rail transit PISs, some cases/examples are employed to fill the gap.

2.1.1 Application Research on PIS Software

(1) PIS Research on Rail Transit

Yao et al. (2013) systematically introduced the architecture innovation of on-board PISs from the perspective of the application of a train-level security detection sensor network to ensure network transmission of PIS data, resulting in the improvement of RAMS. He et al. (2006) developed an on-board digital PIS (DPIS) architecture based on peer-to-peer (P2P) and mobile agent technology. The DPIS enables on-board interactive communication and better satisfies passengers' expectations during long journeys. Yu (2012)'s research focused on the train–ground information exchange of PISs with the new network architecture to decrease interference and optimize communication. Wang et al. (2017) proposed a cognitive control approach to improve the performance (QoS) of both the train–ground communication "communications-based train control (CBTC)" and the PIS.

(2) PIS Research on Air and Bus Transport

An integrated air passenger information system is proposed to improve operation efficiency and on-board service quality (Csiszár and Nagy, 2017). Čelan et al. (2017) carried out a case study of a real-time PIS in the bus industry and concluded that the real-time passenger information (RTPI) decreases passengers' waiting time by reaching 92% time accuracy against predictions while also imposing higher requirements for the supporting facilities of the municipality. The same research was performed on ITSs and bus RTPI is by Politis et al. (2010), whose findings from 300 questionnaires survey indicated that bus passengers' travelling experience satisfaction is improved by 30% with RTPI.

(3) PIS Research on Whole Urban Transport Systems

Hannikainen et al. (2001) also studied the architecture of PISs for transportation, but they expanded the research object scope to the public transport service and introduced a

PIS named Tampere University of Technology Passenger Information System (TUTPIS), which could provide passengers with a personalized, real-time information service throughout their whole journey process. Surugiu et al. (2017) conducted intelligent transportation system (ITS) research on multimodal transportation for urban travel from the perspective of environmental optimization. A European PIS was proposed as a sustainable transnational interoperable PIS for public transport throughout the whole of Europe (Tibaut et al., 2012). The IT systems for transport companies (ITTC) is a new integrated model adopted by a public transport company to plan the schedule, dispatch resources, control the operation, provide passenger information and so on (Dohmen, 2017).

In general, few articles have studied PISs in the transport industry and even fewer have focused on rail transit. Most of the existing PIS articles have concentrated on technical architecture research and ITSs instead of the software failure phenomenon, let alone performing causal analysis or identifying PIS software failures' influencing factors. Very few attempts have been made by researchers to undertake in-depth investigations of the software failure mechanism of PISs for rolling stocks.

2.1.2 Characteristic Analysis of PIS Software Projects

The term software project refers to the whole process from the customer's demand to the realization of the demand; that is, the customer puts forward the requirements for the realization of a certain function. After receiving the customer's demand, the software company carries out the functional analysis of the demand and transforms it into technical requirements, and then the developers realize the customer's requirements through various software development means. As introduced in Chapter 1, an on-board PIS in rail transit comprehensively uses different kinds of technologies, such as networks, communication, computers, displays, and control, to provide passenger information services. It is a comprehensive service platform integrating a vehicle operation information service, multimedia real-time information release, radio and television programme production and broadcasting, television monitoring and

equipment monitoring. Yao et al. (2013) discussed the new architecture of on-board PISs for trains from the perspective of an Ethernet-based technical platform to increase the data efficiency of real-time data transmission and content updates. Yu (2012) concentrated on the optimization of the train–ground wireless communication of an urban rail PIS. TUTPIS services were developed to network passengers with companies that provide public transportation services, including buses, taxies and trains (Hannikainen et al., 2001). The digital passenger information system (DPIS) enables on-board infotainment and fully integrates passenger information with interactive communication, passenger information, video entertainment and surveillance (He et al., 2006). The interference between the PIS and the communications-based train control (CBTC) system influences the quality of the service provided to customers, and the authors have proposed a cognitive control approach to mitigate the interference (Wang et al., 2017). Real-time passenger information (RTPI) is the requirement from an intelligent public transport system, and Čelan et al. (2017) adopted the qualitative approach of Maribor as a case to improve the reliability to nearly 100%.

Compared with other traditional projects, an on-board PIS system has the following characteristics:

1. High Complexity

Whitney and Daniels (2013) researched for the root cause of IT project failures from the perspective of system emergency, non-monotonicity and non-ergodicity and conclude that it is complexity. Khan and Malik (2017) defined software complexity as “the most essential challenge seen in any software development process” (P17502). Thieme et al. (2020b), Thieme et al. (2020a) both traced the failures in software-intensive systems to the system complexity, system interactions and cascading effects. PIS development itself involves multidisciplinary knowledge, such as computer science, communication technology, management science and behavioural science. It is a comprehensive high-tech activity; the problems to be solved by a PIS are relatively complex. From the processing of general daily affairs to emergency management, it involves different

departments, institutions, personnel, equipment and business fields. At the same time, the needs of customers and the external environment are undergoing a process of dynamic changes. When developing, it is necessary to clarify the diversified needs and take corresponding measures, which will also increase the complexity. With the expansion of the construction scale in China, rail transit, such as high-speed railways and urban subways, and the rapid development and iteration of technology, the complexity of PISs is further enhanced.

2. Strong Uncertainty

Uncertainty is not only an important feature of software development projects but is also considered to be a key factor in the failure of IS/IT projects (Alami, 2016, Taipalus et al., 2020, Thomas and Mengel, 2008). On the one hand, the uncertainty of a system depends on data and information. In addition to their own inherent attributes, they are inextricably related to many factors, such as human quality, time and space, the environment and technical means. On the other hand, the continuous expansion of system development technology and development methodology not only imposes higher requirements for the sustainable development of the system but also increases the uncertainty of the system. These factors affect the cost and delivery time of the system and thus the success or failure of the whole system.

3. High Quality Requirements

Because a PIS provides support for customer service and travel decision making, incorrect and untimely information often generates dissatisfaction or is even fatal to passengers. Therefore, the reliability of the hardware is required to be high, and the machine and equipment should be able to be switched automatically to standby equipment without failure or, in the case of failure, to ensure the safety and reliability of the channel and the unimpeded flow of information. The software should not be wrong; otherwise, it will interrupt the operation of the whole system or produce incorrect processing results. For example, the data and information provided by the on-board CCTV system are used to help the driver in dealing with more complex travel

conditions and can better assist the driver and dispatcher to observe the operation.

4. Strong Systematicness

Considering the scale and complexity of rail transit construction and special software technologies, any PIS development needs to have its own new development conditions and requirements, conduct specific research and exploration, and comprehensively consider the resources of all the parties involved to complete the system development in time, with quality and quantity, and then optimize the configuration of the centre and platform equipment, realize the unified monitoring and centralized operation and maintenance of the system, reduce the hardware investment and the operation and maintenance cost, provide centralized storage and management of the data of the original independent on-board subsystem, integrate business data to a greater extent and provide the basis for subsequent data analysis and other functions.

5. Strict Maintainability

“The organization shall establish, implement and maintain documented processes to manage RAMS/LCC activities”, as stipulated in section 8.8 of ISO/TS22163: 2017 (Institute, 2017), which has special requirements to consider maintenance as an important activity throughout the project life cycle (Tsunoda et al., 2021) that must be applicable to electronic systems, including PISs. Software maintenance includes not only removing faults after software release, but also updating or modifying its functions because of the changes to requirements or application context (Tsunoda et al., 2021). The software maintenance phase tends to be ignored or emphasized less during the software development phase of the whole software life cycle (Chen and Huang, 2009), while it is the most costly (Yip and Lam, 1994) and least efficient phase (Sousa and Moreira, 1998) mainly because of developer ‘churn’ and knowledge loss (Etemadi et al., 2022). The study by Chen and Huang distinguished the different “problem factors in the software development phase” and the software maintenance phase. The results showed that DOCs and PGMs are the top two problem factors influencing software maintainability (Chen and Huang, 2009).

In short, software projects have greater complexity and uncertainty and weak process solidity, making the failure rate of software projects much higher than that of traditional projects, especially uncertainty, which poses a greater challenge to project management. Developers are not familiar with the business, and non-standard implementation activities are one of the objective reasons for project failure.

Therefore, this paper searches the target literature for the two aspects of software results and input elements. The results include “software failure”, “software success” and “IS failure”; the input perspective involves “influencing factors”, “risk factors” and “requirements management”. Given that the research focuses on on-board PISs of rail transit in China, corresponding literature on “PIS” as well as “IS” is also worth exploring and studying. The “key words” systematic literature database search plus the cross-reference snowballing study method are applied in the research to “decrease the biases and increase the review quality” (Stapic et al., 2016):

Step 1: Computerized key word research in Google scholar. The key words “failure of software development”, “performance of software development”, “influencing factor + software failure”, “risk factors + software success”, “requirement management” and “passenger information system + train” are respectively input to search for articles/ books/conference papers in the databases of libraries, mainly from the University of Manchester and Shanghai Jiao Tong University. Optional key words, “mixed methodology” and “combined research method”, are also adopted for an extension search.

Step 2: Snowball research from the citation and reference sections of the existing articles/books/conference papers. This is a “convenient and highly efficient way to find most of the main articles in the area sharing similar research topics by following authors’ citations and reference information” (Guo, 2016).

2.2 Research on the Success or Failure of Software Projects

The success of a project depends on a variety of evaluation criteria, and it has generally been considered by scholars at home and abroad that evaluation is difficult. The

performance or result of a software development project is defined as a success or failure from a specific perspective using predefined evaluation criteria. Here, specific perspectives refer to customers or suppliers and in some cases both. The evaluation criteria may be different. From the perspective of the project implementation process, once customer requirements have been met within the specified time and budget, a project is successful. From the perspective of project deliverables, if the final product or service meets the customer requirements, it is a successful project. Different stakeholders have different perceptions of success. (Ralph Jonkers, 2015) For the end-user, as long as the project satisfies the contract, it is successful, regardless of whether the project implementation process is successful. Meanwhile, the developer wants both the success of the project implementation process and the success of the project deliverables.

2.2.1 Criteria for Software Project Success

The result of software development may be a great success, but it may also be a disappointing failure. How can companies conclude whether a software development project is successful? Agarwal and Rathod (2006) considered a software project to be a success if the product was delivered with the predefined quality, at the given time and within the budget. They also stated that customers' satisfaction or stakeholders' interests are core criteria for defining a software project's outcome. Savolainen et al. (2012) performed a systematic literature review on the success and failure of software development, in which they distinguished the concept of project success/failure from that of project management success/failure. The use of traditional project management success/failure criteria, that is, time, budget, scope and quality, is not enough to determine the success/failure of software development projects. A sample of seven articles were studied, finding the three success criteria of "customer satisfaction, short-term business benefits and long-term business benefits" (Savolainen et al., 2012) mainly concerning stakeholders' interests (Agarwal and Rathod, 2006).

2.2.2 Views of Software Project Failure

Correspondingly, people's understanding of project failure is vague: "different scholars have different views". Although the cost, quality and progress are strictly managed and controlled in the process of project implementation, uncertain factors, such as the market environment, demand change, personnel change and technical development, occur from time to time, affecting the budget, lead period and quality of the project. With the expansion of the project scale, the development technology changes with each passing day, the process becomes increasingly complex and the development team becomes unstable.

In April 1997, KPMG's Schedule Management Department conducted a questionnaire survey on the causes of project failure in Canada, and the failure of projects was defined as follows: (1) the project cost exceeds the original budget by more than 30%; (2) the project progress exceeds the original plan by more than 30%; and (3) the project deliverables originally specified cannot be submitted (insufficient function or failure to meet expectations), or the project is cancelled or delayed in the middle of its process, except due to unplanned project changes and project priority changes. If any of the above three conditions is satisfied, the project is declared to have failed. KPMG's survey was conducted after the survey data released by the Standish Group (1995) (the Standish Group has published statistics and carried out research on the failure rate of IT projects since 1994 and issues the Chaos research report every 2 years). KPMG stated that overspending and overdue delivery of a project within 30% are "tolerable"; that is, it is still a successful project. However, the definition of project delivery delay and midway cancellation is not so accurate.

The Standish Group held that the definition of software project failure as "being canceled" or "the budget, delivery time or business objects of the project being not satisfied" (Linberg, 1999). Following this definition, the average failure rate of software development projects reaches 84%. Further data provided by Linberg indicate that 31.1% of software development projects end up being cancelled and the completed 52.7% are

189% over budget. In this article, a total of six sets of success factors were identified.

Ye (2006) analysed project failure through a literature review and expert interviews. He believed that there is no general method to judge the project failure standard in China and that all parties have serious differences in their understanding of project failure. As far as the project developer is concerned, the subjective opinions of the project manager and project executives play an important role. He also believed that the "Iron Triangle standard (time, budget and quality)" is a generally recognized standard. A. Ebad (2016) quoted Schwalbe's three criteria to define a software project not success: (1) not meet the standards of scope, cost and schedule; (2) unsatisfaction of customers or end-users; (3) the result does not meet the main target. While Nizam (2022) tried to build a process definition of software project failure, emphasizing "the process leading a project to failure" (p. 34426) by team behaviours.

In sum, the definition of software (project) failure is difficult to be agreed and this study defines the term mainly from the perspective of RAMS requirements for the rail transit industry.

2.2.3 Research on Software Failure's Consequences

With the development of modern technology, the scale of software is becoming more and more complex. A failure in a software system will have a certain impact on its reliability. Lyytinen and Hirschheim defined four major notions or categories as "correspondence failure, process failure, interaction failure and expectation failure" in their survey of empirical literature on IS failure (Kalle Lyytinen, 1987, Yeo, 2002) . Fenton and Ohlsson (2000) systematically reported the software failure problem based on comparative research on two versions of Ericsson Telecom applications, *release n* and *release n+1*. Subsequently, Andersson and Runeson (2007) and Eberlein and Grbac (2013) conducted two conceptual repetitions of Fenton and Ohlsson's (2000) research, and most of the results were quite consistent. Generally, software failure will lead to expensive emergencies, damage to the brand reputation, system safety hazards and sometimes casualties. Such failures, caused by either Bohrbugs or Mandelbugs, have

effects on end-users and operational personnel, resulting in “prolonged outages, increased maintenance costs and a low perceived quality” (Grottke et al., 2016). For example, according to the estimation from “the US National Institute of Standards and Technology (NIST), the U.S. economy loses $60 billion per year in costs” related to the development and distribution of patches to eliminate software failures, as well as productivity losses caused by computer malware and other problems caused by software failures (Zhivich and Cunningham, 2009).

2.2.4 Relevant Evaluation Methods

Different scholars have proposed different methods for the evaluation of information systems and information technology applications. For example, when Meredith and Suresh (1986) evaluated investment projects in advanced manufacturing technologies (AMTs), they classified the justification methods as economic, analytic and strategic approaches. Zhang Qing and Huang (2003) divided the current evaluation methods of IT investment value into three types: the financial performance evaluation method based on capital time value, the process-oriented evaluation method based on competitive advantage and process change, and the production function method based on microeconomics. Cui Yaodong et al. (2001) separated the evaluation methods into five categories according to the needs of IS project evaluation: pursuing economic benefits, obtaining a competitive advantage, maintaining a competitive position, realizing strategic objectives and realizing effective integration. They concluded that different scholars have different perspectives on IT project evaluation, which will directly affect the success/failure of a project.

2.2.5 Relevant Methodological Research

Rodriguez-Repiso et al. (2007) reviewed the three emerging methodologies to identify, classify and evaluate the CSFs, specially CSCs, the AHP and FCMs, from the traditional perspective of IT project success. Stoica and Brouse (2013) broke from the traditional factor research of IT project success with certain context perspectives and presented a multi-method approach combining system dynamics, intangible social factors and social

theory regarding the factor analysis of information technology (IT) project failure. Subsequently, Stoica and Brouse (2014) introduced adaptive and preemptive IT theory to identify the root cause on IT project success and/or failure from the perspective of reversing the pervasive and costly failure trend. Lin et al. (2021) investigated influencing factors on construction safety of high-speed train station by applying mixed-method research approaches including DEMATEL, ISM and questionnaire survey.

2.3 Research on the Influencing Factors of Software Project Failure

What factors actually lead to the failure or lack of success of software development project? Project success criteria do not match as different groups of stakeholders hold different perspectives (Agarwal and Rathod, 2006). Therefore, the specific client and vendor stakeholders' perspectives should be considered as another dimension that cannot be neglected in software development projects' success or failure evaluation (Haried and Ramamurthy, 2009). Most research on the success/failure of software development projects has taken into consideration the dialectical correlation/process dimensions of both the client and the vendor: the analysis of the dynamic client–vendor relationship (Heiskanen et al., 2008), the transfer of risk from the client to the vendor (Natovich, 2003), the relational client–vendor approach to IT sourcing projects (Haried and Ramamurthy, 2009), the vendor's trust in the client and the client's control over the vendor (Mao et al., 2008) and expertise coordination-related coproduction (Shim et al., 2010). Some studies have also been carried out from the perspective of either the vendor or the client, specifically risk assessments performed by vendor firms for outsourced IT projects (Taylor, 2007, Na et al., 2007) and meeting the "scope" of software development as the strongest determinant of success (Agarwal and Rathod, 2006).

2.3.1 Reasons for the High Failure Rate of Software Projects

Scholars have their own views on the causes of project failure. Van Genuchten and Koolen (1991) believed that the main reasons for cost overruns and schedule delays in software projects are the continuous change of design, overly optimistic plan,

underestimation of difficulties, unplanned activities, incomplete requirements, and changes in customer management. Jones (1994) put forward some new explanations for the failure of software project development. He suggested four fundamental reasons for cost overruns and schedule delays: the cost/duration prediction cannot be completely correct at the beginning of the project as the project is still ongoing; when the cost and schedule are determined, the scope of the project begins to expand; inadequate planning and estimation means; and the project cannot obtain sufficient historical data. According to the investigation of project failure by the Steady Group, 80% of project failures make people pay more attention to the project manager (Sijun, 2002). Professor Jinzhi (2014), from the Institute of Mathematics and Systems Sciences of the Chinese Academy of Sciences, believed that the lack of end-user participation and incomplete requirements are the top two reasons for project failure according to a survey carried out in USA on 8,000 software projects from 1995.

Dwivedi et al. (2015) reached the conclusion that the reasons for failed IS implementation are complicated and contested by enumerating plenty of information system failure examples. The authors summarized the mainstream articles "research investigating IS failure proposes different perspective", as Table 2-1 shows, among which Heeks's model of design–actuality gaps provides a good explanation for the high IS failure rate in developing countries(Heeks, 2002, Heeks, 2006).

Perspective	Category	Factor	Reference
Social and organizational	Expectation failure	Correspondence process interaction	Hirschheim and Newman (1988)
	Termination failure	Termination	Sauer (1993)
Project management	Process	Poor estimation and/or scheduling insufficient risk management insufficient planning shortchanged quality assurance poor requirements determination contractor failure insufficient management controls wasted time in the fuzzy front end code-like-hell programming abandonment of planning under pressure inadequate design insufficient resources planning to catch up later	Nelson (2007)
	People	Ineffective stakeholder management weak personnel and/or team issues insufficient project sponsorship inattention to politics lack of user involvement unrealistic expectations undermined motivation wishful thinking friction between developers & customers heroics adding people to a late project premature or overly frequent convergence noisy, crowded offices uncontrolled problem employees	
	Product	Scope creep research-oriented development requirements gold-plating push-me, pull-me negotiation developer gold-plating	
	Technology	Silver-bullet syndrome lack of automated source-code control overestimated savings from new tools or methods switching tools in the middle of a project	
Enterprise systems	Organization-enterprise system misfit	Functionality data usability role control organizational culture	Strong and Volkoff (2010)
Developing countries	Archetypes	Country context gaps Hard-soft gaps Public-private sector gaps	Heeks (2002, 2006)
User resistance	Individual issues	Uncertainty input Control/power self-efficacy	Klaus and Blanton (2010)
	System issues	Technical problems complexity	
	Organizational issues	Facilitating environments communication training	
	Process issues	Jo/job skills change workload lack of fit	

Table 2-1 IS Failure (Dwivedi et al., 2015)

Lehtinen et al. (2014) applied the root cause analysis method to analyse the causes of

software project failure and their relationship through case studies of four software development companies. For each case, a causality diagram containing 130–185 causes was identified. Different cases had similar results. On average, 50% of the causes were bridging ones, among which a lack of cooperation, task backlog and lack of software testing resources were the most common. Bridge causes and causes related to the task, people and method frequently appeared among the issues considered to be the most feasible objects for process improvement. Causes related to the project environment often occurred but were rarely regarded as feasible objects for process improvement. Therefore, they believed that, to prevent software project failure, it is necessary to control areas outside the process that lead to failure in combination with case analysis, which requires cooperation between individuals and managers who are responsible for different process areas. Nundlall and Nagowah (2021) reviewed the literature related to development task allocation and team collaboration under the mode of distributed agile software development and identified different influencing factors, dependency factors, challenges and methods. Simão Filho et al. (2017) also emphasized the importance of team collaboration. Gauld (2007) filled the void of public sector IS failure and explored it using the experience and lessons of hospital organizations in New Zealand, concluding that a higher failure risk exists in the public sector because of organizational and political influences.

2.3.2 Factors Affecting Software Project Results

As early as 1975, scholars tried to extract the determinants of project success and failure from the aspect of results (Lucas, 1975). McLeod and MacDonell (2011) reviewed the relevant factors affecting the output of software system development projects by analyzing the relevant literature in the period 1996–2006, which provided a good reference for the research on software project failure. In their research, they identified and classified factors that have an impact on software development outcomes (as shown in Table 2-2). Tamburri et al. (2021) reviewed papers on success and failure of software engineering published between January 1970 and August 2019, “harvesting 561 factors

and 40 core concepts in total, and final 14 clusters of factors for success- and failure-specific measurement and risk-analysis" (p. 599). With reference to the Walsham's (1993) theoretical research results as their conceptual basis, they constructed a reliability research framework and sorted out the factors affecting software development project outcomes from four aspects: institutional context, people and action, development processes and project content.

Study	Scope	Categories (as defined in original study)
Lyytinen and Hirschheim [1987]	Classificatory framework of reasons for system failure	1. Features of the system—technical and operational 2. Features of the system environment—lack of fit of the system with users, rest of organization, or operating environment 3. Features of the systems development process—methods, decision-making, assumptions, nature of work, implementation, contingency factors 4. Features of systems development environment—cognitive and skill limitations of analysts or users
Poulymenakou and Holmes [1996]	Contingency framework of variables affecting system failure	1. Macro (organizational) contingent variables—culture, planning, accountability, irrationality, evaluation 2. Micro (project) contingent variables—power and politics, user resistance to change, development methods
Butler and Fitzgerald [2001]	Model for user participation and management of change in systems development	1. Institutional context—organizational policy 2. Project-related factors—project initiator, top management commitment, type of system, project complexity, task-structure, time for development, available financial resources, resultant change 3. Process-related factors—user-developer relationship, influence and power, communication 4. User-related factors—user perceptions, commitment, willingness, ability, characteristics and attitudes
Scott and Vessey [2002]	Model of risk factors in ERP system implementations	1. External business environment 2. Organizational context—culture, structure, strategy, IT infrastructure, business processes 3. Systems context—data, technology, project governance 4. Project—project focus and scope, project management, change management

Table 2-2 Classification of Influencing Factors on Software Development Outcomes (McLeod and MacDonell, 2011)

Al-Azzani et al. (2014), Khan et al. (2013) stated that "ambiguous and unrealistic requirements are major sources of failure in the software-intensive systems" (p. 21); they stem not only from the user and the client but also from the environment. Pan and Pan (2006) regarded "abandonment decision-making" of IS projects as a type of software project failure and further studied the impact of coalition dynamics on decision making. Machuca-Villegas et al. (2021) studied the influencing factors of software development productivity, emphasized the role of social and human factors (SHFs),

designed a tool containing 79 items to evaluate 13 different SHFs and then evaluated the validity and reliability of the tool, obtaining positive results. Trendowicz et al. (2008) identified the most influential factors on software development productivity by integrating data and expert factors analyses. Ahmedshareef et al. (2014) applied mixed methods and utilized actor–network theory (ANT) to "expose the influencing factors on project schedule delay". Nundlall and Nagowah (2021) and Simão Filho et al. (2017) identified a larger number of influencing factors (including dependencies and alternatives) in distributed software development in the remote team context for task allocation and coordination. Since most of the studies have conducted a mixed research approach to evaluate or to perform data collection, both of the articles proposed a mixed method as a research approach to the study of this field, while neither article has incorporated the factors in any model, framework or mechanism to overcome the challengers.

Soft (non-technical) factors have been stressed as affecting the software development team performance through such aspects as the team climate, support, diversity, conflicts, leadership, management, governance and even trust and communication. Goparaju Purna et al. (2011) classified the factors affecting the performance of software development teams through a literature review and emphasized the soft (non-technical) factors, including the "team climate, team diversity, team innovation, team member competencies and characteristics, top management support and team leader behavior", from which mutual trust and communication effectiveness were prioritized as factors affecting team performance.

Haq et al. (2019) studied the relationship between the project governance mechanism and the software development project performance based on the moderating role of requirement risk. The results of the research showed that both contractual and relational governance has a significant impact on project performance and helps to reduce opportunism. What is more, the requirements risk tends to moderate negatively this impact of governances on performance.

2.3.3 Factors Affecting Software Failure

Many factors affect the failure of software development projects. Some research has demonstrated empirically that "a software project failure is a result of a multidimensional process where people, tasks, methods and project environment are interconnected" (Lehtinen et al., 2014). The nature of the failure will vary from project to project; however, common traits of software failures are likely to be a failure to deliver the software on time, significant problems with the functionality of the software doing what it should do, and the use of more resources within the firm and its partners to develop and deliver the software. Explaining what software failure occurred is the main focus of research on software failure. For example, Ostrand and Weyuker (2002) explored the method of identifying faults by studying the relationship between fault distribution, index range, fault density and fault persistence in 12 versions of an inventory control system from pre-release to post-release and from one version to the next. Hamill and Goseva-Popstojanova (2009) studied two cases, one concerning "software data from a NASA flight mission" and the other involving "applications from C compiler of the GNU compiler Collection (GCC)", to explore fault location and the main fault types in large-scale software applications. The results showed that requirement faults, coding faults and data problems are the top three types of NASA mission software faults.

Interpersonal interaction, high complexity and product versatility are considered to be the three main factors affecting software project failure (Sarigiannidis and Chatzoglou, 2014, Jorgensen, 1999, Subramanian et al., 2007). Charette (2005) believed that the most common factors leading to software project failure are unrealistic objectives, wrong estimation, unclear definition of system requirements, poor description of the project status and uncontrollable risk. Zahid et al. (2018) found similar factors including "incomplete requirements, ambiguous goals, time shortage, cost management, lack of user involvement during the development" (p. 67). However, research on software failure in different fields has shown that the influencing factors of software failure are

highly heterogenous. Therefore, a systematic strategy was adopted to identify the factors of software failure is very important. Both Shou and Ying (2005) and Yeo (2002) conducted research on the CFFs of ISs, respectively, in Chinese and Singaporean enterprises to define the most influential failure factors. Yeo first developed a triple-system model, which was used to assess Singapore-based survey data to evaluate the most influential factors of IS/IT failure. Shou and Ying categorized the CFFs from the literature review and conducted a survey in Zhejiang, China, to identify the most influential failure factors. A comparison of the results between the two articles revealed similarity and differences: Yeo emphasized “project planning” and “corporate culture” as the top two CFFs, while Shou and Ying ranked “corporate culture” and “corporate management” as the most influential CFFs. The main difference was that one of the Chinese managers concentrated too much on managerial issues when implementing IS “because of immature corporate culture and management” while Singaporean managers paid attention to “the process- and outcome-driven” strategy.

A. Ebad (2016) found the main factors affecting the software project failure to “be more human related than technical or financial factors. In particular, lack of top management support, organization culture, business process reengineering, lack of training and unavailability of PMO” (p. 1151) Top management support (TMS) is not only regarded as the most important factor for project success (CSFs) in the context of information systems (ISs) (Young and Jordan, 2008, Rockart and Crescenzi, 1984, Doll, 1985, Lederer and Mendelow, 1988), but also seems to be stronger in necessity than sufficiency (Young and Poon, 2013). In this way, the board of directors as well as top managers and high-level executives should admit the impact of their control on the success/failure of software projects.

Whitney and Daniels (2013) examined the basic reasons for the failure of IT project management put forward in the existing literature and studied the complexity of projects from the perspective of system emergency, non-monotonicity and non-ergodicity. It was found that complexity itself is the real root cause of the failure in IT project

management. Alami (2016) constructed a complex ecosystem of software projects, analysed the case from the aspects of frequent changes of the supply scope, uncertainty, ambiguity and complexity using second-hand documents, studied the qualitative data obtained from the analysis of second-hand documents through the grounded theory methodology and concluded that failure is an opportunity for learning and improvement.

Taipalus et al. (2020) carried out systematic research on the uncertainty of information system development: following semi-structured interviews with 11 professionals in the field of information system development, they analysed the interview contents by applying the traditional content analysis method and further studied the causes of uncertainty, the impact of uncertainty and the mechanism for dealing with uncertainty to provide a practical reference for the information system development industry and education.

Multiple sources of the failure of software development projects result from poor requirement management, and Kumar and Kumar's (2011) report stated that 71 per cent of software development failures are due to this problem, which is the single biggest cause of project failure. The term "requirement" is used to describe the set of "desired characteristics and attributes possessed by a particular product or a service" (Ali, 2015). Software requirements have been defined as "formal descriptions of customer demands on software system solution" (Leffingwell and Widrig, 2000). The demands usually refer to a textual statement about capabilities of the software system. Software requirements include both functional and non-functional categories (Pozgaj et al., 2003), which exist throughout the software development life cycle, and many software problems are possibly caused by "poor requirement definition, improper requirement use and frequent requirement change". The poor management of requirement change following customers' requests would cause consequences of higher project costs, late delivery and poor quality (Loconsole, 2004). All these consequences could lead to the failure of software development projects. Ali's (2015) research showed that more than

half of software development failures can be attributed to poor requirement management. This can mainly be traced back to the characteristics of tacit knowledge, the changes of customers' requirements and so on. On the one side, it is challenging to translate customers' or experts' tacit knowledge into readily understandable forms (Nonaka and Konno, 1998). On the other side, the software development requirements change frequently within a single project or among different projects, even those from one customer.

In addition, software outsourcing is a decision that is often encountered in software development projects. Verner and Abdullah (2012) carried out research on the risks of outsourced software development project failure from the perspective of the client through a single case study, and the conclusion showed that the failure of the BSkyB project studied resulted from both the client and the vendor, while the major fault was judged to lie with the vendor. The finding that the vendor took responsibility for the major failure fault is also worthwhile for this paper to verify its application in the PIS business.

Jorgensen (2014) studied the factors influencing the failure of small and medium-sized software projects in the global outsourcing market based on more than 780,000 projects and/or tasks. Through the analysis of a binary logistic regression model, it was found that 74% of project failures can be correctly predicted during their start-up phase, and it was considered that the customer's characteristics are almost as important as the supplier's contribution to the project failure. The risk of failure will increase with the customer's emphasis on low price and project size.

2.3.4 Factors Affecting the Software Implementation Process

There are different types of risk factors in each phase of the SDLC to which a software development project becomes susceptible (Hijazi et al., 2014). This situation is defined as the main threat to the successful completion of software development projects (March and Shapira, 1987). Hijazi et al. (2014) identified and explained the risk factors in each phase of the SDLC and provided a detailed list of 100 risk factors. Pontakorn

Sonchan (2014) listed the top 20 software risks following a content analysis and Delphi study. Many studies have shown that there is a direct relationship between risk management and software development project failure (Jiang and Klein, 2000, Jiang et al., 2001, Raz et al., 2002, Wallace and Keil, 2004, Wallace et al., 2004b, de Bakker et al., 2010, Han and Huang, 2007). Many other studies have performed risk-factor analysis from different perspectives such as of an organizational risk diagnosing (ORD) framework (Vrhovec et al., 2015), of planning software development (top ten risk planning software development factors mitigated by using thirty control factors) (Elzamly and Hussin, 2015) and of inadequate testing resources (proper testing, test planning and QA team) (Gamage, 2017).

Correctly identifying and monitoring risk factors play a decisive role in the success of software development projects and software quality (Jiang and Klein, 2000, Jiang et al., 2001, Raz et al., 2002, Wallace and Keil, 2004, Wallace et al., 2004b, de Bakker et al., 2010, Han and Huang, 2007). The identification of risks or risk factors is the most influential activity in risk management (de Bakker et al., 2010, López and Salmeron, 2012) and is widely performed in agile and traditional software development methods (Neves et al., 2014). The low efficiency of the risk identification process in complex system development is considered to be one of the main reasons for project failure (Reeves et al., 2013). Boehm (1989) believed that software project risks can be divided into two types: general risks, which are common to all projects, and project-specific risks, among which some are easy to identify and manage and others are less obvious or it is more difficult to predict their possibility or impact. Boehm(1991) suggested 10 risk factors of IT projects: a shortage of personnel, an unrealistic budget and arrangement of funds, their function and nature, an inability to meet customer requirements, insufficient customer participation, the addition of new functions, changing requirements, a lack of new technical support, an unstable project team and insufficient support of senior leaders. Mizuno (2000) and others have shown that the nine important risks of software projects are unrealistic customers, too optimistic an estimation of technical problems,

insufficient empirical estimation of successful projects, wrong employment, unclear responsibility and unclear authorization, the low morale of some developers, insufficient understanding of ensuring the continuity of employees by some managers, insufficient change management of requirements or specifications and a lack of progress report. Reifer (2001) proposed 10 risks faced by software projects: a shortage of manpower, inconsistency with the business objectives, an unrealistic expected construction period among customers, variability of plans, instability of demand analysis and design, variability of software objectives, new methods and unstable tools, high employee mobility, friction within the development team and an inefficient working environment. Mc Farlan (1981) identified three dimensions of project risks, namely the project scale, technical experience and project organization structure. He suggested that the project manager needs to organize a comprehensive software project document for the software project. Chittister (1994) divided the risks into three "viewpoints": the functional viewpoint (i.e. risk factors related to requirements, products, processes, personnel, management and environment), the source-based viewpoint (i.e. risks related to hardware, software, organizations and individuals) and the temporal viewpoint (i.e. risks associated with different stages of the software development process). Lyytinen & Hirschheim (1987) put forward communication failure, process failure, interaction failure and expectation failure as risk factors. Lyytinen (1996) classified risks into risks arising from the system environment (i.e. users may not have experience in using the software system type under development), risks arising from the development environment (i.e. developers lack experience in analysing this system environment type) and risks arising from the management environment (due to the bias, laziness and ignorance or inaction of managers who ignore valuable information). Through a questionnaire survey, Karolak (1998) found that software project risk is reflected in three aspects, technology, cost and schedule, and runs through the whole development cycle. The technical aspects are related to performance and functionality; the cost includes the budget, profit and so on; and the schedule consists of the flexibility and

reality of the schedule. Pressman and Maxim (2015) divided risks into seven risk categories: product scale, business impact, customer characteristics, process definition, development environment, construction technology and the number and experience of personnel. Each risk category contains eight to 20 risk items. Barki et al. (1993) investigated 120 projects and identified 35 risk indicators from five dimensions: technological innovation, implementation scale, expertise, implementation complexity and organizational environment. Higuera (1994) of Software Engineering Institute (SEI) believed that software project risk is an organic system composed of people, technology, software, hardware, construction period and cost. Kliem (2000) identified 38 risks of business process reengineering project and divided them into four categories: manpower, management, business and technology. Wallace and Keil (2004) and Wallace, Keil and Rai (2004a, 2004b) defined 27 software risks, which they divided into six dimensions, namely user, demand, project complexity, planning and control, team and organizational environment. Du et al. (2007) identified six risk factors through a questionnaire survey of 118 IT project experts and 140 IT novices: the suitability of project methods, user involvement, the use of project management rules, the similarity to previous projects, the project complexity and the change frequency of project requirements. The software sector in the industry also has its own deep insights into the risk of software producers. It has divided it into 10 dimensions: resources, requirements, the technology development process, the project interface, the management process, the development system, the design, the management method, the working environment, and integration and testing (Redmill, 1998). Yeo (2002) proposed a three-factor model. Xue and Jia (2004) divided software risk into seven dimensions: scope risk, quality risk, schedule risk, cost risk, technology risk, human resource risk and legal risk. Pontakorn Sonchan (2014) extracted and classified the top 20 risk factors from the 30 most frequently cited and recently published documents on software project risks, as shown in Table 2-3.

Class	Category	Risk Item	Time
Product Engineering	Requirements	Unclear customer requirements	29
		Requirement creep	27
		Unable to meet user requirements	9
	Code and unit test	Lack of technical skills	14
		Technical complexity	10
	Integration and test	Low software performance	11
Development Environment	Development process	Inefficient team capability	16
		Inappropriate development process	8
	Development system	Problems with new technology	13
		Inadequate infrastructure	8
	Management process	Unrealistic schedule	17
		Optimistic resource planning	16
		Lack of executive involvement	12
	Work environment	Communication gaps	21
		Conflicts among team members	11
Programme Constrain	Resources	Staff turnover	17
		Unrealistic budgeting	15
		Resource insufficiency	9
	Programme environment	User resistance	21
		Lack of law enforcement	9

Table 2-3 Top 20 Most-Cited risks in the Literature Categorized by SEI Taxonomy-based Risk Identification (Pontakorn Sonchan, 2014)

Most of the literature on software risk management has focused on risk identification and analysis, that is, risk assessment (Boehm, 1991, Fairley, 1994, Redmill, 1998). However, Bannerman (2014) believed that few studies have paid attention to risk management in software development projects. Although they have realized its necessity, they have rarely applied it. Boehm (1989) proposed the concept of “software risk management”, and Boehm (1991) elaborated the composition of the software risk management system and considered that software project risk management refers to “trying to standardize the control of the risks affecting the success of the project with a

feasible principle and practice" and that its purpose is to "identify, describe and eliminate the risk factors, so as to prevent them from threatening the successful operation of the software". Redmill (1998) asserted that software risk management aims to evaluate and control the risk factors that may lead to adverse effects and reduce the risk to a range that is acceptable to managers. The Software Engineering Research Institute of Carnegie Mellon University in the United States (SEI), from the perspective of guiding practice, has successively put forward classified risk identification, the continuous risk management mode, team risk management, software risk assessment and so on, forming a complete SEI system. The software Capability Maturity Model Integration (CMMI) proposed by the SEI listed risk management in the key process area at the third level. The IEEE (2001) also formulated the process specification of software risk management in combination with the software life cycle model.

Higuera and Haimes (1996) proposed continuous risk management based on the SEI framework model, which divides risk management into five steps: risk identification, analysis, planning, tracking and control. The risk management method is a continuous cycle, and its core is risk communication. It requires attention to be paid to various activities of risk management in all the stages of the software project life cycle. Inspired by the SEI continuous process improvement and the PDCA quality management method, Hall proposed six discipline risk management models (Redmill, 1998). Kontio (2001) proposed a set of software project risk management theories with "Risk Analysis Diagram" as the core and provided a risk management improvement framework to support the continuous and systematic improvement of the risk management process. Haimes et al. (2002) developed a set of risk management systems called "hierarchical holographic modeling" (HHM) and applied it to software project risk management. Brown (1998), after studying a large number of events, found that, in the development process of software projects, about 30% cannot effectively control the risks because they cannot meet the expectations, which directly leads to their failure. Other projects also lead to the slow progress of software development for similar reasons, which

increases the cost. From the perspective of software project management, software risk includes any unplanned and unforeseen events that may have a negative impact on the cost, time or quality of the project. April H. Reed and Knight (2011) and Arnuphaptrairong (2011) stated that inexperienced employees, personnel mobility, insufficient budget and delay are some highly mentioned risks that may occur in each stage of a software project. Khan et al. (2014) identified communication risks in the context of global software development.

Alter (1979) identified eight risk factors, specifically uncooperative users, multiple users or implementation units, handover between multiple participants, an inability to specify goals or purposes, an inability to buffer the impact on others, a lack of support, a lack of experience and the effectiveness of technology or investment. Li et al. (2009) classified 82 risk factors into 14 types that will lead to software project failure, analysed the four stages of software project risk management, constructed a risk matrix model based on the possibility and influence of risk and divided software project risk into three levels according to the matrix model, namely high risk, medium risk and low risk. Huang and Han (2008) analyzed the risk impact of software projects from six risk exposure dimensions of software projects: users, requirements, project complexity, planning and control, team and organizational environment. Samantra et al. (2016) applied an interpretive structural modelling (ISM) approach to identify clearly the hierarchical relations of 23 risk factors from extensive literature review “which impose negative impact on schedule time, quality, cost, requirement or total failure for the software projects” (p. 20).

2.4 Conclusions of the Literature Review

To sum up, scholars have carried out extensive and in-depth research on the failure or success of software development projects. The research on project success criteria, project failure factors and project risk management has greatly promoted the development of project management in the field of software development. Although requirements defect, incompetence of project manager, insufficient support of senior

leaders, inadequate project management and nonstandard development process have often been mentioned as failure factors, a unified cause of project failure has not yet been formed. Most studies have referred to different criteria of software project failure or success in addition to the traditional factors of cost, time and scope (quality and functionality) (Savolainen et al., 2012, Agarwal and Rathod, 2006, Heiskanen et al., 2008, Natovich, 2003, Mao et al., 2008, Taylor, 2007, Chow and Cao, 2008, Na et al., 2007). Besides the relationship between client and vendor resulting in trust and control, risk has been widely discussed in many articles as an important factor for project success (Haried and Ramamurthy, 2009, Heiskanen et al., 2008, Natovich, 2003, Mao et al., 2008, Taylor, 2007). The lack of effective requirement management has been considered to be the biggest cause of software failure, resulting in more than 70% of software failure problems (Kumar and Kumar, 2011, Muhammad Naeem Ahmed et al., 2013). However, the problem of software development in relation to the industry background is an important factor that should not be ignored. Based on the understanding of the rapid development of China’s rail transit industry, this paper takes the rolling stock business background as the research boundary to explore the influencing factors of software failure regarding different business characteristics of this industry and other industries. In addition, most of the research on software risk in the existing literature has focused on the identification and analysis of risk but has not paid attention to the unique high-risk factors of software development: demand and personnel. Existing studies have mainly explored the influencing factors from the perspective of software success and there are few studies on the influencing factors from the perspective of software failure and even fewer in-depth studies on the action mechanism of influencing factors.

Therefore, combined with the author’s own practical experience, this paper aims to conduct exploratory research in this area. The main research contents are the following:

(1) This paper studies the failure of on-board PIS software in the field of rail transit in China. When judging the failure of a project, its success standard needs to be refined

further, especially in combination with research on the industry characteristics. From the existing literature, it is apparent that research on the success standard of a project is often applicable to the whole software development project and even the information project. In practice, the success standards of software development projects vary with their type, scale and level of technical content. Therefore, based on the background of China's rail transit industry and the investigation of the current situation of on-board PIS software development, the influencing factors leading to PIS software failure are defined and identified through semi-structured interviews with employees of Company A.

(2) Combined with the questionnaire survey and literature review, the influencing factors are studied systematically. In the past, research on project failure factors has often started from the positive side, and the research samples are frequently all software projects, including projects with different degrees of success and failure and projects with different scales, without considering failure problems from the opposite side, i.e., selection of failure projects as research samples, on which this discussion can better reflect the pertinence of the sample.

(3) There is an interactive relationship between the factors of project failure, but previous studies on the action mechanism of the influencing factors of PIS software failure have not considered the influence of this relationship. For example, customers do not participate directly in the project, but they can affect the final project results through links such as the project manager, project team and technology selection.

(4) A structural equation model is used to study the action path of influencing factors. In terms of research methods, previous studies have mainly focused on simple descriptive statistics. For example, the greater the frequency, the more important the factor is. Although descriptive statistics are relatively straightforward, they can only explain surface phenomena and lack an in-depth analysis of the problem. Furthermore, although some articles have performed factor analysis, the factors were independent of each other and could not explain their mutual influence relationship. Therefore, there is a lack of

effective solutions to improve software success to reduce the failure rate of subsequent software development projects.

2.5 Summary

Starting with a general introduction of the PIS research literatures with some failure cases as a supplement to the insufficient research literature available in the specific field of rail transit PIS and an analysis of PIS software development characteristics, this paper reviews the relevant research on software project success/failure from four aspects: software project success criteria/standard, software project failure viewpoints, software project failure and influence and software project evaluation method. The author then summarizes four aspects of the influencing factors of software project failure: the reasons for the high failure rate of software projects, the factors affecting the results of software projects, the factors affecting software failure and the factors affecting the software implementation process. Finally, the paper comments on the literature and proposes the starting point of the research.

Chapter 3. Research Design and Research Methods

Information system development (ISD) projects own common characteristics of high complexity, uncertainty, changeability, invisibility and a high failure rate (Taipalus et al., 2020) and, as a specific IS application in the field of rail transit, on-board PISs inevitably face challenging software development failures. It is vital to study the impact mechanism of PIS software failure in relation to the rapid development of China's rail industry. This chapter introduces the research design and research methods in detail. Firstly, combined with the research ideas, this paper presents the technical path of the research and then analyses the research methods.

3.1 Research Design

An on-board PIS mainly provides passengers with operation management information, such as the train departure time, arrival time and passenger instructions, as well as releasing public information, such as news reports, weather forecasts, entertainment highlights, advertisements and live events. In the case of natural disasters, terrorist attacks and other emergencies, a PIS can provide dynamic emergency evacuation tips, information guidance and other contents so that passengers can take rail transit trains safely and conveniently. Identifying and monitoring the relevant factors that may lead to PIS software failure from the full life cycle of PIS software is the core of PIS software project management. Revealing and quantifying the influencing factors of PIS software failure and clarifying the action mechanism of these influencing factors on software failure are essential to the realization of PIS software project objectives.

3.1.1 Research Ideas

The research on software project failure can be traced back to 1975 (Lucas, 1975); McLeod and MacDonell (2011) reviewed the research results concerning factors affecting the output of software system development projects, providing a good reference for research on software project failure. Lehtinen et al. (2014) studied the influence relationship between factors using root cause analysis. Stoica and Brouse

(2013) pointed out that most of the existing studies used second-hand data or questionnaires to identify failure factors, but failure is still a common phenomenon in reality. They believed that the existing studies have defects in the identification and exposure of failure factors. Grounded in Luhmann's social systems theory (Luhmann, 1993), they suggested a four-stage adaptive multi-model method, which comprehensively applies grounded theory, a literature review, social theory, adaptive theory and so on. In their follow-up study, adaptive and pre-emptive IT theory (AdaPIT Theory) was proposed (Stoica and Brouse, 2014). It can be seen that the existing studies have mainly focused on empirical research methods and the widely used literature review, case analysis and questionnaire survey methods, but few empirical studies have focused on the failure of PIS software in the field of rail transit. Therefore, this paper uses the research methods of Stoica and Brouse (2013) for reference and carries out research based on the background of China's rail transit. Firstly, through semi-structured interviews with employees of Company A, this paper analyses the problems, causes and influencing factors in the practical development of rail transit PIS software. Based on this method, combined with the influencing factors found in the existing literature, an influencing factor system of PIS software failure is established and then the influencing factors are quantitatively analysed by the Decision Making Trial and Evaluation Laboratory (DEMATEL) method to identify the cause factors and result factors for exploration of the causal relationship. Next, the paper builds the PIS software failure analysis framework, puts forward theoretical assumptions, forms the theoretical model of the impact mechanism of PIS software failure in China rail transit, conducts scale development and analysis and finally discusses the empirical analysis and results.

3.1.2 Technical Route

The research path of this paper is formed according to the research ideas and corresponding research methods. The specific technical route is shown in Figure 3-1.

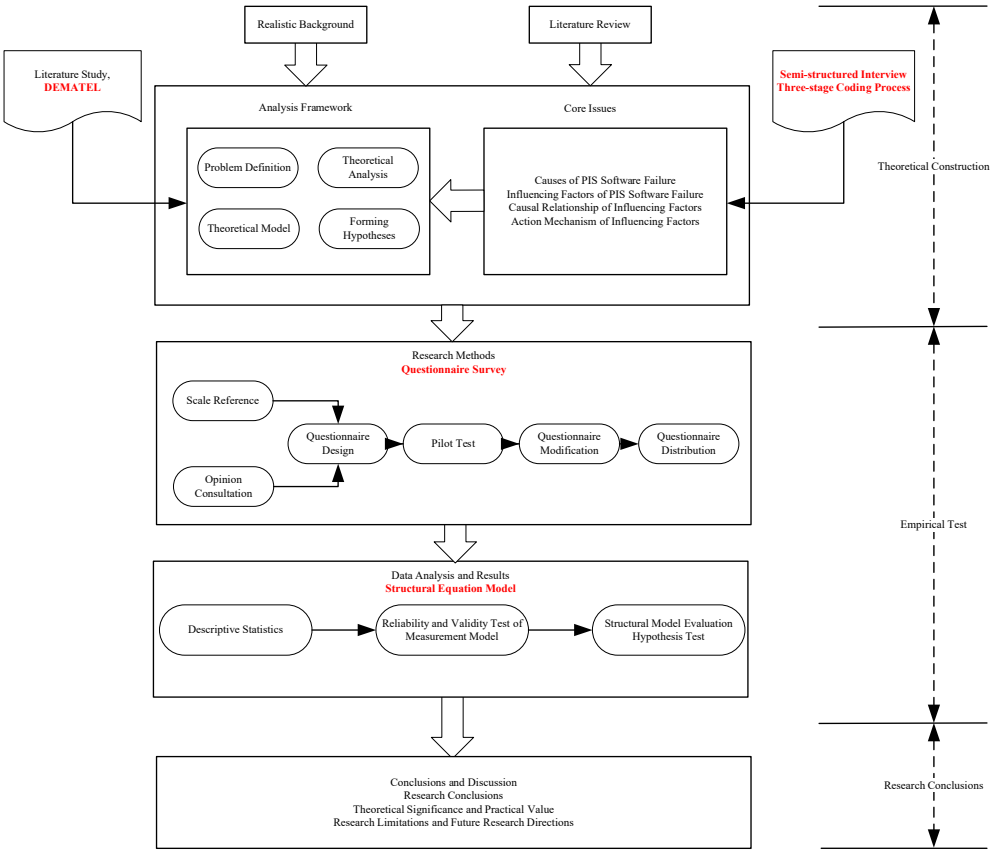


Figure 3-1 Technical Route of the Research

3.2 Research Methods

3.2.1 Semi-structured Interviews

Interviewing is important if the aim is to determine what certain targeted people think, how an event or even events are interpreted or what people have done or are planning to do (Aberbach and Rockman, 2002). The semi-structured approach is preferred as it combines structured and unstructured formats, and semi-structured interviews on the one side enable the participants to concentrate on the interview questions and on the other side leave them enough space to present their attitudes, feelings and perceptions about the topic (Simão Filho et al., 2017). Semi-structured interviews are most often used for exploratory investigation. This paper adopts this method to investigate software development problems with the personnel of various departments in the PIS software development process of Company A, including the development project director,

manager, developer, maintenance engineer and quality management engineers, obtain corresponding information and data and identify the problems and influencing factors of PIS software failure in Company A.

In February 2021, the author conducted formal interviews with a total of 16 members of the project teams of different customer divisions of Company A. Among them, according to the regional differences of the project team, some project team members who reside in Beijing or Shanghai participated in telephone interviews and most project team members attended face-to-face interviews in Changzhou. The project team members interviewed covered the main work contents of the software development project, including demand analysis, architecture development, programming, debugging and maintenance.

Semi-structured interviews can not only allow the interviewees' conversation to focus on the research problems but also make the conversation relatively flexible and identify the relevant problems. During the interviews, I created a relaxed communication atmosphere so that the interviewees could fully express their views and I could have more in-depth communication with them. In different project groups, each project group selected two to five members for "one-to-one" interviews. The length of each interview was about 30 minutes to 1 hour. During the interviews, the data collection mainly depends on a combination of audio, text and image recording. After the interviews, the off-site and on-site recordings were combined and transformed into written materials, totalling about 47,000 words, which are important sources that this paper uses to analyse problems and propose improvement countermeasures.

3.2.2 A Three-stage Coding Process

A three-stage coding process is applied for the inductive analysis of collected data and refers to the operational process of breaking up the collected data, identifying concepts and then putting the data together again in a new way, and it consists of three levels. The first is open coding (primary coding), which gradually conceptualizes and categorizes the data records; that is, a large number of data records are reduced level by

level according to certain principles, and the data contents are correctly reflected by concepts and categories, the process of breaking, crushing and resynthesizing the data records and abstract concepts. Next, axial coding (secondary coding) refers to the process of connecting various categories derived from open coding through searching for relationships between and among these categories. Finally, aggregate, theoretical coding, or selective coding, identifies one or several core concepts (categories) with strong generalization and association ability through analysis and integrates other concepts (categories), systematically connecting them to other categories, verifying the relationship between them, supplementing the conceptualization of undeveloped categories with a complete process and finally forming a theoretical framework (Corley and Gioia, 2004).

The purpose of applying three-stage coding process in this paper is not to construct a new theory, but to analyze and deal with the primary interview data that were collected. In the traditional literature-based research on influencing factors, most scholars have shown strong subjectivity in sorting out the factors, coding and refining them more casually, and the credibility of the conclusions has often been questioned. This paper draws on data coding, follows the idea of open coding to refine the original materials and data, performs a gradual coding analysis of the content of semi-structured interviews and suggests corresponding solutions by refining and straightening out the problems, causes and influencing factors of PIS software failure in the selected case – Company A.

3.2.3 DEMATEL Method

The DEMATEL method is a systematic factor analysis method proposed by the Bottelle Research Institute of the United States in 1971 to solve complex and difficult problems in the real world. The direct influence matrix is determined through the analysis of the logical relationship between various factors in the system, and then the influence degree and affected degree of various factors on other factors are obtained through the calculation of a direct influence matrix. Next, the cause degree and centrality of each

factor are calculated, and finally the key factors affecting the problem are found. On this basis, the system structure diagram is adjusted to promote the rationalization of the system structure (Zhou Dequn, 2010).

For a PIS software development project, we can gather development, testing, management and other relevant personnel to brainstorm and collect relevant internal and external information wit the aim of determining all the elements involved in PIS software development. First, we select representative personnel to establish that the collected information is comprehensive and objective, ensuring correct analysis of the system. The specific steps are as follows:

(1) Determine the relevant influencing factor system of the research. Using the information relevant to the research in this paper, the influencing factor system is determined, and the specific factors are numbered sequentially, marked as S1, S2, S3, ..., Sn.

(2) Analyse the direct influence relationship between different factors. Judge the existence of the influence relationship between various factors through the method discussed by the expert group and determine the strength of the influence degree (it can be quantified by the method of assignment, such as strong = 3, medium = 2, weak = 1, none = 0).

(3) Establish a direct impact matrix. The direct influence relationship between the above factors is expressed by a matrix. For a system containing n factors, the n-order matrix $M = (a_{ij})_{n \times n}$ represents the direct influence relationship between elements, where a_{ij} is the data on the connection between elements S_i and S_j; that is, element S_i has a direct influence on element S_j. If $i = j$, $a_{ij} = 0$.

(4) The calculation normalization directly affects the matrix. The normalized direct impact matrix G ($G = (g_{ij})_{n \times n}$) is obtained by planning the direct impact matrix, where $0 \leq g_{ij} \leq 1$:

$$G = \frac{1}{\max\limits_{1 \leq i \leq n} \sum_{j=1}^{n} a_{ij}} M$$

(5) Determine the comprehensive impact matrix. To analyse the relationship between various factors, it is necessary to find the comprehensive influence matrix T, $T = G(I - G)^{-1}$, where "I" is the unit matrix.

(6) Calculate the influence degree f_i and the affected degree e_i of each factor. Add the elements in matrix T by rows to obtain the influence degree of corresponding elements, and add the elements in matrix T by columns to obtain the affected degree of corresponding elements.

(7) Calculate the centrality and cause degree of each factor. The centrality degree is $m_i = f_i + e_i$, i = 1, 2, ..., n, and the cause degree is $n_i = f_i - e_i$, i = 1, 2, ..., n. If the cause degree is greater than 0, it indicates that this element has a great impact on other elements and becomes a cause factor; If the cause degree is less than 0, it is the result factor.

(8) Make suggestions. Through the above calculation, the mutual influence relationship between the factors and the degree of influence on the whole system can be judged according to the influence degree and affected degree of each factor, and then the importance of each factor in the influencing factor system can be determined according to the centrality of each factor and the position of each factor in the system can be determined according to the size of the cause degree. In this way, a number of elements can be deleted and the complexity of the relationship between elements can be simplified according to the above quantitative relationship, enabling suggestions to be made.

3.2.4 Questionnaire Survey

A questionnaire survey is a common tool to collect data in social survey and research activities. Using questions related to the research objectives, a detailed questionnaire design is implemented, and the respondents are required to answer accordingly to collect data.

To clarify the influencing factors of on-board PIS software failure in China's rail transit, the importance of factors and the analysis framework of factor action paths, the

questionnaire was designed. In the questionnaire design, the issues relevant to this study were communicated and exchanged with the middle and senior managers with rich development experience of software-related enterprises. They made many suggestions on software project and structural equation modelling, which provided timely solutions to some theoretical and practical problems in this research and ensured the robustness, validity and operability of the research.

In addition, the validity of the questionnaire can be controlled by selecting appropriate samples and objects according to the concept definition of the research object in the paper. The collected questionnaires were analysed using professional statistical software and the structural equation modelling method. The main analysis methods used include descriptive statistics, regression analysis and PLS, reflecting the validity and preciseness of the research.

3.2.5 Structural Equation Model

3.2.5.1 Introduction

Structural equation modeling (SEM) is a statistical modelling technology that uses a linear equation system to express the relationship either between observed and latent variables or between latent variables. It is widely used in management, psychology, pedagogy, sociology, behavioural science and other fields. Multiple regression, factor analysis and path analysis are special cases of structural equation modelling. A structural equation model is solved through simultaneous equations, but it has no strict hypothetical constraints and “allows measurement errors in the independent variables and dependent variables” (Sun et al., 2013). More and more papers using a structural equation model as a data processing method have appeared in China’s mainstream academic journals of management, and “SEM software tools, such as LISREL, AMOS, EQS, Mplus and so on” (Wanstrom, 2002), are increasingly being used by scholars. Compared with traditional multiple regression analysis, a structural equation model has several advantages (Suhartanto et al., 2020, Lejla and Nijaz, 2020), including the following:

(1) There are no strict hypothetical constraints, and measurement errors of the independent variables and dependent variables in the regression equation are allowed. The traditional multiple regression analysis assumes that either the independent variable or the dependent variable is measured accurately, while the dependent variable and independent variable are difficult to observe directly in relevant research in the field of management, usually being accompanied by measurement error. Therefore, the structural equation model enhances researchers' ability to address practical research problems.

(2) The ability to handle more complex models. Traditional multi-variate statistical analysis methods can only deal with one dependent variable in regression analysis, and can only provide direct effects between variables rather than showing possible indirect effects. However, a structural equation model can deal with multiple dependent variables and multiple independent variables as well as an iterative or recursive relationship between dependent variables and independent variables. Moreover, it can show the indirect effects between variables graphically, which greatly improves the complexity of the regression model, enabling scholars to study more complex models, to build new theories and to promote the original theories.

(3) A measurement model and structural model can be processed simultaneously. The traditional multi-variate statistical analysis method separates the measurement of latent variables from the treatment of the relationship between latent variables. First, measure the latent variables and evaluate the reliability and validity of the measurement model of latent variables. Only after passing the evaluation criteria can the data be used for further analysis. In the structural equation model, the measurement model and the structural model are analysed and processed at the same time, and the overall fitting index of the structural equation model, as well as the reliability and validity index of the measurement model, can be given.

(4) A high degree of correlation between the observed variables is required. In traditional multiple regression analysis, any multi-collinearity between variables is a

difficult problem that cannot be solved. In additional multiple regression, any multi-collinearity between variables will lead to an incorrect theory being tested. Although the multi-collinearity between variables can be partially eliminated through the centralized processing of data, it cannot be eliminated entirely, causing great difficulties to the theory construction and model verification. The structural equation model requires a high correlation between the observed variables of the latent variables (e.g., internal consistency greater than 0.7), which makes the multi-collinearity of the observed variables one of the indicators to judge the quality of the measurement model. Therefore, the structural equation model fundamentally solves the problem of variables' multi-collinearity, so scholars can avoid making type I errors in the process of theoretical development.

Based on the above analysis, this paper selects SEM as its data analysis method. The fitting process of the structural equation model usually includes four main steps:

(1) Model construction: To set the initial theoretical model of hypotheses according to relevant theories or previous research results. It consists mainly of ① the relationship between observed variables (i.e. indicators, usually topics) and latent variables (i.e. factors, usually concepts); ② the relationship between latent variables; and ③ in complex models, the values or relationships of parameters such as the factor load or factor correlation coefficient can be limited.

(2) Model fitting: After adopting a new model, we must try to find its solution, in which the primary concern is the estimation of the model parameters. In some cases, because the model is wrongly set, its parameters cannot be identified and the unique estimated value cannot be obtained, so the model has no solution. After judging that the model can be identified, the model parameters are estimated. In structural equation model analysis, the goal is to find parameters to minimize the "gap" between the implied covariance matrix of the model and the covariance matrix of the sample. There are many different definitions of the gap between these matrixes, resulting in different model fitting methods and corresponding parameter estimations. The most used model estimation

methods are maximum likelihood (ML) and generalized least square (GLS).

(3) Model evaluation: After obtaining the parameter estimation value, it is necessary to evaluate whether the model fits the data and compare it with the fitting index of the alternative model.

(4) Model correction: If the model cannot fit the data well, it needs to be corrected and set again. In this case, researchers need to decide how to delete, add or modify parameters of the model. The fitting degree of the model can be improved by resetting the model.

The above steps constitute the basic work of applying a structural equation model to study a theoretical model.

3.2.5.2 Determination of the Sample Size

The sample size requirements of SEM should be considered in relation to at least two aspects: statistical precision and statistical power. Statistical precision generally refers to the confidence interval width of the parameter estimator. Statistical power refers to the probability of rejecting the null hypothesis when the alternative hypothesis is true in the significance test. Therefore, a consistent method that can accurately calculate the minimum number of samples required for SEM does not exist.

Kline pointed out that the sample size requirements of SEM are affected by the following four factors: (1) the more complex the model is, the more parameters need to be estimated, and the bigger the sample size requirement is; (2) the observed variables are continuous and conform to normal distribution, the variables are linear, there is no interaction model, and the sample size demand is small; (3) the lower the reliability of the scale, the bigger the sample size demanded; the more measurement indicators of latent variables are used, the smaller the sample size required; and the more missing data in the questionnaire, the bigger the sample demand; (4) some kinds of structural equation models have requirements for a special sample size. In factor analysis, for example, fewer indicators per factor may require larger samples. The factors explain unequal proportions of the variance across different measurement indicators; some

measurement indicators with obvious covariance with multiple factors, an increase in the number of factors and low covariance between factors will all increase the necessary sample size. (Kline, 2016)

Although the sample size required for SEM is affected by many factors, there are still simple rules of thumb. For a structural equation model with continuous data and normal distribution estimated using the maximum likelihood method, Jackson (2003) believed that the N:q rule can be used to determine roughly the number of samples required. N is the number of samples, and q is the number of parameters to be estimated in the model. The recommended ratio is 20:1, which can also be relaxed to 10:1. Jackson (2003) did not point out that a smaller proportion is unacceptable, but as the proportion decreases (e.g., 5:1), the credibility of the model operation results will also decrease. Bentler and Chou (1987) stated that the ratio of the number of samples to the estimated parameters must be at least 5:1 to ensure the credibility of the estimated value of the parameters and at least 10:1 to ensure the effectiveness of the significance test. Therefore, both studies suggested that 10:1 is the ratio of comparative insurance. Chin et al. (2003) gave the evaluation standard of the SEM sample size based on variance, and it is acceptable as long as the sample size is bigger than any one of the following conditions: ① up to 10 times the observed variable constituting the latent variable; and ② 10 times the number of latent variable paths with the largest number of structural paths.

For papers using SEM as the analysis method, Barrett (2007) suggested that reviewers should reject articles with a sample size lower than 200 except for the strict number limit of the research population. This suggestion is not an absolute standard, but it shows that SEM needs a relatively large amount of sample data. The parameters to be estimated using SEM are about twice the number of questions in the questionnaire. Based on the 5:1 ratio given by Bentler and Chou (1987), it is suggested that the sample size should be bigger than 10 times the number of questions in the questionnaire. To ensure the effectiveness of the significance test, the ratio of 10:1 is the lowest value; that is, the number of samples should be at least 10 times greater than the number of

questions in the questionnaire. To be rigorous, researchers should determine the parameters to be estimated by the structural equation model before collecting the data and calculate the minimum number of samples according to the N: q rule.

Taking the research model of this paper as an example, the structural path coefficient of the latent variable with the largest structural path is 5, which requires the sample size to be greater than 50. Among the constituent latent variables, the largest number of observed variables is inadequate project management, with seven, which requires the sample size to be bigger than 70. This study obtains a total of 313 effective samples, which is much bigger than 50 or 70, meeting the sample size requirements of variance-based SEM. The SEM based on covariance requires the sample size of the survey data to be greater than five times, preferably more than 10 times, the total observation variables in the research model. The research model in this paper has 36 observation variables, and the sample size is required to be at least 180.

3.2.5.3 Comparison of Two Structural Equation Modeling Methods

According to the processing objects and estimation methods, structural equation models can be divided into covariance-based and variance-based models. The former aims to minimize the square difference of each element of the covariance matrix, applies optimal estimation algorithms such as maximum likelihood estimation (MLE) or generalized least squares (GLS), to fit the measurement model and structural model and gives the overall fit index of the model, such as the comparative fit index (CFI), residual mean square root (RMR) and so on. The corresponding analysis software includes LISREL, AMOS, EQS and so on. The variance-based structural equation model aims to maximize the explanatory variance of the independent variables to the dependent variables and uses the partial least square (PLS) method to fit the measurement model and structural model, but it does not give an overall fit index of the model. The corresponding analysis software consists of PLS-Graph, PKS-GUI, SmartPLS, LVPLS and so on. Although the variance-based SEM cannot provide an overall fit index of the research model, it is better than the covariance-based SEM in terms of data distribution

requirements, measurement model types that can be processed, model complexity and sample size requirements. The covariance-based SEM requires the survey data to conform to multi-variate normal distribution; otherwise, it will be difficult to obtain the overall fit index that satisfies the requirements. With the questionnaire data of most information systems, it is difficult to meet the requirements of normal distribution, while the SEM based on variance has no strict requirements for the distribution of survey data, so it is more suitable for empirical research on information systems. The variance-based SEM can deal with both reflective measurement models and formative measurement models, while the covariance-based SEM can only deal with reflective measurements and cannot deal directly with formative measurements. If the measurement model is wrongly set, it will lead to an error in the structural model, resulting in a type I or type II error in the theoretical construction. Furthermore, the variance-based SEM requires a smaller sample size than the covariance-based SEM.

Based on the above analysis, the data analysis method selected in this paper is the SEM based on variance, and SmartPLS is used for data analysis. The structural equation model analysis process used to obtain a high-quality structural equation model fit effect is shown in Figure 3-2.

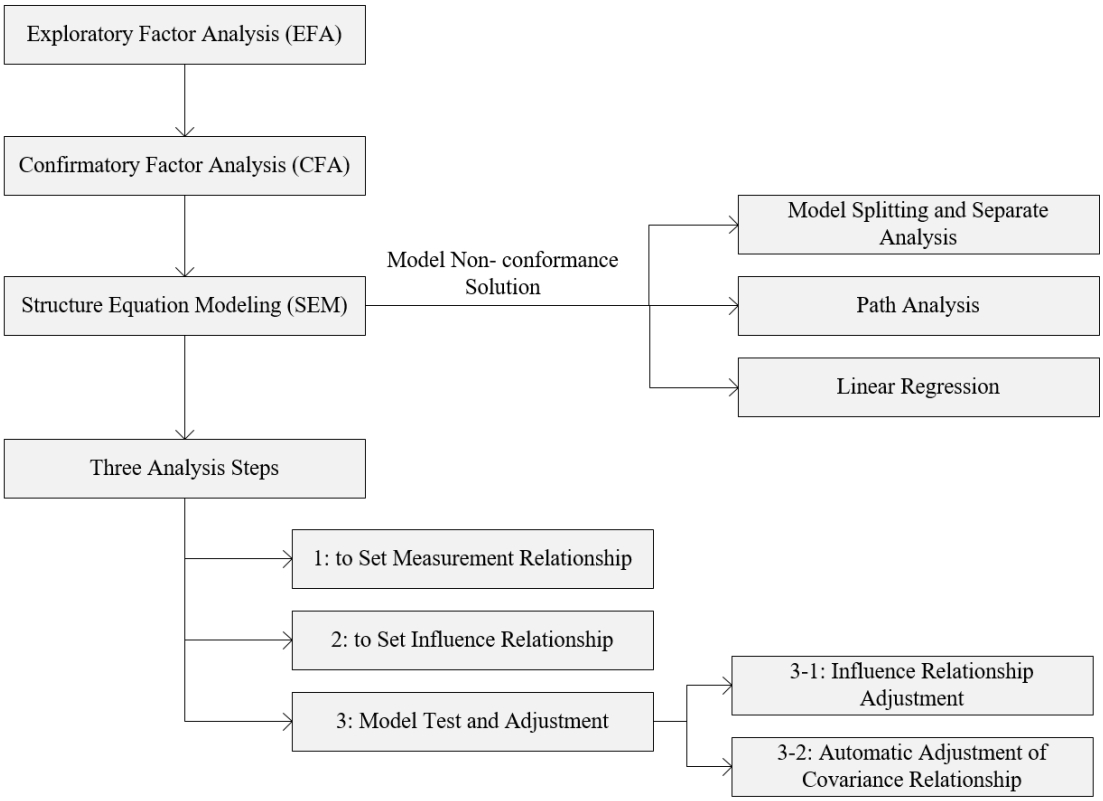


Figure 3-2 Analysis Flow of the Structural Equation Model

3.3 Summary

This chapter is the part of research design and research method selection, which mainly involves research ideas, technical route and research methods of PIS software failure. Because the research design is the basis for the selection of research methods, whether the research methods are properly selected directly affects the reliability and credibility of the research results of this paper. This chapter first gives the research ideas of PIS software failure according to which this paper puts forward the research technical route, and then explains the use of research methods combined with the research contents.

Chapter 4. Identification and Analysis of the Influencing Factors of PIS Software Failure

This chapter identifies the influencing factors of PIS software failure through qualitative analysis of the contents of the semi-structured interviews conducted with Company A. Then, combined with the research results in the existing literature and the questionnaire survey of industry experts, the relevant influencing factors are obtained and then the influencing factor system of software failure is constructed. Finally, using the DEMATEL method, a quantitative analysis of the influencing factors is performed and the causal relationship between the factors is identified.

4.1 Identification of the Influencing Factors of PIS Software Failure Based on Company A

Company A, a joint venture between the KTK Group from China and the TLV Group from Belgium, is a high-tech enterprise specializing in the design, manufacturing and maintenance of PISs. KTK is the global number one interiors manufacturer and the first company to provide on-board PISs for China's rail transit. TLV is the leading PIS supplier in Europe and the USA, with cutting-edge knowhow. Since its founding in 2004, Company A has become one of the three PIS manufacturers in the Chinese high-speed railway industry and has covered 100% of the market shares in CRH1. It also takes an active part in urban rail transit and holds dozens of metro lines in Shanghai, Shenzhen, Guangzhou, Shenyang and Changzhou at home and in Queensland, Cairo, Bombay and so on abroad. Company A has established a long-term strategic cooperation with companies such as CRRC, ALSTOM (Bombardier), Siemens and ROTEM.

Different from its competitors in China, which only perform system integration, Company A carries out in-house design and in-house manufacturing, which enable it to accumulate rich experience in R&D, especially in software development, and it has experienced success and failure in both. Therefore, the research on software failure has

strong practical significance.

4.1.1 Implementation of Semi-structured Interview

(1) Selection of interviewees

The author, as the General Manager of Company A, has been engaged in the on-board PIS business in the rail transit industry since 2004 and is mainly responsible for the overall management of the company's market, R&D, quality and operation. During his daily work, in addition to close cooperation and communication with his colleagues in the project team, he also exchanges information and communicates with other project team members.

The interviewees have different positions and experiences, ranging from project manager and software engineer to quality manager and software maintenance engineer. The 16 respondents are engaged in projects involving high-speed railway, subway, door controller and other projects. Due to the male-dominated characteristics of the software development industry, especially in the field of rail transit, the interviewees are all male, and they are from the Marketing Department (4), R&D Department (9), Quality Control Department (2) and Operation Department (1). Detailed information on the interview group and interviewees is shown in Table 4-1 below:

Group	Member	Department	Project Task Type	Task Description	Position	Working Experience (Year)
A	A1 Ni	Marketing Depart.	Project management	Project demand docking, schedule management, cost management	Project manager	14
	A2 Tao	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	11
	A3 Song	R&D	Software	Develop software	Software	9

		Depart.	development	according to the requirements of the project manager	engineer	
	A4 Sun	QC Depart.	Quality control	Analysis and handling of customer quality complaints	Quality engineer	7
	A5 Gong	Operation Depart.	After-sales maintenance	Defective product maintenance	Maintenance engineer	7
B	B1 Yu	Marketing Depart.	Project management	Project demand docking, schedule management, cost management	Project manager	11
	B2 Zhao	Marketing Depart.	Project management	Project demand docking, schedule management, cost management	Project manager	10
	B3 Zhang	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	8
	B4 Feng	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	11
	B5 Zhang	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	5

	B6 Wang	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	4
	B7 He	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	3
	B8 Zhang	QC Depart.	Quality control	Overall quality control of the company	Director of quality control department	12
C	C1 Tong	Marketing Depart.	Project management	Project demand docking, schedule management, cost management	Project manager	5
	C2 Cheng	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	4
	C3 Luo	R&D Depart.	Software development	Develop software according to the requirements of the project manager	Software engineer	4

Table 4-1 Detailed Information on the Interview Group and Interviewees

(2) Design and implementation of the interview outline

The interviews have the following objectives: 1) to identify the problems leading to the failure of PIS software in Company A and analyse the causes; 2) to identify the factors affecting the failure of PIS software in Company A and determine the key factors; and 3) to construct an influencing factor model of PIS software failure. The interview outline is

designed accordingly. Su Jingqin believed that the design of the interview outline should establish the element dimensions, then refine the element dimensions to obtain the sub-dimensions of each element and subsequently explore the relationship between each element dimension and each sub-dimension (Jingqin SU and Miao CUI, 2011). However, to avoid a guiding influence on the respondents and ensure the interview effect, it is necessary to use open-ended questions for the interviews (Pallant, 2016), and the number of questions should be between two and eight as far as possible to avoid boredom among the respondents and avoid affecting the quality of the interview data. Researchers must extract answers for the measurement of factors and the measurement of the relationship between factors from the responses obtained. Finally, the interview outline of this paper consists of eight questions, as shown in Appendix 3-1.

In February 2021, formal interviews were performed with a total of 16 members of the project teams from different customer divisions of Company A. After the interviews, the off-site recording and on-site recording were combined and transformed into written materials totaling about 47,000 words.

4.1.2 The three-stage Coding Process of the Interview Contents

Under the guidance of the coding process, Nvivo12 is used to encode the interview contents. The process of data analysis is divided into three hierarchical nodes with subordinate relationship.

(1) Open Coding

The initial concept is formed by encoding the interview materials line by line and word by word. After coding 16 interview records word by word and sentence by sentence, the similarities and differences between various concepts are compared, and the initial concepts appearing less than twice are eliminated. Finally, 484 initial concepts are formed from the original materials as bottom analysis nodes (Table 4-2).

Categories	Some Extracts from Original Statements (Initial Concept)
Disorderly Software Download Mode	A5: The download methods of software are not unified, and everyone's habits are different ... Some people like to pack together, while others like to download one by

	one. The more complex it is, the easier it will be to make mistakes after sales. And it will be easy to get report faults and problems. B4: All of our software is downloaded in different ways. There are legacy problems ... A large amount of software was chaotic, and some was lost, just before PDM was applied for documentation.
Weak Technical Support	A2: There are two reasons. One is that the production of the products was stopped. Maybe it was in the middle and late stages when we selected them. Considering that they are stable and reliable, the service life may be relatively short; there are many unsolvable problems in the previous project, which can only be changed in the next project. A3: What I want to say is related to time, including the development cycle and test cycle, which are not only limited in our company but also in the carriage. Especially in the later stage, when our platform is complete, all the functions we should do are completed at home. There is no problem. When we need to get on the bus for debugging, it is difficult to control the time, and the efficiency on site is very low. A4: If you modify or add new functions to other people's code, you will spend more time becoming familiar with the code than writing new code yourself. If you modify or add new functions to other people's code, you will spend more time familiarizing yourself with the code than writing new code yourself.
Inadequate Project Management	A1: We had no experience before. Now, when we decide that it is a Chinese character, we cut off two bytes in advance to avoid splicing errors and making jokes. The customer said you were fast or slow. A4: I have never participated in the inter-departmental review ... What's more, we may not have fully identified some customers' previous needs, and some requirements may not have been fully identified. B2: The current software control is still relatively poor. There may have been a release process in the past, which is to go through a formal release procedure after verification on the platform, right? To send the program... This test condition, right? After adding

	some function, you can only go to the site to test them. Second, there are a lot of such costs that the customers won't pay, will they? It causes costs and normally two engineers are required for the site work.
Insufficient Customer Support	A1: The demand is not clear. For common things, such as the display scrolling, the customer requires a good visual effect, but there is no quantitative value. I don't know how many times a second. A value may be adjusted according to my experience. The customer says you are fast or slow. When demands cannot be quantified, it is a great challenge. B2: For example, the display interface of the Changzhou Line 1 dynamic map and the display interface of the TV set are different because everyone's aesthetics are different. Our company produced three sets of schemes at that time, and then the customer has different leaders. Maybe the first leader finished reading the scheme; he said it was OK. And then it was read by a higher-level leader. He said the scheme was too ugly and had to be changed again. B3: We must have a simple understanding of our whole PIS system and some PIS related systems. In this way, your program, including your program framework, will be considered more, and the later program expansion will be better. C1: Sometimes the customer's business changes so fast that the developed products or functions fail in a short time ... There are too many factors affecting the verification on site, because the environment is more changeable on site. D2: Customers often complain that they don't want to see all kinds of information irrelevant to their business in an interface of the system, because it will increase their visual fatigue, but customers of other business departments must keep this information.
Company Employee Characteristics	A1: And to modify other people's codes. Some engineers leave their jobs. It may take a long time to become familiar with their codes. A4: Engineers write the software with a strong personality; that is, the software written by each engineer is not necessarily the same ... Each engineer works in his own way, and there is no core. You can coordinate with everyone to see where the problem is or

	whose problem I am judging. B7: In compiling and decoding software, it is impossible to avoid mistakes when a person writes a function and tens of thousands of lines of code. B8: Our company is in Changzhou. First of all, it has something to do with the staff in Changzhou, because our current staff is not the top people in the industry, nor the very well-known universities, doctors or master's students with high academic qualifications. Most of them are undergraduates. Of course, some of them graduated from 211 or 985 universities (the first level universities in China), but there is still a lack of personnel ability.

Table 4-2 Open Coding Categorization

Note: Refer to Appendix 4-1 for more detailed interview contents.

(2) Axial Coding

Through the comprehensive consideration of software failure and by a process of axial coding, the 484 initial concepts are summarized in 34 categories (i.e., micro influencing factors of software failure) (Table 4-3).

Secondary Node	Primary Node	Ref. Point
Product Characteristics	Poor maintainability	5
	Poor stability	10
	Complex and quickly updated Software	19
	Disorder download mode	4
Imperfect Management System	Insufficient technology outsourcing	11
	Lack of product planning	19
	Lack of effective knowledge management	13
	Poor employee incentive mechanism	10
Employee Characteristics	The responsibility for work is quite miscellaneous	11
	Different levels of developers	13
	Lack of core talents	8
	Personnel mobility	7

	Software development habits	27
Weak Technical Support	Insufficient testing	55
	Test environment not allowed	9
	Insufficient technical capacity	14
	Poor software development process	32
	Hardware changes from time to time	28
Insufficient Customer Support	Customer's leaders disagree	2
	Change in customer requirements	28
	Unclear customer requirements	23
	Complex site environment	18
Role of Company Leader	Harmonious relationship between leaders and employees	3
	Leadership support	9
	Project manager's orders	4
Inadequate Project Management	Poor product control	16
	Poor cost control	3
	Inadequate internal communication	38
	Lack of effective communication with customers	21
	Lack of project experience	11
	Unclear software requirements and objectives	7
	Inadequate prior risk review	6

Table 4-3 Node System of Influencing Factors on PIS Software Failure

(3) Selective Coding

Based on the analysis of the relationship between the main category and the other categories, the 34 categories are further abstracted and refined into seven main categories: product characteristics, weak technical support, leadership response behaviour, imperfect company management system, company employee characteristics, inadequate project management and insufficient customer support. In terms of reference

point coverage, “weak technical support” consists of 138 coding reference points, accounting for 28.5% of the total reference points; “leadership response behaviour” contains 16 coding reference points, accounting for 3.3% of the total reference points; “imperfect company management system” has 53 coding reference points, accounting for 11.0% of the total reference points; “inadequate project management” consists of 102 coding reference points, representing 21.1% of the total reference points; “company employee characteristics” contains 66 coding reference points, making up 13.6% of the total reference points; “insufficient customer support” has 71 coding reference points, accounting for 14.7% of the total reference points; and “product characteristics” encompass 38 coding reference points, accounting for 7.9% of the total reference points. Product characteristics refer to the characterization of PIS software failure, including “poor product maintainability”, “poor product stability”, “complex and quickly updated product system” and “disorderly software download mode”. Other categories express the influencing factors of PIS software failure.

4.1.3 Factor Identification of PIS Software Failure

In the three-stage coding process of Nvivo12, there is direct equivalence between the coding reference points and the number of corresponding research contents. A high number of statistics means that the research content is highly relevant, indicating the main problem; on the contrary, a low number means that the attention is low and the problem is secondary. In this paper, there are 16 respondents, and the nodes with more coding reference points are relatively important influencing factors of PIS software failure.

4.1.3.1 Problems in the PIS software system

Although the methodology of Company A’s software development project was quite mature, combined with the interview results of project members, it is found that the PIS software system still faces the following main problems:

1) The software product system is complex and quickly updated

In the interview, the interviewees said that there were many problems with software

products, such as “many PIS system nodes connected”, “a great many core system versions” and “version control is not very good”.

2) Poor stability of software products

Project members reported, in the process of software development projects, problems such as “there are various bugs in the software” and “modifying a problem will bring out more unexpected problems”.

3) Poor maintainability of software products

Project members stated that problems such as “it is more difficult and error prone to modify previous software”, “the old hardware platform does not support updated software” and “long product service cycle” often occur in the process of software development projects.

4) Software product download mode is disorderly

The respondents pointed out problems such as “download methods of software are not unified” and “different download methods”.

Through the “matrix coding” of the Nvivo12 software, the respondents’ understandings of the problems in the PIS software system can be analysed and compared from the perspective of their positions and work experience. The results are shown in figures 4-1 and 4-2.

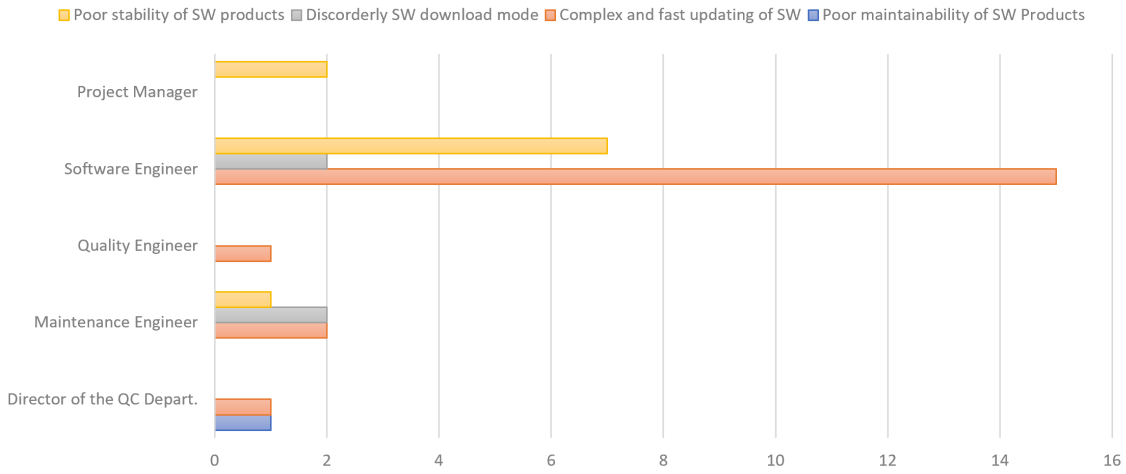


Figure 4-1 Differences in the Understanding of PIS Software Problems among Respondents in Different Positions

In Figure 4-1, the software engineer has the most comprehensive understanding of the software system and attaches the most importance to the complexity and stability of the software product itself, and the project manager pays more attention to product stability. However, the quality engineer only pays attention to the complexity of the product itself, which matches the engineer's task of analysing and handling customer quality complaints. The director of the Quality Control Department is responsible for total quality control, so he pays attention to maintainability but does not involve the other two aspects, indicating that there are still some problems in the company's quality control to a certain extent.

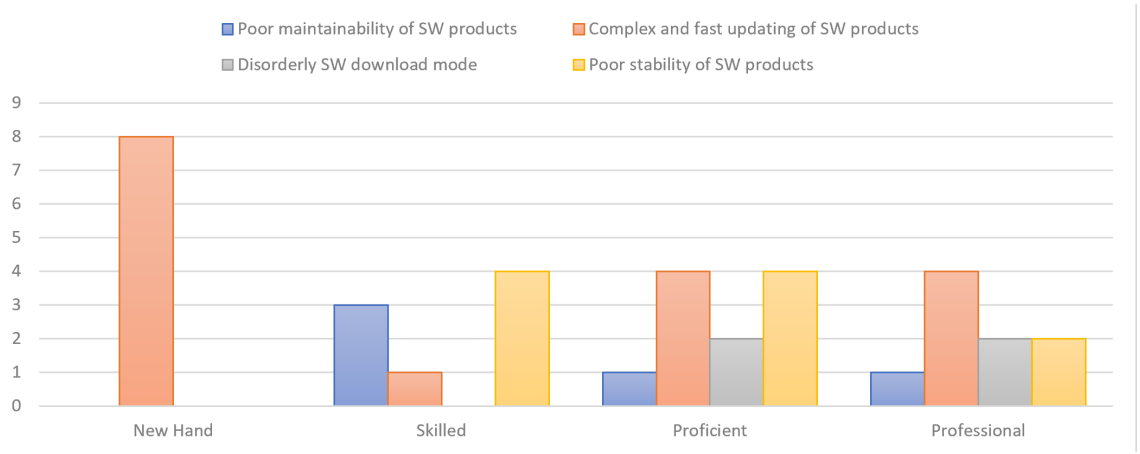


Figure 4-2 Differences in the Understanding of PIS Software Problems among Respondents with Different Work Experience

As can be seen from Figure 4-2, with the accumulation of work experience, the understanding of the PIS software system becomes more and more comprehensive. Newcomers start with the software product itself and pay more attention to the application of the software development technology and system structure; after mastering the technical aspect, the attention devoted to product stability and operability exceeds the attention paid to the product itself; and, after more than 5 years of working experience, relevant personnel gradually begin to attend to the software download mode, the importance of the product itself and its stability.

Through the operation of the "new project diagram" in the "diagram" of the Nvivo12 software, we can clearly understand which questions were raised by which respondents

and can relate them to attributes such as respondents' departments to gain a better understand of the corresponding relationship between respondents and PIS software system problems, as Figure 4-3 shows.

As can be seen from Figure 4-3, 11 interviewees from the Quality Control Department and Operation Department described the problems existing in the PIS software. Only one of the four interviewees from the Marketing Department commented on the existing problems, and seven of the nine interviewees from the R&D Department responded. At the same time, it can be seen that most respondents believe that the PIS software system is complex, is quickly updated and has poor product stability; a few people think that the disorderliness of software downloading is also a problem; and QC and R&D personnel pay attention to the maintainability of products.

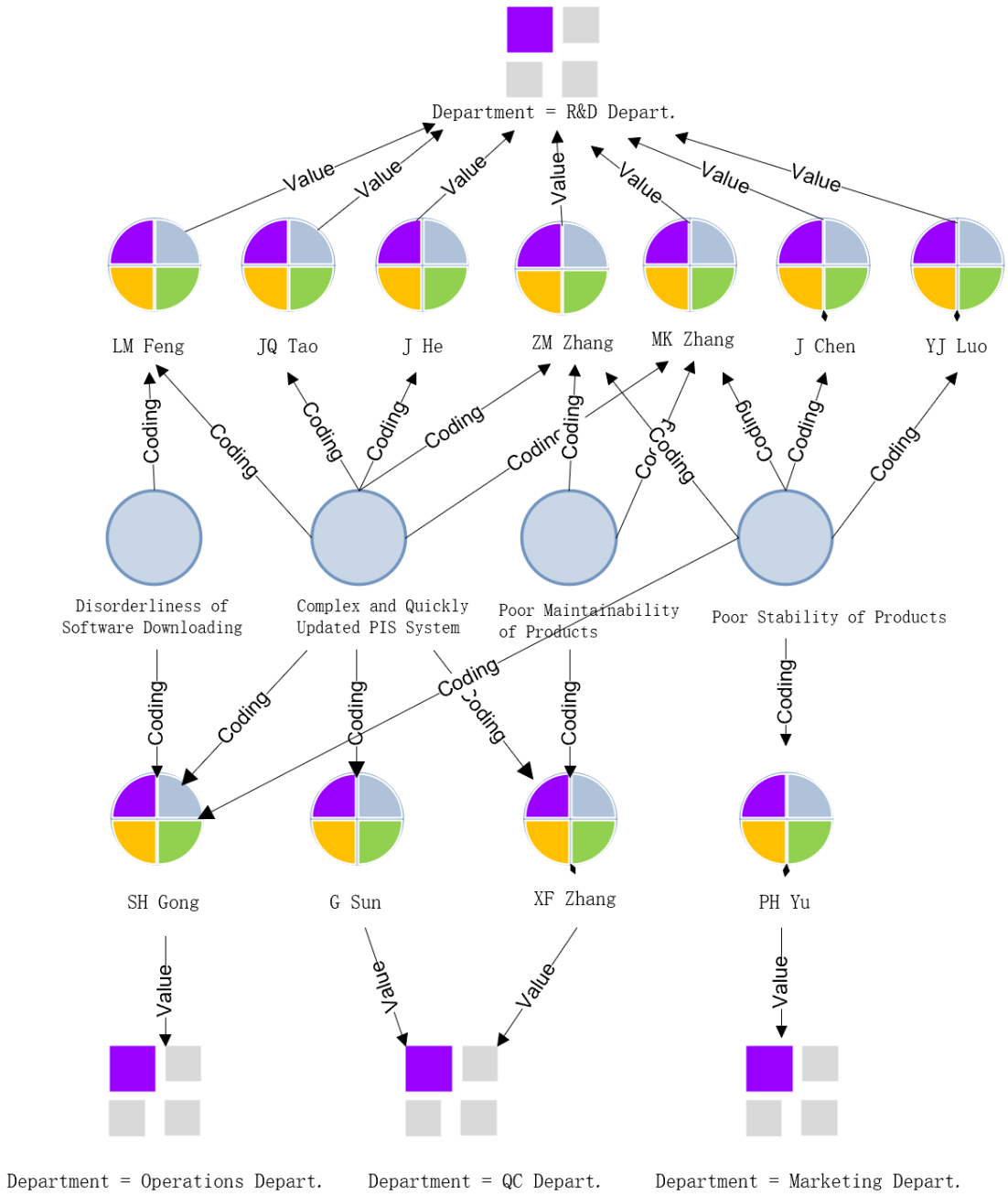


Figure 4-3 Correspondence between Respondents and PIS Software Problems

4.1.3.2 Cause Analysis of PIS Software Failure

Through the interviews, it is found that the main causes of the failure of Company A's PIS are weak technical support, inadequate project management, insufficient customer support, the characteristics of the company's employees and the imperfection of the company's management system. Specifically, they consist of the following:

1) Weak technical support

During the interviews, the respondents said that “insufficient software testing”, “the company has no on-site testing environment”, “poor software development process”, “lack of complete platform and other technical capabilities”, “frequent changes in hardware platform” and other aspects will cause software failure.

2) Inadequate project management

The respondents mentioned that inadequate project management, such as “not participating in inter departmental review”, “poor software control”, “sometimes urged by the owner”, “lack of project experience”, “insufficient demand identification”, “lack of risk assessment” and “increasingly high cost” will also affect the failure of PIS software.

3) Insufficient customer support

The respondents believed that “unclear customer demands”, “inconsistent functional results and customer expectations”, “too vague customer demands”, “customer implicit demands failed to be met”, “customer demands changing too fast”, “unclear customer input demands”, “constantly changing of customer demands”, “inconsistent opinions of customer leaders” and a “complex on-site environment” would lead to software failure.

4) Company employee characteristics

The respondents described the characteristics of the company’s employees regarding the aspects of developer’s work tasks, development ability level, development habits and personnel flow and responded that software failure is mainly reflected in “miscellaneous work”, “sufficient software development personality”, “lack of personnel ability”, “loss of personnel” and “lack of core talents”.

5) The company’s management system is not perfect

In the software failure interviews, the respondents also put forward their own views on the company’s management system. Institutional factors such as “lack of product planning”, “lack of effective knowledge management”, “lack of outsourcing”, “poor employee incentive mechanism” and “lack of employee training” will also lead to software failure.

To sum up, considering the three-stage coding process, which was supported by Nvivo12 software, the emergence of the concept of superior nodes comes from the level-by-level summary of subordinate nodes. Therefore, the research on the influencing factors of PIS software failure should not stay at the macro level of secondary nodes but should carefully analyse the micro factors making up each secondary node, helping to clarify the influencing factors of PIS software failure and their relationship. The detailed factors (primary nodes) come from the overview classification of the original concepts in the interview records of the above respondents, as shown in Table 4-4.

Secondary Node	Primary Node	Encoding Reference Point
Weak Technical Support	Insufficient testing	55
	Test environment not allowed	9
	Insufficient technical capacity	14
	Poor software development process	32
	Hardware changes from time to time	28
Inadequate Project Management	Poor product control	16
	Poor cost control	3
	Inadequate internal communication	38
	Lack of effective communication with customers	21
	Lack of project experience	11
	Unclear software requirements and objectives	7
	Inadequate prior risk review	6
Insufficient Customer Support	Customer's leaders disagree	2
	Changing customer requirements	28
	Unclear customer requirements	23
	Complex site environment	18
Company Employee Characteristics	The responsibility for work is quite miscellaneous	11
	Different levels of developers	13

	Lack of core talents	8
	Personnel mobility	7
	Software development habits	27
The Company's Management System Is Not Perfect	Insufficient technology outsourcing	11
	Lack of product planning	19
	Lack of effective knowledge management	13
	Poor employee incentive mechanism	10

Table 4-4 Node Hierarchy and Its Encoding Reference Points

As can be seen from Table 4-4, there are five secondary nodes, namely weak technical support, inadequate project management, insufficient customer support, imperfect employee characteristics and the management system of the company; there are 25 primary nodes, which are subordinate to the corresponding secondary nodes. For example, insufficient customer support is due to inconsistent opinions of customer leaders, changing customer requirements, unclear customer requirements and a complex site environment. The numbers in the table represent the number of times that the corresponding primary nodes are coded in the interview content, which is consistent with the initial concept.

In the "weak technical support" node, the coding point of "insufficient test" is much higher than the others, indicating that insufficient testing has a relatively large impact on PIS software failure. The second is "poor software development process", followed by "hardware changes from time to time", and the last are "insufficient technical capability" and "test environment not allowed". Although there are relatively few coding points for "test environment not allowed", as an important factor in software development, the support of the test environment for testing and field simulation and the impact on software development failure cannot be ignored.

Regarding "inadequate project management", "inadequate internal communication" and "lack of effective communication with customer" are the top two issues, which are related to communication, undoubtedly showing that communication needs to be

strengthened in terms of demand identification, task division and system analysis. “Inadequate prior risk review” and “poor cost control” ranked last. The review of development capability, technical risk and software scheme directly affects the software function and product stability. The interviewees pointed out that the after-sales service and travel costs will be greatly increased, which will also reduce the competitiveness of products and lead to software failure. Project managers and company executives should pay special attention to these problems.

“Insufficient customer support” basically focuses on customer needs, including “unclear requirements” and “changing requirements”; a “complex site environment” will not only affect the on-site debugging of software but also increase the company’s operating cost. If customers do not actively cooperate, it will undoubtedly affect the realization of software functions. There are few encoding reference points for “inconsistent opinions of customer leaders”, but this problem is often encountered in practice. China’s rail transit industry is dominated by large state-owned enterprises, and enterprise leaders are promoted and replaced relatively frequently, resulting in changing requirements, which act as one of the important reasons for software failure.

Regarding the “company employee characteristics”, the “software development habits” ranks first. Due to the complex structure of PIS software system, if developers do not follow a standard process but develop their own habits, they will bring an unnecessary workload to the testing and maintenance, resulting in software failure. “Different levels of developers” and “miscellaneous responsibilities” reflect the impact of personnel ability level and professional division of labour on software failure. “Lack of core talents” and “personnel flow” affect the stability of software core modules and functions, resulting in software failure.

The “imperfect management system of the company” contains “lack of product planning”, “lack of effective knowledge management”, “insufficient technology outsourcing” and “poor employee incentive mechanism”. There is little difference in the coding reference points in the interview content, indicating that these are the aspects

that the company urgently needs to improve to ensure the success of its software development.

To gain a more intuitive understanding the hierarchical position of each node in all nodes, a hierarchical analysis is carried out using the Nvivo12 software to generate a hierarchical chart, as shown in Figure 4-4.

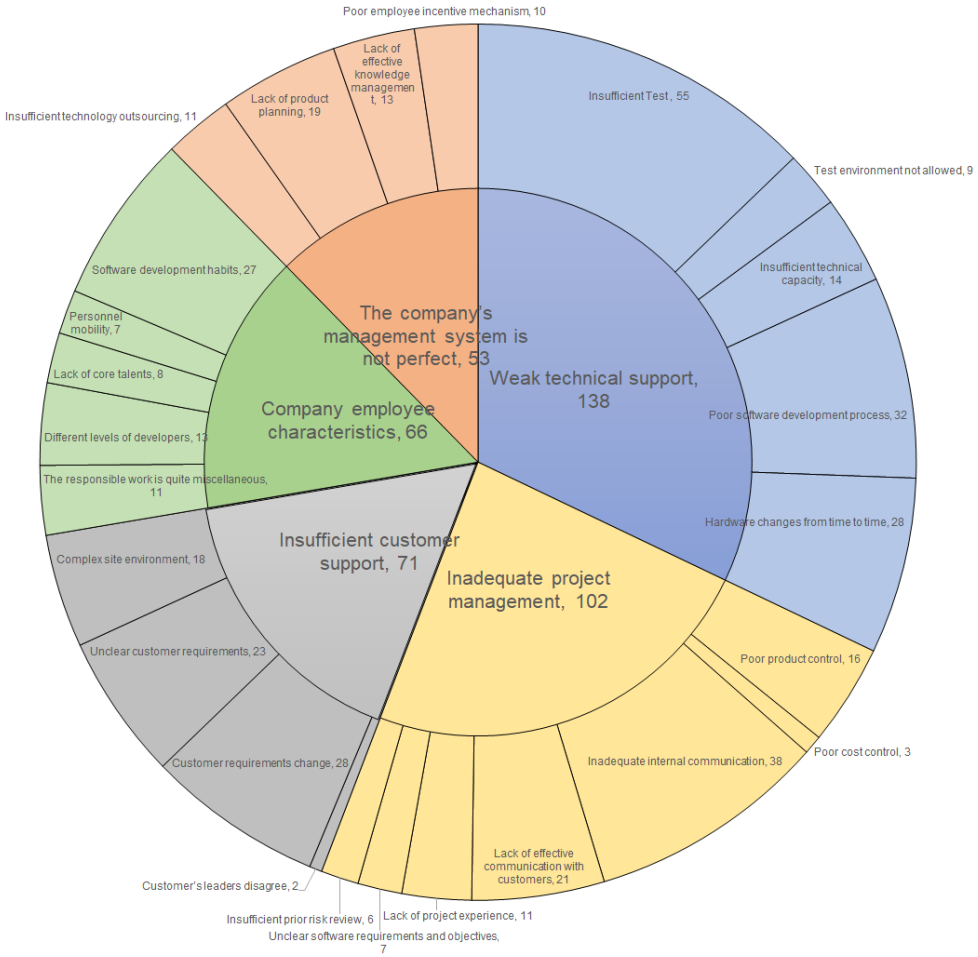


Figure 4-4 Node Coding Hierarchy

From the outside to the inside of Figure 4-4, the area between the first two circles represents the coded primary nodes (micro influencing factors of PIS software failure), and the number represents the coding reference points of the corresponding nodes, that is, the frequency in the interview content. The more reference points, the stronger the respondents' recognition of a factor and the larger the area of the graph; the area size represents the influence of this factor on PIS software failure. The area between the two internal circles is the coded secondary node (the middle-level influencing factor of PIS

software failure), which is composed of external primary nodes; the area composed of primary nodes is the resultant force of their influence, which represents the influence of the secondary nodes that they form. For all the secondary nodes in the figure, “weak technical support” accounts for the largest area, nearly one-third of the total area. “Inadequate project management” accounted for 24 per cent, “insufficient customer support” represents 17 per cent, “imperfect employee characteristics of the company” constitutes 15 per cent and “imperfect management system of the company” accounts for 12 per cent. Among the influencing factors of PIS software failure, “weak technical support” is the most important, followed by “inadequate project management”. Both are directly related to the software project itself and represent more than half of the influence, which is in line with the characteristics of strong technicality, customization and order production mode of the PIS software system. There is little difference in the weight of the other three factors. “Insufficient customer support” ranks top among the three, which also highlights the importance of customer relationship management. The other two are human resources and management systems related to the company’s own construction and development, and these are in line with the mainstream direction of modern enterprise management development.

4.1.3.3 Role of Company Leaders

In the interviews, some respondents suggested that the company’s leaders play a positive role in software development, and others suggested that the project manager plays a mandatory role in the process of project development, mainly involving a “harmonious relationship between leaders and employees”, “leadership support” and “project manager’s orders”, all of which will affect employees’ software development behaviour.

Through the “grouping” query in the Nvivo12 software, the respondents’ cognition of leadership response behavior in the process of software development is formed, as shown in Figure 4-5.

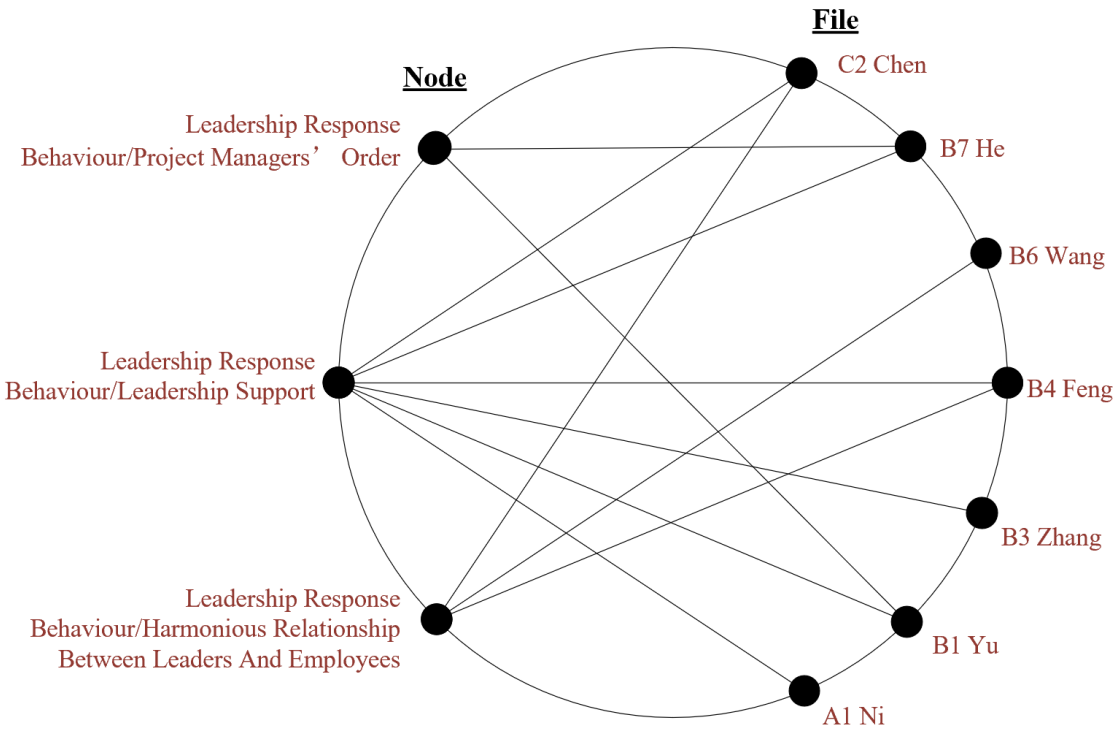


Figure 4-5 Respondents' Different Understanding of Leadership

According to Figure 4-5, only seven respondents explained the impact of leadership response behaviour on PIS software development during the interviews, and only 43.75% of the respondents responded. In the cognition of the respondents, leadership response behaviour can be divided into three types: leadership support, project manager command and a harmonious relationship between leaders and employees. The corresponding leaders affect software development through positive activity and forced intervention and by creating a harmonious relationship atmosphere.

4.1.4 Result Analysis

Semi-structured interviews were conducted with 16 employees from different departments of Company A about the failure of PIS software, and the Nvivo12 software is used to analyse the interview contents qualitatively, describe the problems existing in the current PIS software development of Company A and identify the influencing factors affecting the failure of PIS software. First, the purpose of the interviews, the design of the interview outline and the basic situation of the interviewees are briefly introduced. The positions and experience of the interviewees are different. The job span

is relatively large, from project manager and software engineer to quality manager and software maintenance engineer. Then, based on the three-stage coding process, the interview contents are coded to form 484 initial concepts and 34 categories, which are finally refined into seven main categories, specifically product characteristics, weak technical support, leadership response behaviour, imperfect company management system, company employee characteristics, inadequate project management and insufficient customer support.

Then, through qualitative research, it is found that PIS software products have four problems – a complex system and quickly updated software, poor stability, poor maintainability and a disorderly download mode – and the differences in respondents' perception of problems in their positions and work experience are compared and analysed.

The research also finds that the main root causes of PIS software failure in Company A include weak technical support, inadequate project management, insufficient customer support, imperfect employee characteristics and the management system of the company. It is also noted that the two aspects of PIS software product features, "complex and quickly updated software product system" and "disorderly software product download mode", are related to technology. Therefore, the technical factors affecting the failure of PIS software can be analysed in relation to the aspects of "insufficient testing", "poor software development process", "hardware changes from time to time", "insufficient technical ability", "test environment not allowed", "complex and quickly updated software product system" and "disorderly download mode of software products". Other organization and management activities related to technology, tasks and personnel can be attributed to customer requirements, project management and employee characteristics, including "insufficient customer support", "company employees' characteristics", "project management not in place" and "imperfect company management system".

At the same time, according to the analysis of the interview results, "leadership

response behaviour" reflects the leadership style of the company and belongs to the organization management subsystem. Some respondents believe that leaders provide positive support, some say that leaders promote projects through commands and others believe that leaders support software development by creating an atmosphere. It is not clear whether the leadership style has an impact on PIS software failure, which can be attributed to the external environment.

Therefore, it is necessary to refine the many influencing factors of PIS software failure so that managers can focus on the key ones.

4.2 Combing the Influencing Factors of PIS Software Failure Based on the Literature Review

The failure of information technology projects has been a hot topic in theoretical and practical circles for a long time (Nelson, 2007, Group, 2004). In China, the success rate of information technology projects is only 10-20% (Lu Xinyuan et al., 2006). As early as 1975, researchers tried to extract the determinants of project success and failure to increase the likelihood of achieving successful results (Lucas, 1975). However, ensuring software success remains a major challenge (McLeod and MacDonell, 2011). The following section briefly reviews the influencing factor analysis framework of software failure and then summarizes and analyses the research results of the corresponding literature.

4.2.1 Influencing Factor Analysis Framework of Software Failure

McLeod and MacDonell (2011) divide the impact of software system development into four dimensions, each of which contains many influencing elements. The four dimensions interact with each other to form an organic whole and finally affect the development process of the software system, including the outcomes of the software system (as shown in Figure 4-7).

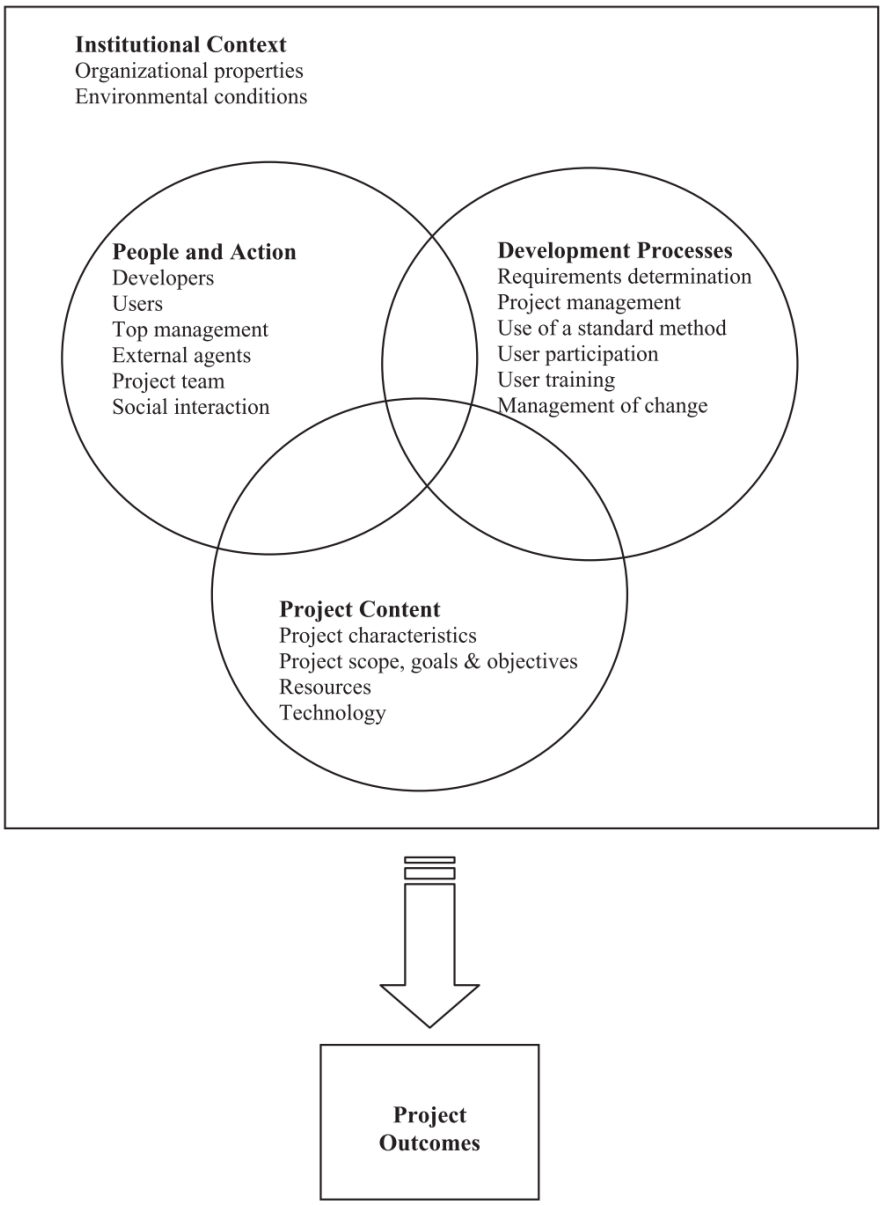


Figure 4-6 Analysis Framework of Influence Factors on Software Development Outcomes (McLeod and MacDonell, 2011)

Gupta et al. (2019) systematically summarized the current trend and future research direction of project failure and constructed a project control and evaluation framework based on I-P-O model (as shown in Figure 4-8).

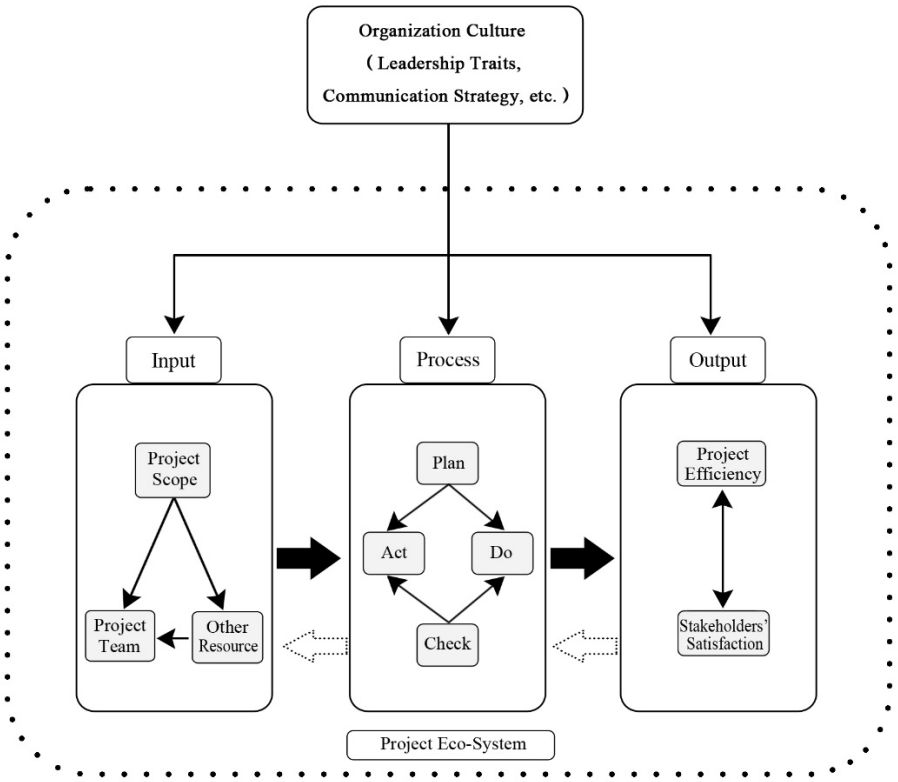


Figure 4-7 Project Monitoring and Evaluation Framework on an I-P-O Model (Gupta et al., 2019)

This framework provides good theoretical references for combing and analyzing the influencing factors of PIS software failure in this paper.

4.2.2 Main Influencing Factors of Software Project Failure

The influencing factor analysis in this paper is performed mainly through a literature review, case analysis and questionnaire survey. McLeod and MacDonell (2011) systematically reviewed the relevant research on the influencing factors of software development and deployment results from 1996 to 2006 and highlighted various influencing factors in four dimensions: the institutional context, people and action, development process and project content. Linberg (1999) discussed the views of software developers on software project failure through a case study. Savolainen et al. (2012) summarized the research on the success and failure of software development projects from the perspective of software suppliers. Jorgensen (2014) studies the influencing factors on the failure of small and medium-sized software projects in the global outsourcing market based on more than 780,000 projects and/or tasks. Through

the analysis of a binary logistic regression model, it was found that 74% of project failures can be correctly predicted during the start-up phase of the project, and customer's characteristics are considered to be almost as important as the supplier's contribution to the project failure. The risk of failure will increase with the customer's emphasis on low price and project size. Lehtinen et al. (2014) listed the common causes of software project failure in the prior studies (Figure 4-9), referring especially to the outcome summarization of the four dimensions by McLeod and MacDonell (2011), and further systematically studies the interconnection between different perceived failure causes to fill the research gap in prior studies in this field.

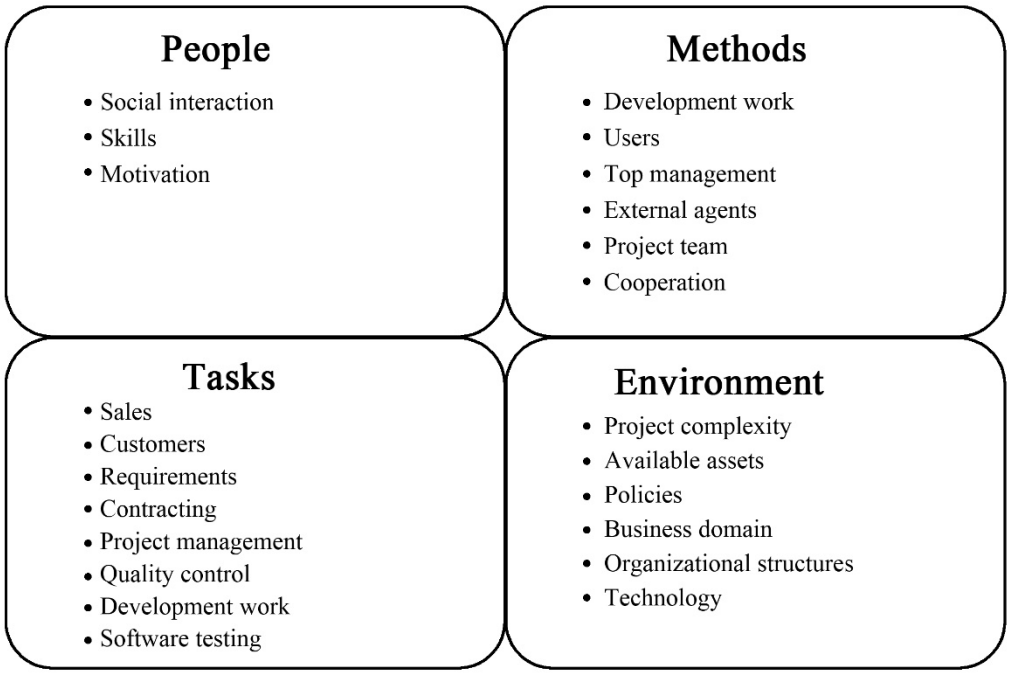


Figure 4-8 Summary of the Common Causes of Software Project Failure (Lehtinen et al., 2014)

4.3 Influencing Factor System of PIS Software Failure

Based on the above analysis, a questionnaire is designed to collect expert opinions to evaluate the potential risk impact.

4.3.1 Risk Factor Identification in the PIS Software Development Process

To facilitate the later questionnaire design, the research first identifies the possible risk factors and analyses the causes in each phase of the full life cycle of a software

development project. By reading an enormous number of articles and sorting out materials, the author collects information from experienced software project managers (project managers or supervisors, etc.) about the risks in the process of software project implementation to achieve the above objectives.

(1) The interviewees in the interview-based qualitative research are selected based on the following conditions:

① About 10 software project managers or supervisors with rich practical experience;

② More than five scholars or professors engaged in software project risk management research;

③ More than five company leaders or project leaders of the project owner.

The purpose of the interviews is to identify the project risk factors.

(2) The research objects of the questionnaire survey-based qualitative research in this paper are mainly software project managers and practitioners from Shanghai and Jiangsu. Through an effective evaluation and summary of the identified software project risk factors, this paper analyses their internal relationship, reveals various laws of software project risks and constructs a risk evaluation system for software development risks.

There are no special restrictions on the research objects of this questionnaire survey for two main reasons: first, there are few managers of software projects at all, let alone limited to the rail transit industry, and they may not meet the sample size requirements to a certain extent; second, software project practitioners have different views and opinions on the risks involved in project development as they see them from different angles. Therefore, to ensure that the project risk can be evaluated completely from different angles, the scope of the questionnaire survey objects encompasses all kinds of practitioners of software projects, including project manager, system chief designer, programmer, program developer and all kinds of relevant stakeholders of the project.

From the perspective of operability, to identify project risks, plan risk management, monitor and implement risk response measures in combination with the life cycle of

software development project, the research first establishes the possible risk factors and then analyses the causes in each project stage before designing the questionnaire.

(3) Failure factor identification

The domestic and foreign literature on software project development failure factor identification and evaluation is comprehensively summarized to gain a general understanding of the factors that can lead to the failure of software development projects. Table 4-5, classifies and summarizes the factors that can explain the failure of software development projects from different angles.

Factor Category	**Factor Item**
A Requirement Factors	A1 Requirements are not clearly defined
	A2 Insufficient requirement research
	A3 Changing requirements
	A4 Lack of effective requirement change management
	A5 Cannot reach consensus on controversial requirements
	A6 Gold plating requirements
B Technical Factors	B1 Software design defects
	B2 Advanced but immature technical proposal adopted
	B3 Development facilities are not in place or the development environment is chaotic
	B4 The interface with the third-party system is not smooth
	B5 Software iteration management confusion
	B6 The development process is not standardized or scientific enough
C Schedule Factors	C1 Unreasonable project schedule
	C2 Lack of effective monitoring of progress
	C3 Project management and coordination are difficult
	C4 Project resources are not in place

	C5 The quality assurance system is not effectively implemented
	C6 The development environment of the project is poor
	C7 Lack of support from the senior management
D Cost Factors	D1 Party A's bidding price is low
	D2 The bidding price of Party B is low
	D3 The construction budget cost exceeds the bid budget cost
	D4 The financial cost exceeds the budget cost
	D5 Project costs exist that are not estimated
	D6 No attention to the management and control of the quality cost and construction period cost
E Personnel Factors	E1 The project manager is inexperienced
	E2 The project technicians lack experience
	E3 Key personnel changes
	E4 Insufficient user cooperation leads to requirement deviation
	E5 Project personnel are not in place in time or the personnel turnover rate is high
	E6 Friction within the development team
F External Factors	F1 Unfair or non-compliant contract
	F2 The government or another institutions has placed restrictions on the development of the project
	F3 New industry standards or laws and regulations have been issued recently
	F4 Unpredictable market turmoil

Table 4-5 Software Failure Factor List

4.3.2 Construction of the Influencing Factor System

Based on Table 4-3 and two rounds of expert consultation on the failure-influencing factors of PIS software development, the expert opinions are counted and analysed and finally the influencing factor system of PIS software failure is constructed. The system includes the customer requirements, technical factors, project management, company system, employee characteristics, product characteristics, leadership style and external

environment, each of which is also composed of several influencing factors, as shown in Table 4-6.

Dimension	Code	Influencing Factor
Customer Requirements	S1	Requirements are not clearly defined
	S2	Unclear requirements
	S3	Changing requirements
	S4	Inconsistent opinions of customer leaders
Technical Factors	S5	Insufficient testing
	S6	Site test environment not allowed
	S7	Advanced but immature technical proposal adopted
	S8	Development process is not standardized or scientific enough
	S9	Hardware problem resulting in software function change
	S10	Product quality control not in place
	S11	Cost is out of control
Project Management	S12	Difficult project management and coordination
	S13	Insufficient project management experience
	S14	Lack of effective requirements change management
	S15	Inadequate prior risk review
Company System	S16	No good planning for products
	S17	Poor knowledge management
	S18	Lack of incentive mechanism for employees
	S19	Basically, no technology outsourcing
Employee Characteristics	S20	Developers are inexperienced and have poor development habits
	S21	No core technical talents
	S22	Tasks of project personnel are complicated
	S23	The level of developers is uneven
	S24	High employee mobility

	S25	Complex site commissioning environment
Product Characteristics	S26	Software product is complex and quickly updated
	S27	Many and disorderly ways of downloading software
Leadership Style	S28	No harmonious relationship between leaders and employees of the company
	S29	Excessive leadership intervention (project manager)
	S30	Lack of support from senior management
External Environment	S31	The government or another institution has placed restrictions on the project development
	S32	New industry standards or laws and regulations have been issued recently
	S33	Unpredictable market turmoil

Table 4-6 Influencing Factor System of PIS Software Failure

4.4 Analysis Model of Influencing Factors of PIS Software Failure Based on the DEMATEL Method

PIS software failure involves many dimensions and influencing factors, and the relationship between them is not isolated but interdependent and interactive, resulting in a more complex system. Clarifying the relationship between various elements is the basis and premise for promoting the development of the PIS software industry. The DEMATEL method is an effective means to achieve the above objectives. The influencing factors of PIS software failure are analysed using the DEMATEL method.

4.4.1 Determination and Calculation of a Direct Influence Matrix

In this paper, the 0, 1, 2, 3 scale method is adopted. Having consulted a large amount of literature, the direct influence relationship between various factors is determined through the expert consultation method, and the direct influence matrix of PIS software failure is obtained, as shown in Table 4-7.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22	S23	S24	S25	S26	S27	S28	S29	S30	S31	S32	S33
S1	0	3	3	2	3	2	2	0	1	3	3	2	0	0	3	3	0	0	0	0	0	2	0	1	0	2	2	1	1	0	0	0	0
S2	3	0	3	2	3	2	2	0	1	3	3	2	0	0	3	3	0	0	0	0	0	2	0	1	0	2	2	1	1	0	0	0	0
S3	3	3	0	2	3	2	2	0	1	3	3	2	0	0	3	3	0	0	0	0	0	2	0	1	0	2	2	1	1	0	0	0	0
S4	3	3	3	0	3	2	2	0	1	3	3	3	0	0	2	3	0	0	0	0	0	2	0	1	0	1	0	1	2	0	0	0	0
S5	0	0	0	0	0	0	0	0	3	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	2	0	0	0	0	0	0
S6	0	0	0	0	3	0	0	0	3	3	3	2	0	0	0	0	0	0	0	0	0	1	0	1	2	3	2	0	0	0	0	0	0
S7	0	0	0	0	3	3	0	0	0	2	2	0	0	0	2	0	0	0	0	0	0	0	0	0	3	3	2	0	2	0	0	0	0
S8	0	0	0	0	1	0	0	0	0	2	2	1	0	0	0	1	0	0	0	0	0	2	1	0	0	2	1	0	0	0	0	0	0
S9	0	0	2	0	3	2	2	0	0	3	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0
S10	0	0	0	0	2	0	0	1	0	0	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
S11	0	0	3	3	3	0	3	0	1	3	0	3	0	0	0	0	0	1	0	0	0	3	0	0	0	3	1	0	2	3	0	0	0
S12	2	2	2	2	2	2	0	1	1	2	2	0	0	1	2	2	0	0	2	0	0	2	0	0	0	0	0	0	0	2	0	0	0
S13	2	2	2	2	2	2	0	1	1	3	3	2	0	0	2	2	0	0	2	0	0	2	0	2	0	0	0	0	3	0	0	0	0
S14	1	2	3	1	3	3	0	1	1	3	3	1	0	0	0	2	0	0	0	0	0	3	0	0	1	2	2	0	2	0	0	0	0
S15	1	1	1	1	3	2	2	0	1	3	3	3	0	2	0	3	0	0	1	0	0	3	0	0	3	3	2	0	2	0	0	0	0
S16	1	1	1	1	3	2	2	0	1	3	3	3	0	2	0	0	0	0	0	0	0	3	0	0	2	3	2	0	2	0	0	0	0
S17	0	0	0	0	2	0	0	0	0	3	3	2	3	0	3	3	0	0	0	0	0	2	0	0	0	1	0	0	3	0	0	0	0
S18	0	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	3	0	0	0	0	0

S19	0	0	0	0	2	0	3	0	0	3	0	1	3	0	2	2	0	0	0	0	0	3	0	0	0	3	3	2	0	0	0	0	0
S20	0	0	0	0	2	1	2	3	0	2	2	1	0	0	2	2	0	0	0	0	3	2	0	0	0	2	1	0	0	0	0	0	0
S21	0	0	0	0	3	0	3	0	0	3	3	0	0	0	3	3	2	0	0	0	0	0	3	3	0	3	2	0	0	0	0	0	0
S22	0	0	0	0	2	0	0	1	0	2	2	1	0	0	1	1	1	0	0	0	0	0	0	2	0	2	1	0	1	0	0	0	0
S23	0	0	0	0	2	0	1	1	0	1	1	0	0	0	1	1	1	0	0	0	0	3	0	2	0	1	1	0	0	0	0	0	0
S24	0	0	0	0	2	0	0	2	0	2	2	2	0	0	0	0	3	0	0	1	3	2	3	0	0	2	1	3	0	0	0	0	0
S25	0	0	0	0	3	0	0	0	0	3	3	3	0	0	0	0	0	0	0	0	0	3	0	0	0	3	0	0	2	0	0	0	0
S26	0	0	1	0	3	3	0	0	0	3	2	0	0	2	0	0	0	0	0	0	0	3	0	0	0	0	3	1	1	0	0	0	0
S27	0	0	0	0	3	3	0	0	0	3	2	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
S28	0	0	0	0	1	0	0	0	0	3	2	0	0	0	0	0	0	3	1	1	1	2	0	3	0	0	0	0	0	0	0	0	0
S29	0	0	2	3	2	0	1	1	0	1	1	0	0	0	1	1	0	0	0	0	0	2	0	2	0	2	0	3	0	0	0	0	0
S30	0	0	0	3	2	2	0	0	0	2	2	3	0	0	2	2	0	3	3	0	3	3	0	3	0	2	1	2	0	0	0	0	0
S31	0	0	3	0	3	3	3	0	0	3	3	3	0	0	3	3	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	3	3
S32	0	0	3	2	2	2	2	0	0	2	3	3	0	0	3	3	0	0	0	0	3	3	0	2	0	2	0	0	0	0	2	0	3
S33	0	0	2	2	2	2	1	0	0	2	2	2	0	0	2	2	0	0	0	0	0	2	0	3	0	3	0	0	0	2	3	3	0

Table 4-7 PIS Software Failure Direct Impact Matrix

The matrix data in Table 4-7 intuitively show that there are various relationships among the factors in the system, but it is difficult to identify the key factors. In view of this, according to the matrix data in the table and the steps of the DEMATEL method, the comprehensive influence matrix among various factors is constructed, and, on this basis, the influence degree, affected degree, centrality and cause degree of various factors in the PIS software failure influence factor system are calculated, as shown in Table 4-8:

Code	D Value of Influence Degree	C Value of Affected Degree	D+C Value (M) of Centrality	D-C Value (R) of Cause Degree
S1	2.609	1.063	3.672	1.546
S2	2.609	1.102	3.711	1.507
S3	2.609	2.139	4.748	0.47
S4	2.607	1.743	4.349	0.864
S5	0.866	5.158	6.023	-4.292
S6	1.274	2.494	3.768	-1.22
S7	1.25	1.955	3.205	-0.705
S8	0.735	0.77	1.505	-0.035
S9	1.135	1.457	2.591	-0.322
S10	0.577	5.406	5.983	-4.829
S11	2.023	5.019	7.043	-2.996
S12	2.066	3.101	5.166	-1.035
S13	2.396	0.221	2.617	2.176
S14	2.141	0.662	2.803	1.479
S15	2.499	1.998	4.497	0.502
S16	2.145	2.271	4.416	-0.126
S17	1.682	0.42	2.102	1.262
S18	0.463	0.433	0.896	0.031
S19	1.588	0.521	2.108	1.067
S20	1.503	0.114	1.616	1.389
S21	1.877	0.539	2.416	1.338

S22	1.02	3.851	4.871	-2.831
S23	0.921	0.348	1.268	0.573
S24	1.586	1.505	3.091	0.081
S25	1.153	1.134	2.287	0.019
S26	1.217	3.915	5.132	-2.698
S27	0.731	2.435	3.166	-1.703
S28	0.92	1.047	1.967	-0.127
S29	1.41	1.874	3.284	-0.464
S30	2.385	0.715	3.1	1.67
S31	2.528	0.147	2.675	2.381
S32	2.852	0.174	3.027	2.678
S33	2.524	0.174	2.698	2.349

Table 4-8 Comprehensive Influence Relationship of Factors

4.4.2 Analysis and Discussion of the Results

In terms of the influence degree and affected degree, in the PIS software failure influencing factor system, the following factors have a high degree of comprehensive influence on other factors: S32: the recent introduction of new industry standards or laws and regulations (2.852); S1: unclearly defined requirements (2.609); S2: unclear requirements (2.609); S3: changing requirements (2.609); S4: inconsistent opinions of customer leaders (2.607); S31: the government or another institution has imposed restrictions on project development (2.528); S33: unpredictable market turmoil (2.524); S15: inadequate prior risk review (2.499); S13: insufficient project management experience (2.396); S30: lack of support from senior leaders (2.385); S16: lack of good product planning (2.145); S14: lack of effective requirement change management (2.141); S12: difficult project management and coordination (2.066); S11: cost out of control (2.023); and S21: no core technical talents (1.877). The highly affected factors are S10: product quality control not in place (5.406); S5: insufficient testing (5.158); S11: cost is out of control (5.019); S26: software product is complex and quickly updated (3.915); S22: the tasks of project personnel are complicated (3.851); S12:

difficult project management and coordination (3.101); S6: site test environment not allowed (2.494); S27: many and disorderly ways of downloading software (2.435); S16: lack of good planning for products (2.271); and S3: changing requirements (2.139). The distribution of the influence degree and affected degree of the factors is shown in Figure 4-10.

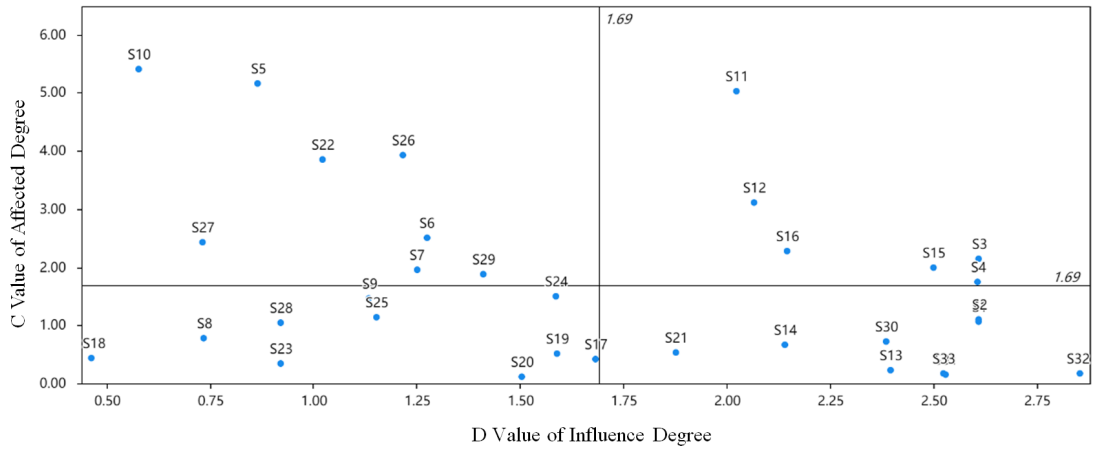


Figure 4-9 Distribution Diagram of Influence Degree and Affected Degree of Factors

In terms of centrality and cause degree, it can be read from Table 4-8 that S11: cost out of control (7.043); S5: insufficient testing (6.023); S10: product quality control not in place (5.983); S12: difficult project management and coordination (5.166); S26: software products are complex and quickly updated (5.132); S22: the tasks of project personnel are complicated (4.871); S3: constant changing requirements (4.748); S15: inadequate prior risk review (4.497); S16: lack of good planning for products (4.416); and S4: inconsistent opinions of customer leaders (4.349) have high centrality values, which means that they play a key role in PIS software failure. In terms of reasons, S32: new industry standards or laws and regulations have been issued recently (2.678); S31: the government or another institution has imposed restrictions on project development (2.381); S33: unpredictable market turmoil (2.349); S13: insufficient project management experience (2.176); S30: a lack of support from senior leaders (1.67); S1: requirements are not clearly defined (1.546); S2: unclear requirements (1.507); S14: lack of effective requirements change management (1.479); S20: developers are inexperienced and have poor development habits (1.389); S21: no core technical talents

(1.338); S17: poor knowledge management (1.262); S19: basically no technology outsourcing (1.067); S4: inconsistent opinions of customer leaders (0.864); S23: the levels of developers are uneven (0.573); S15: inadequate prior risk review (0.502); S3: changing requirements (0.47); S24: high employee mobility (0.081); S18: lack of an incentive mechanism for employees (0.031); and S25: complex site commissioning environment (0.019) all have cause degree values greater than zero and constitute the causal factors. Among them, the cause degree values of S32, S31, S33, S13, S30, S1, S2, S14, S20, S21, S17, S19, S4, S23, S15 and S3 are high, meaning that they are the key cause factors. The distribution of the centrality and cause degree of the factors is shown in Figure 4-11.

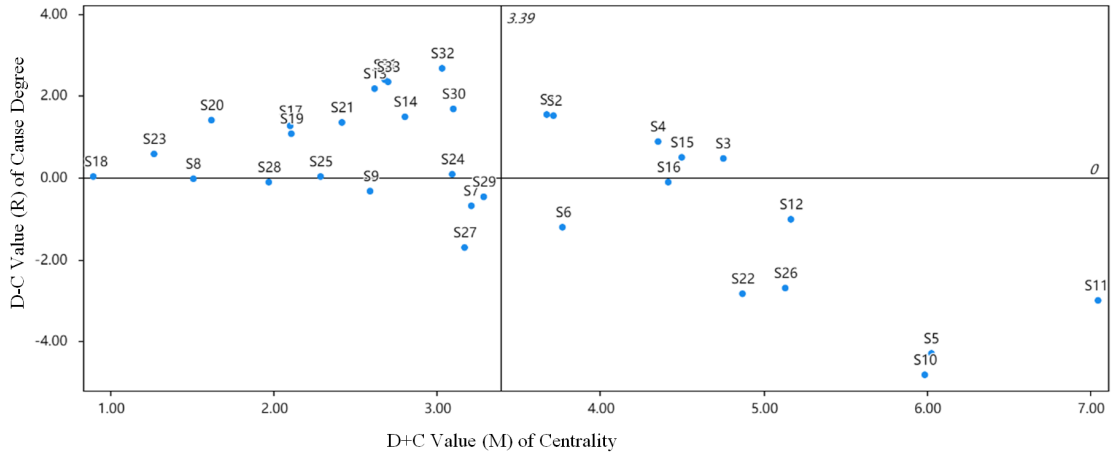


Figure 4-10 Factor Centrality and Cause Degree Distribution

In terms of the comprehensive impact relationship, the corresponding index data on the impact of each dimension on PIS software failure are obtained by adding the influence degree, affected degree, centrality and cause degree of the factors included in each dimension, as shown in Table 4-9.

Dimension	Influence Degree	Affected Degree	Centrality	Cause Degree
Customer Requirements	10.434	6.047	16.48	4.387
Technical Factors	5.26	11.834	17.092	-6.574
Project Management	11.702	16.407	28.109	-4.703
Company System	5.878	3.645	9.522	2.234
Employee Characteristics	6.907	6.357	13.262	0.55

Product Characteristics	3.101	7.484	10.585	-4.382
Leadership Style	4.715	3.636	8.351	1.079
External Environment	7.904	0.495	8.4	7.408

Table 4-9 Analysis of the Impact of Each Dimension on PIS Software Failure

Table 4-9 shows that the cause degrees of customer requirements, the company system, employee characteristics, the leadership style and the external environment are greater than zero; that is, all of them are cause factors. However, the cause degrees of technical factors, project management and product characteristics are less than zero; that is, all of them are classified as result factors. The centrality of project management is 28.109, which is the highest among all the dimensions, indicating that it is very important for software development. The result that its influence degree and affected degree are the highest shows this dimension is in a very important position and is a phased result factor. Its behaviour is restricted by other dimensions to a certain extent. The cause degree and influence degree of customer requirements are both large positive values, as well as the centrality, indicating that the requirements of the customer, as the demander of PIS software, has a central impact on PIS software failure, but the impact role cannot ignore the important position of project management. The external environment has the highest cause degree, indicating that it is a direct cause of PIS software failure. Therefore, customer requirements and the external environment are the two most important dimensions that affect the function failure and poor sales of PIS software.

4.5 Summary

This chapter identifies the influencing factors of PIS software failure from both the semi-structured interviews and the literature review. Through the literature reading and data sorting of a large amount of software project development risk management and risk factor identification, combined with potential risk analysis in the project implementation cycle and project expert investigation, the research analyses the phenomenon of PIS software failure in seven dimensions: customer requirements, technical factors, project management, the company system, employee characteristics, the leadership style and the external environment. As a result, the influencing factor system of PIS software failure is constructed. Multiple methods, including expert consultation, a questionnaire survey and DEMATEL, are applied to perform the

influencing factor analysis of PIS software failure. The outcome of the expert consultation and questionnaire survey is screened in two rounds. The DEMATEL method is used to evaluate various factors regarding the influence degree, affected degree, centrality and cause degree, respectively. Finally, by adding the influence degree, affected degree, centrality and cause degree of the factors to each dimension, the corresponding index data on the impact of each dimension on PIS software failure are obtained, the comprehensive impact relationship is analysed and the corresponding conclusions are drawn.

Chapter 5. Construction of the Theoretical Model and Design of the Questionnaire

Based on the identification and analysis of the influencing factors on PIS software failure, and focusing on PIS software failure, this chapter constructs an analysis framework of the relationships between customer requirement defects, employees' negative characteristics, insufficient software testing, inadequate project management, an uncertain external environment and PIS software failure to put forward theoretical hypotheses. Combining the analysis framework with the theoretical assumptions, a theoretical analysis model of the PIS software failure mechanism is formed, the measurement of relevant variables is analysed, a scale is developed and a questionnaire is formed.

5.1 Construction of the Research Model

According to the identification and analysis of the factors influencing PIS software failure in Chapter 4, through the qualitative research on the interview contents of Company A, and the factors influencing PIS software failure identified in the literature, through the analysis of the DEMATEL model, it is found that inadequate project management, employees' negative characteristics and insufficient software testing are the direct factors leading to PIS software failure and that customer requirement defects and an uncertain external environment will affect inadequate project management, employees' negative characteristics and insufficient software testing, resulting in software failure. Therefore, the conceptual model studied in this paper is proposed as shown in Figure 5-1. The model involves six constructs – customer requirement defects, an uncertain external environment, employees' negative characteristics, insufficient software testing, inadequate project management and PIS software failure – forming the action path between the factors to explore the impact mechanism of PIS software failure.

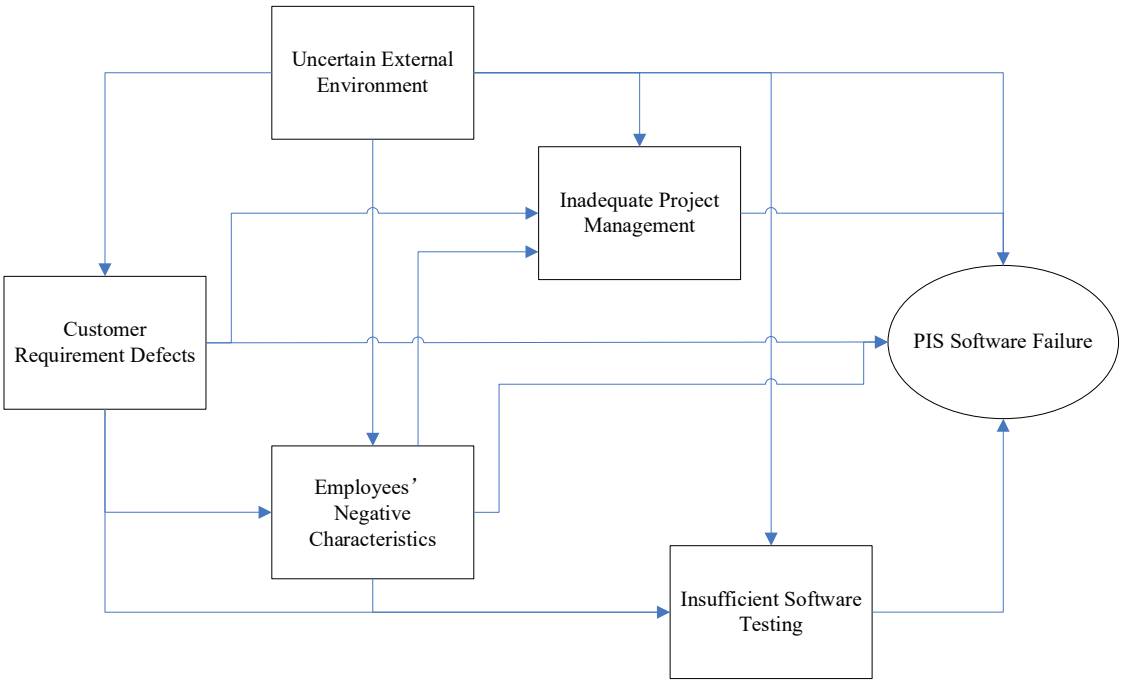


Figure 5-1 Conceptual Model of PIS Software Failure

The conceptual model of this paper will mainly answer the following sub-questions of the influencing mechanism of PIS software failure:

(1) Can project management effectively reveal the influencing mechanisms of PIS software failure?

(2) What is the impact of the software testing on PIS software failure?

(3) How do employees' characteristics lead to PIS software failure?

(4) What impact does the external environment of software development projects have on the whole process of PIS software development?

(5) What is the mechanism through which customer requirement affects PIS software failure?

(6) What is the action path of these influencing factors on PIS software failure?

5.1.1 PIS Software Failure

Software failure is caused by the execution of defective programs, which are a result of human factors, and the software failure mode has not been determined yet (LIU, 2000).

Some people divide software defects into requirement defects, design defects and code defects (2009, Vitharana, 2015); others divide software defects into requirement defects, design defects, documentation defects, algorithm defects, interface defects and performance defects (Lawrence H. Putnam and Myers, 1992, Chittister and Haimes, 1996). Software defects can also be divided into requirement defects, design defects, coding defects, documentation defects, management defects and installation defects (Chittister and Haimes, 1996). According to the results of the semi-structured interviews conducted in Company A, software failure is mainly reflected in "poor stability of software products", "poor maintainability of software products" and "poor customer satisfaction with products". These three aspects can basically include the previous classification methods. Therefore, they are applied to measure PIS software failure.

5.1.2 Inadequate Project Management

A Guide to the Project Management Body of Knowledge, published by the American Project Management Institute (PMI) (2008a), lists project risk management as one of 10 knowledge areas and divides project risk management into seven parts: planning risk management, identifying risk, implementing qualitative risk analysis, implementing quantitative risk analysis, planning the risk response, implementing the risk response and supervising risk. The quality of project management directly affects the software development. Therefore, poor project management leads directly to PIS software failure. According to the results of the semi-structured interviews and the risk factor identification, the contents of inadequate project management can be summarized as follows: "product quality control is not in place", "project management and coordination are difficult", "cost is out of control", "project management experience is insufficient", "prior risk review is insufficient", "lack of effective requirement change management", "the company's knowledge management is poor", "products are not well planned", "advanced but immature technology is adopted" and "the development process is not standardized and scientific"; these will lead directly to the failure of PIS software development. Therefore, the paper puts forward the following assumption:

Hypothesis 1: Inadequate project management is positively related to PIS software

failure.

5.1.3 Insufficient Software Testing

Through software testing, we can identify the defects and bugs that may lead to software function failure. However, no testing technology or tool can find all the defects in a set of software codes. At different stages of the software life cycle, different testing techniques (methods or tools) need to be used to locate different types of defects and avoid software failure. Therefore, this paper extracts the factors related to software testing, including "insufficient testing", "complex on-site debugging environment", "lack of an on-site testing environment", and considers that some characteristics of the software product itself will also affect software testing. When measuring these constructs, "complex and fast upgrading software products" and "many and chaotic software download methods" are also included. The word "insufficient" refers here to all the above discussed factors concerning software testing and the inability of current methods to identify and remedy all software defects and bugs. Therefore, the paper suggests the following assumption:

Hypothesis 2: Insufficient software testing is positively correlated with PIS software failure.

5.1.4 Employees' Negative Characteristics

Software failure is mostly caused by people, so the characteristics of the company's employees deserve special attention. In the semi-structured interviews, the employee characteristics proposed by the respondents include: "the tasks of project personnel are complicated", "the company has no core technical talents", "developers lack experience and have poor development habits", "lack of an incentive mechanism for employees", "high employee mobility" and "uneven level of developers". All these factors are used as measurement items for employees' characteristics. In particular, the use of technology and the knowledge and experience of developers and testers will cause software defects, which will lead to software failure. Therefore, the paper proposes the following hypotheses:

Hypothesis 3: Employees' negative characteristics are positively correlated with

software failure;

Hypothesis 4: Employees' negative characteristics are positively correlated with insufficient software testing;

Hypothesis 5: Employees' negative characteristics are positively correlated with inadequate project management.

5.1.5 Customer Requirement Defects

Customer requirements include the problems to be solved, the objectives to be achieved and the conditions required to achieve these objectives. They constitute a description of the software development work. Because a software product is an intangible product, it is impossible to define all the requirements accurately in the initial stage. With the progress of the project and users' perception, the requirements constantly change. If multiple leaders of the customer hold inconsistent ideas, the requirement risk will increase further. According to the semi-structured interviews with Company A, the requirement factors leading to software failure are unclear customer demand, inconsistent functional results and customer expectations, too vague customer requirements, failure to meet the customer's implicit demand, unclear customer input demand, constantly changing customer demand, different opinions of customer leaders and so on. The questionnaire helps to identify the requirement risk factors as "the requirement definition is not clear", "the requirement research is insufficient", "the requirement is constantly changing" and "there is a lack of effective requirement change management". Based on comprehensive consideration, "unclear requirement definition", "unclear requirement", "constantly changing requirements" and "inconsistent opinions of customer leaders" are selected to measure customer demand. The reflection of customer requirements in software products needs to be realized by people, technology and management throughout the whole software development process. Therefore, the paper puts forward the following assumptions:

Hypothesis 6: Customer requirement defects are positively correlated with PIS software failure;

Hypothesis 7: Customer requirement defects are positively related to inadequate

project management;

Hypothesis 8: Customer requirement defects are positively correlated with employees' negative characteristics;

Hypothesis 9: Customer requirement defects are positively correlated with insufficient software testing.

5.1.6 Uncertain External Environment

A PIS software development project is an open system, not just a simple technical project, but a complex social interaction process composed of various elements, which can be divided into the technology subsystem, personnel subsystem, organization management subsystem and environment subsystem. In this paper, the external market and leadership style are attributed to the environment subsystem of software projects. Through semi-structured interviews and risk factor identification, the uncertainty of the external environment includes "too much intervention by leaders (mainly project managers)", "the relationship between company leaders and employees is not harmonious", "insufficient support from senior leaders", "new industry standards or laws and regulations have been issued recently", "the government or another institution has imposed restrictions on the project development" and "unpredictable market turmoil". The effect of software project implementation will not only be affected by many factors, such as technology, organization and management, but will also fluctuate with the dynamic changes of the organizational environment and external social environment, resulting in behaviour deviation of the organizations and individuals in the system. When the self-repair and adjustment function of the system cannot correct these deviations, the result is that the system may fail or even cause accidents. Therefore, this paper puts forward the following assumptions:

Hypothesis 10: An uncertain external environment is positively related to PIS software failure;

Hypothesis 11: An uncertain external environment is positively related to inadequate project management;

Hypothesis 12: An uncertain external environment is positively correlated with

employees' negative characteristics;

Hypothesis 13: An uncertain external environment is a positively correlated with insufficient software testing;

Hypothesis 14: An uncertain external environment is a positively correlated with customer requirement defects.

To sum up, the research model of this paper is illustrated in Figure 5-2:

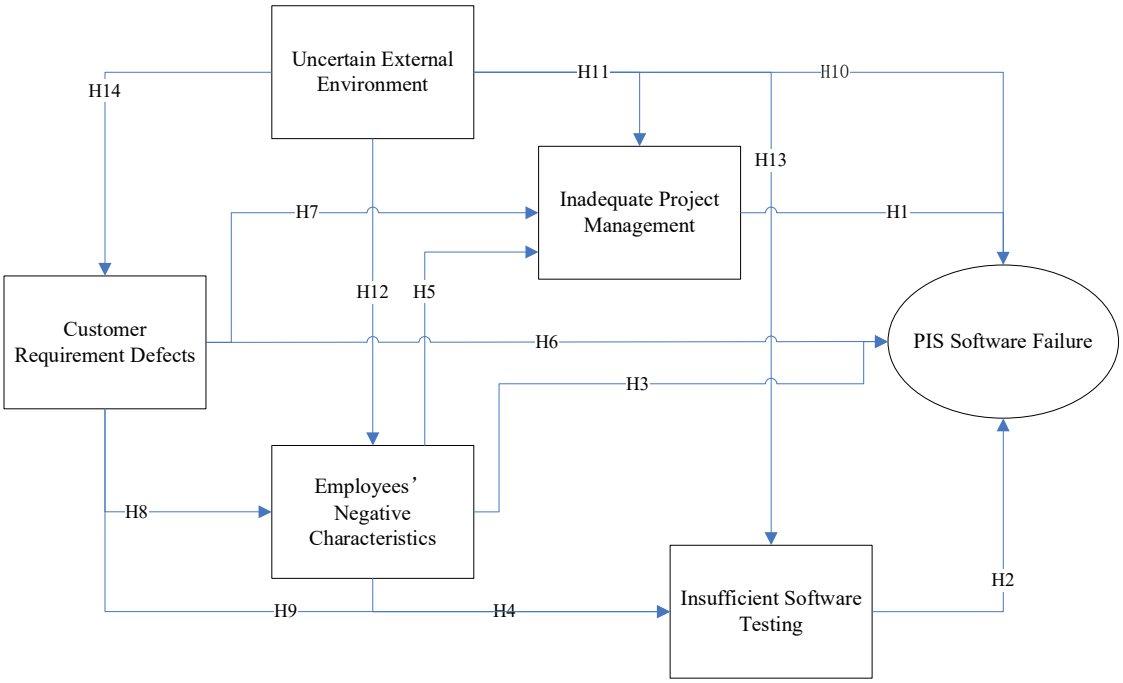


Figure 5-2 Research Model

5.2 Scale Development

This study mainly uses the questionnaire survey method to test the influencing factors of PIS failure further. Based on the data obtained from practical operation, it verifies all the hypothetical relationships proposed and then further explains and generalizes the conclusions. The scale of high validity and reliability is the most important factor in the process of empirical research. Inappropriate field sampling, a poor factor structure and low reliability of internal consistency are the main problems restricting the construction and verification of the theory (Hinkin, 1995). The lack of a good framework guiding researchers to complete the scale development process step by step will lead to

defective measurement tools (Price, 1997). To develop a scale with high validity and reliability, it is necessary to follow the scale development process.

5.2.1 Scale Development Process

Designing a scientific and reasonable questionnaire is the primary premise of scientific research. To ensure the reliability and validity of the questionnaire, according to Hinkin's (1995) research, the development of a scale consists of six stages: generation of initial measurement items, questionnaire management, reduction of initial measurement items, confirmatory factor analysis, convergent/discriminant validity test and re-test. In the process of this study, to ensure the high validity and reliability of the constructs involved, we distributed the questionnaire survey twice: first to form a preliminary questionnaire based on the generation of initial measurement items, conduct a small-scale questionnaire pilot test, then adjust the initial measurement items through exploratory factor analysis and form a relatively perfect scale and questionnaire; and second, through large-scale distribution, mainly based on the scale and questionnaire of the previous stage, to test the reliability and validity of the scale using confirmatory factor analysis.

5.2.2 Main Tasks of Each Scale Development Stage

Based on the reading of a substantial amount of literature at home and abroad, the scales all refer to existing mature scales. The questionnaire design process consists of the following stages:

Stage 1: Generation of Initial Measurement Items

The key to the success of a measurement item is to clarify the theoretical basis, because this will determine the connotation of the construct and thus the content domain of the measurement item. Domain sampling theory shows that it is impossible to measure the complete domain of a construct, but measurable samples obtained from the potential measurement items can fully represent the construct.

Based on the previous semi-structured interviews, factor identification and literature review, this paper has a clear understanding of the research problems and gives the operational definition of the constructs involved in the research. The initial

measurement items of the construct can be generated through the deductive method, and a pilot test is carried out through a small-scale questionnaire to improve the reliability and validity of the scale. In the final data analysis process, confirmatory factor analysis is further used to verify the factor structure, and the reliability and validity of each scale are analysed and evaluated. The expression of the scale needs to follow some principles; for instance, the expression should be as simple as possible, the language should be familiar to the respondents, the consistency of all the measurement items in perspective should be ensured and evaluation behaviours or feelings should not be mixed. In terms of the number of measurement items, a construct uses at least three items. Most scholars believe that it is more appropriate to measure each construct with four to six items. In the process of generating the initial measurement items, this paper develops four to six item measurements for the constructs with relatively mature scales. For the new constructs proposed in this paper, such as inadequate project management, more than six measurement items are developed to ensure that there are enough items for reduction and adjustment in the later exploratory factor analysis.

Stage 2: Expert Interview

According to the initial measurement items generated in the previous Stage 1, a preliminary questionnaire is formed. It is also necessary to improve the questionnaire through content validity evaluation. The aim of content validity evaluation is to judge whether each measurement item is representative and comprehensive. Its effectiveness mainly depends on the actual background of the measurement item. Through the literature review analysis and interview survey, this paper uses content validity evaluation to determine the representativeness and comprehensiveness of the measurement items. Subsequently, the questionnaire is improved and can be distributed. It is divided into two general stages: a small-scale pilot test and a large-scale questionnaire distribution. The main purpose of the small-scale pilot test is to eliminate uncertainty in the variable structure, content selection and questioning angle, to improve the reliability and validity of the scale further to ensure the quality of data after the large-scale questionnaire distribution, to improve the reliability and validity of the

research and to ensure the correctness of the theoretical test. The main purpose of large-scale questionnaire distribution is to collect data that can be used to measure the constructs involved in the research and then study the relationship between constructs and test the hypotheses put forward in the theoretical construction stage.

Stage 3: Initial Scale Reduction

When the data of a small-scale pilot survey are obtained, the initial measurement items need to be reduced through data analysis. In this paper, reliability analysis is used to judge the accuracy or precision of the measurement tools, and the initial measurement items are reduced accordingly. Generally, two indexes are used to determine the reliability of a construct, namely Cronbach's α and item-to-total correlation. Most scholars believe that a Cronbach's α greater than 0.7 and item-to-total correlation greater than 0.5 indicate that a scale has good reliability.

Stage 4: Confirmatory Factor Analysis

Confirmatory factor analysis tests whether the number of factors and factor loads of observed variables are consistent with the expectations based on the pre-established theory. After the large-scale questionnaire is distributed, confirmatory factor analysis should still be performed to test the matching degree between factors and measurement items. Confirmatory factor analysis is generally completed using structural equation modeling (SEM) software, such as AMOS.

In confirmatory factor analysis, many statistical indicators can be used to judge the fitting degree of the measurement model. The commonly used indicators and their thresholds are shown in Table 5-1:

Index	χ^2/df	GFI	NFI	CFI	RSMEA
Threshold Value	<5	>0.8	>0.9	>0.9	<0.08

Table 5-1 Fit Indexes and Threshold Value of Confirmatory Factor Analysis

Stage 5: Convergence/Discriminant Validity Test

Validity is to tell the researcher the correctness of the measurement results, whether the measurement is what he/she wants, and to what extent it gives what he/she wants, that is, the proximity between the measurement results and the expected measurement objectives (Zeng Wu-yi and Bing-yi, 2005, Wang, 2009). Generally, the two basic

characteristics of questionnaires are reliability and validity of which the former measures the consistency of the questionnaire measurement results (i.e., the results obtained when the same method is used to repeatedly measure the same object), rather than considering whether the measurement results are correct; while the latter measures the correctness and accuracy of the questionnaire measurement results, reflecting the effectiveness of the questionnaire measurement, whether the purpose of questionnaire measurement is achieved, may include whether the measured object is expected, whether the operational definition of variables can reflect the original concept and so on. In short, validity is the primary condition of the questionnaire and reliability is an indispensable content of validity. If the validity is high, the reliability is also high; while if the reliability is high, the validity is not necessarily high. On the contrary, if the reliability is low, the validity is also low; while if the validity is low, the reliability is not necessarily low. There are three types of validity: content validity, criterion validity and structure validity (Zeng Wu-yi and Bing-yi, 2005, Wang, 2009, Bartko, 1966).

(1) Content validity. It refers to the appropriateness of the item content of the questionnaire to the sampling of the predicted topic or behaviour range. Content is the simplest and most basic subjective evaluation method for validity. Generally, according to the items selected in the measurement scale, it is only observed from the surface to judge whether they can represent the content or topic to be measured. The necessary conditions for content validity: first, there should be a well-defined content scope. It can be a clear and limited overall topic, specific knowledge and skills, or complex behaviour. Second, the measurement items should be representative samples of the defined content range; that is, the sampling validity is the content validity, which is used to determine that the items contained in the questionnaire should be representative. If all the contents can be regarded as a whole, the questionnaire is a sample containing the materials and conditions of the behaviour to be measured. Only representative samples can be selected to infer the overall performance. Therefore, the appropriateness of sampling is very important. To obtain a high-validity scale it is usually necessary to collect and read the data related to the measurement content and relevant previous research reports as much

as possible so that the designed scale items can better cover all aspects of the relevant content, concretize the measurement objectives into measurement objectives at different levels and determine the proportion of each objective in the overall. Experts are invited to analyze the representativeness and suitability of the measurement items and make necessary modifications until most experts are satisfied. The content validity is highly subjective, and generally adopts expert judgment method, duplicate method, retest method and experience method. The main decision is whether the measurement tool measures the construct of investigating all measurements and whether appropriate samples of the construct are provided. Therefore, following the theoretical framework, collecting all relevant questions and variables and selecting questions that can completely cover the defined research scope can ensure the content validity of the questionnaire.

(2) Criterion validity. It refers to the consistency between the questionnaire measurement results and the validity standard (an external standard assumed to be or defined as valid, i.e., a criterion). According to the validity standard, the acquisition time can be divided into: (a) concurrent validity, that is, correlation between the results of the questionnaire and the validity standard measurement of the research object at the same time; and (b) predictive validity, which refers to the degree of coincidence between the questionnaire measurement results and the future actual results after a period. Concurrent validity and predictive validity directly reflect the correlation validity of the questionnaire.

(3) Construct validity. The concept was first proposed by the Joint Committee of the American Psychological Association (APA), the American Educational Research Association (AERA) and the National Council on Measurement in Educational (NCME). It refers to the consistency between experiment and theory. By comparing with the theoretical hypothesis to test, a questionnaire is designed to measure the relevant concepts and evaluate the extent to which the results can measure the theoretically expected characteristics. Structure validity is used for the measurement of multiple indicators, mainly including convergent validity and discriminant validity. The

former is to ensure that the measurement results of questionnaire items with the same concept are aggregated or highly correlated with each other when measuring multiple indicators of the same construct; the latter refers to the degree of difference between a structure and other structures in the same model, that is, when measuring two different concepts, there is a high degree of distinction between the results of the survey and whether the same questionnaire is used. The criterion for judging convergent validity is that the factor load of the observed variable is greater than 0.5 and reaches the significance level of significance 0.05, while the load value on other common factors is low. If the load value of a question item on all factors is small, it indicates that the problem described by the question item is of little significance and can be deleted. The criterion for judging discriminant validity is that the average variance extracted (AVE) is greater than the common variance of each latent variable.

Among the above three commonly used validity analysis methods, it is often difficult to select an appropriate criterion for the criterion validity method. The analysis of content validity is generally completed in the questionnaire management stage. After confirmatory factor analysis, only structure validity, including convergent validity and discriminant validity, needs to be tested. Scale development is a continuous cycle process. Sometimes the above stages need to be repeated until a scale with high reliability and validity is developed.

5.3 Setting of the Initial Questionnaire Measurement Items

5.3.1 Initial Measurement Items for Software Failure

Among the measures to gauge the outcomes of software development projects, the concept and measurement of software failure still lack consistency, and they are generally measured using the ability to meet the expectations of shareholders. “In general, there remains a lack of consensus on how to define success, lack of success, and failure”(McLeod and MacDonell, 2011). Such terms are perceived to be vague and difficult to measure (Butler and Fitzgerald, 2001, Lynch and Gregor, 2004, Wilson and Howcroft, 2002). Software failure is defined as the dependent variable of the model study in this paper and the main variable concerned in the research. When discussing

the variable selection of project failure, Pinto and Mantel (1990) pointed out three key aspects of the evaluation of project success or failure: (1) the implementation process itself; (2) the perceived value of the project; and (3) the customer satisfaction with the delivered project. In this paper, Company A usually uses a quarterly customer satisfaction questionnaire to survey the project feedback from customers to score respectively the quality, timely delivery rate and service, finally obtaining a comprehensive score. In this paper, "poor stability of software products", "poor maintainability of software products" and "poor customer satisfaction with products" are selected as measurement items with respect to the practical operation of Company A. During the development of the measurement items, suggestions on these three key features are considered, as shown in Table 5-2:

No	Item	Scale Reference
1	Poor stability of software products	**Pinto and Mantel, (1990)**
2	Poor maintainability of software products	
3	Poor customer satisfaction with products	

Table 5-2 Composition of PIS Failure Measurement Items

5.3.2 Initial Measurement Items for Inadequate Project Management

Project management runs through the whole process of software development. The quality of management directly affects the effectiveness of software. Combining the existing literatures at home and abroad with the questionnaire survey, the concept of inadequate project management is measured using the aspects shown in Table 5-3:

No.	Item	Scale Reference
1	Poor product control	Xinyuan Lu (2005); Sixin Xue & GuojunJia (2004)
2	Poor cost control	Jiang & Klein (2000); Wallace & Keil
3	Insufficient internal communication	Taylor (2000); Mizuno (2000); Kapur (1997)
4	Lack of effective customer communication	Wallace et al. (2004); Taylor (2000)
5	Insufficient prior risk review	Xinyuan Lu (2005); Carpers (1994)
6	Lack of effective knowledge	Interview Supplements

	management	
7	Lack of product planning	Xinyuan Lu (2005)
8	Application of an advanced but immature technical proposal	Karolak (1996); Reifer (2002); Xinyuan Lu (2005)
9	Development process is not standardized or scientific enough	Fowler (2005)
10	Lack of project experience	Interview supplements

Table 5-3 Composition of Inadequate Project Management Measurement Items

5.3.3 Initial Measurement Items for Insufficient Software Testing

Software testing can identify the inevitable defects and bugs that arise during different stages of the software life cycle, due to insufficient testing, which can lead to software failure. Combined with the existing literatures at home and abroad, insufficient software testing is measured using the following aspects (Table 5-4):

No.	Item	Scale Reference
1	Insufficient testing of the development system	Andrew (2000); Weijie Ye (2006)
2	Software product is complex and quickly updated	McLeod & MacDonell (2011)
3	Many and disorderly ways of downloading software	Interview supplements
4	Testing environment not allowed	Interview supplements
5	Complex site environment	Interview supplements

Table 5-4 Composition of Insufficient Software Testing Measurement Items

5.3.4 Initial Measurement Items for Employees' Negative Characteristics

McLeod and MacDonell (2011) believed that personnel are an important factor leading to software failure, and the characteristics of the company's employees deserve special attention. In particular, developers' and testers' use of technology and their knowledge and experience can cause software defects, leading to software failure. Combining the semi-structured interviews and the existing literature at home and abroad, the employees' negative characteristics are measured according to the aspects shown in Table 5-5:

No.	Item	Scale Reference
1	Developers lack experience and have poor	Boehm (1989); Mizuno (2000); Pressman

	development habits	(1997); Jiang et al. (2000)
2	The level of developers is uneven	Xinyuan Lu (2005)
3	High employee mobility	Reifer (2002); Boehm(1989); Jiane et al. (2001)
4	The tasks of project personnel are complicated	Interview supplements
5	Lack of core talents	Interview supplements
6	Poor incentive mechanism for employees	Interview supplements
7	Poor employee enthusiasm	Interview supplements

Table 5-5 Composition of Employees' Negative Characteristics Measurement Items

5.3.5 Initial Measurement Items for Customer Requirement Defects

If all the requirements cannot be defined accurately in the initial stage, they will change continuously with the progress of the project and users' perception. If multiple leaders of the customer company hold different ideas and it is difficult to reach a consensus, the requirement risk will be further increased, resulting in software failure. Combining the semi-structured interviews and the existing literature at home and abroad, customer requirement defects are measured using the aspects in Table 5-6:

No.	Item	Scale Reference
1	Customers' requirements constantly changing	McFarlan (1981); Boehm (1989); Standish Group (1995); Sixin Xue & Guojun Jia (2004); Schmidt (2001); Mizuno (2000)
2	Unclear customers' requirements	Boehm (1989); Reifer (2002); Genuchten (1991); Taylor (2000)
3	Insufficient active participation of users	Alter (1979); Schmidt (2001)
4	No consensus of customers' leaders	Interview supplements
5	Requirements not clearly defined	McLeod & MacDonell (2011)

Table 5-6 Composition of Customer Requirement Defects Measurement Items

5.3.6 Initial Measurement Items for Uncertain External Environment

The effect of software project implementation will change with a change in the organizational leadership style or the internal or external environment, resulting in

behaviour deviations of organizations and individuals; when these deviations cannot be corrected, the results may lead to failure or even accidents. Combining the semi-structured interviews and the existing literature at home and abroad, the uncertain external environment is measured using the aspects shown in Table 5-7:

No.	Item	Scale Reference
1	Lack of project support and participation from senior management	Xinyuan Lu (2005); Kerzner (2004); Narki et al. (1993); Jiang et al. (2001); KPMG (1997); Alter (7979); Standish group (1995)
2	No harmonious relationship between leaders and employees of the company	McLeod & MacDonell (2011)
3	Excessive leadership intervention (project manager)	Harold Kerzner (2004)
4	The government or another institution has imposed restrictions on the project development	Wallace (2004); Sijun Xue & Guojun Jia (2004); Weijie Ye (2006)
5	New industry standards or laws and regulations have been issued recently	
6	Unpredictable market turmoil	

Table 5-7 Composition of Uncertain External Environment Measurement Items

5.4 Questionnaire Formation and Pilot Test

5.4.1 Questionnaire Formation

The initial questionnaire for this paper is compiled based on the above analysis (Appendix 5-1). The questionnaire consists of seven parts: the basic personal information of the respondents (gender, age, education level, time engaged in software development, duration of the project, number of project teams and positions held in the project), inadequate project management, insufficient software testing, employees' negative characteristics, customer requirement defects, uncertain external environment and software failure. The Likert 5-point scale marking method is adopted for each item

in all the scales: 1 means very non-compliant and 5 means very compliant. To improve the respondents' interest in and understanding of the survey and eliminate doubts, the survey contains completion instructions, which briefly introduce the research background, research purpose and data confidentiality of the survey. Finally, the questionnaire provides a blank space for respondents to express their opinions freely, enabling them to contribute their own views to the research of this paper.

For the initial questionnaire, to improve the content validity, through semi-structured interviews held in Company A, we conducted in-depth discussions with eight directors and project managers, all of whom have rich experience in software development projects; we then modify and improve the questionnaire according to their opinions, comments and guidance on the selection of measurement items, the expression of measurement items, the design of the questionnaire and so on, including but not limited to the following main concerns: Is the question easy to understand? Is the problem stated clearly and concisely? Is the problem consistent with the actual situation? Are there any items that will not be answered honestly? Are there any suggestive problems? Are there too many questions? On this basis, necessary modifications are made to the question quantity, sentences, words and question design of the questionnaire. To ensure that the respondents are able to answer the questions, some question items are changed into reverse questions to form the questionnaire for the pilot survey.

5.4.2 Small-Scale Pilot Test

To improve the data quality of the research, this paper intends to use small-sample data for a pilot-test and further purify the measurement items by analysing the reliability and validity of the variable scale. The questionnaire is an electronic questionnaire, and the software tool Questionnaire Star is applied to design the question type. It supports data collection through WeChat, e-mail or SMS. After data recovery, classified statistics can be obtained and cross-analysis can be performed. The data can be downloaded and exported to word, Excel, SPSS and so on. A total of 63 questionnaires are collected, all of which are valid.

5.5 Initial Scale Reduction

When evaluating the initial scale, it is necessary first to give the descriptive statistics of each construct and measurement item and then to conduct factor analysis, mainly to calculate the construct reliability, including the Cronbach's α and item-to-total correlation indicators. The initial scale should be reduced according to the two principles of whether Cronbach's α is greater than 0.7 (Nunnally, 1967) and item-to-total correlation is greater than 0.5 (Park and Kim, 2003).

5.5.1 Initial Scale Reduction of Software Failure

As shown in Table 5-8, the Cronbach's α value of three software failure measurement items is 0.925, which is greater than 0.7, and the item-to-total correlation of each measurement item is greater than 0.5, indicating that there is good internal consistency among the three measurement items of software failure and the software failure is measured well.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Poor stability of software products	3.159	1.035	.837	.903	0.925
Poor maintainability of software products	2.921	0.938	.863	.878	
Poor customer satisfaction with products	2.968	0.915	.846	.893	

Table 5-8 Factor Analysis of the Software Failure Initial Scale

5.5.2 Initial Scale Reduction of Inadequate Project Management

As shown in Table 5-9, the Cronbach's α value of 10 measurement items for inadequate project management is 0.896, which is greater than 0.7, and the item-to-total correlation of each measurement item is greater than 0.5, indicating that good internal consistency exists for the 10 measurement items of inadequate project management, showing that the construct is measured well.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Poor product quality control	4	0.916	0.535	0.892	0.896
Poor cost control	4.095	0.995	0.633	0.886	
Insufficient internal communication	4.397	0.871	0.619	0.887	
Lack of effective customer communication	4.175	0.871	0.537	0.892	
Insufficient prior risk review	4.095	0.979	0.702	0.881	
Lack of effective knowledge management	3.905	0.962	0.74	0.878	
Lack of product planning	3.984	1.024	0.768	0.876	
Application of an advanced but immature technical proposal	4.095	1.027	0.684	0.882	
Development process is not standardized or scientific enough	3.921	1.126	0.603	0.889	
Lack of project experience	4.476	0.78	0.591	0.889	

Table 5-9 Factor Analysis of the Inadequate Project Management Initial Scale

5.5.3 Initial Scale Reduction of Insufficient Software Testing

As presented in Table 5-10, the Cronbach's α value of five measurement items is 0.807, which is greater than 0.7, and the item-to-total correlation of each measurement item is greater than 0.5, indicating that the five measurement items of insufficient software testing have good internal consistency and that the measurement item of insufficient software testing is measured well.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α

Insufficient testing of the development system	3.429	1.187	0.601	0.769	0.807
Software product is complex and quickly updated	3.127	0.942	0.644	0.756	
Many and disorderly ways of downloading software	3.143	0.981	0.674	0.746	
Testing environment not allowed	3.381	1.113	0.557	0.781	
Complex site environment	3.333	1	0.51	0.793	

Table 5-10 Factor Analysis of the Insufficient Software Testing Initial Scale

5.5.4 Initial Scale Reduction of Employees' Negative Characteristics

As shown in Table 5-11, the Cronbach's α value of the seven measurement items of employees' negative characteristics is 0.901, which is greater than 0.7, but the item-to-total correlation of the measurement item "poor employee enthusiasm" is less than 0.5. After deleting this measurement item, the Cronbach ' α value will increase, so it is deleted.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Developers lack experience and have poor development habits	3.175	1.1	0.778	0.877	0.901
The level of developers is uneven	3.095	1.027	0.88	0.866	
High employee mobility	3.302	1.072	0.655	0.892	
The tasks of project personnel are complicated	3.317	1.029	0.69	0.888	
Lack of core talents	3	1.15	0.794	0.875	
Poor incentive mechanism for employees	3.222	1.007	0.698	0.887	
Poor employee enthusiasm	3	0.967	0.464	0.911	

Table 5-11 Factor Analysis of the Employees' Negative Characteristics Initial Scale

5.5.5 Initial Scale Reduction of Customer Requirement Defects

As shown in Table 5-12, the Cronbach's α value of the five measurement items of customer requirement defects is 0.766, which is greater than 0.7, and the item-to-total correlation of the measurement item "insufficient active participation of users" is less than 0.5. After deleting this measurement item, the Cronbach' α value will increase, so it is deleted.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Customers' requirements constantly changing	3.603	1.025	0.598	0.702	0.766
Unclear customers' requirements	3.206	1.152	0.655	0.678	
Insufficient active participation of users	3	0.967	0.22	0.819	
No consensus of customers' leaders	3.143	0.913	0.526	0.729	
Requirements not clearly defined	3.317	1.133	0.713	0.654	

Table 5-12 Factor Analysis of the Customer Requirements Defects Initial Scale

5.5.6 Initial Scale Reduction of Uncertain External Environment

As shown in Table 5-13, the Cronbach's α value of six measurement items for an uncertain external environment is 0.872, which is greater than 0.7, and the item-to-total correlation of each measurement item is greater than 0.5, indicating that good internal consistency exists among the six measurement items and the uncertain external environment is measured well.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α

Lack of project support from senior management	2.619	0.958	0.724	0.841	0.872
No harmonious relationship between leaders and employees of the company	2.492	0.914	0.614	0.86	
Excessive leadership intervention (project manager)	2.619	0.831	0.612	0.86	
The government or another institution have imposed restrictions on the project development	2.746	0.999	0.774	0.831	
New industry standards or laws and regulation have been issued recently	2.841	0.919	0.715	0.842	
Unpredictable market turmoil	3.032	0.897	0.596	0.863	

Table 5-13 Factor Analysis of the Uncertain External Environment Initial Scale

5.5.7 Questionnaire Formation of This Study

Through the pilot test and factor analysis of each measurement model, some measurement items are deleted, and finally the questionnaire of this study is optimized and upgraded. The survey questions are divided into seven parts: basic personal information, inadequate project management, insufficient software testing, employees' negative characteristics, customer requirement defect, uncertain external environment, and software failure. There are 40 question items in total, of which four are objective questions and the remaining 36 questions are used to measure the six constructs involved in this research model. See Appendix 5-2 for the questionnaire.

5.6 Summary

Based on the identification and analysis of the influencing factors of PIS software failure, this chapter constructs a research model and puts forward the research hypotheses. The scale is developed accordingly: first, the scale development process and the focus of each stage are introduced; then, the initial measurement items of the main variables in the research model are introduced; and finally, they are tested with

small samples to form the questionnaire of this paper.

Chapter 6. Analysis of the Empirical Research Process and Result

This chapter focuses on the process and results of the data analysis. Firstly, the questionnaire collection is described; then, the measurement model is evaluated; subsequently, a confirmatory factor analysis is carried out regarding the structure of the latent variables and their observation variables; and, finally, the structural equation model is evaluated, a process that involves assessing the model's explanatory power and testing the hypotheses.

6.1 Basic Information on the Questionnaire Collection

Because this paper studies the failure of PIS software, the most appropriate person to complete the questionnaire should be the person in charge of the software development projects. However, in the actual research, it is very difficult to achieve this because the PIS industry is specialized, consisting of "relatively small segments", so this paper expands the survey respondents to people who have certain software development project experience or know the whole software development project well, such as project managers, developers and implementation consultants.

6.1.1 Data Collection

The large-scale questionnaire of this study officially began in middle of August 2021 and lasted for about one week. Questionnaire Star was mainly used to collect the data needed in this study. To improve the effectiveness and reliability of the recovery questionnaire, the researcher personally distributed and collected the questionnaire through WeChat and Questionnaire Star. However, due to the limited interpersonal scope of the researcher, some questionnaires were distributed by the project manager or colleagues to the project members around them.

The sample size requirements of a large-scale questionnaire distribution and small-scale pilot tests are different. For small-scale pilot tests, there is no unified standard for the number of samples. The widely used standard for the large-scale questionnaire distribution is that of Gorsuch. He believed that the sample size should ensure that the

ratio between the measured items and the number of returned questionnaires should be at least 1:5 and preferably 1:10. When using structural equation analysis, the sample size is related to the measurement items, free parameters and other factors. The freer the parameters, the larger the sample size. However, with an increase in the sample size, some model judgment coefficients will become inappropriate. Therefore, it is necessary to refer to more parameters to determine the estimation and fitting of the model at the same time.

Providing enough research data, 341 completed questionnaires were received within the specified time period through WeChat, mobile phone and network link. They were deleted according to the response speed, those with a response time of less than 100 seconds being removed. A total of 313 valid questionnaires remained, with an effective rate of 91.8%. The questionnaire response channels are shown in Figure 6-1:

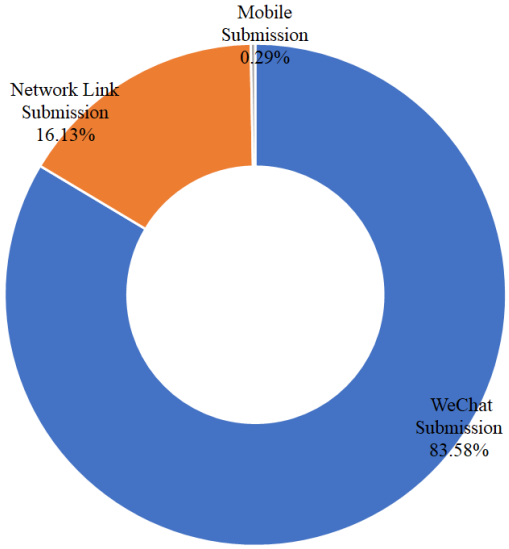


Figure 6-1 Questionnaire Recovery Channels

The regional distribution of the questionnaire respondents shown in Figure 6-2:

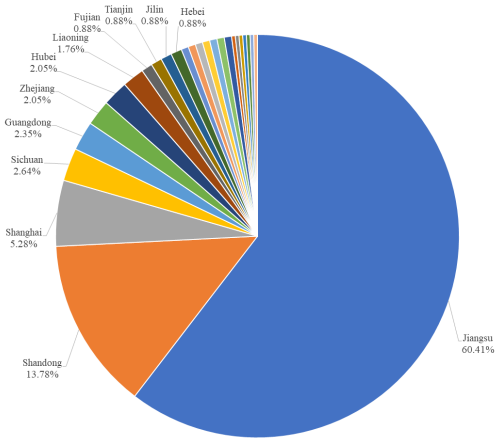


Figure 6-2 Regional Distribution of the Questionnaire Respondents

The figure shows that the technicians who filled in the questionnaire are mainly distributed across Jiangsu, Shandong, Shanghai, Sichuan, Guangdong, Hubei, Zhejiang, Liaoning, Hebei and other provinces. In short, from the perspective of coverage, the questionnaire of this study is representative.

6.1.2 Descriptive Statistical Analysis of the Samples

The sample descriptive statistical analysis in this study is mainly carried out for the demographic characteristic variables, and the results are shown in Table 6-1. In terms of the time spent in the software industry, most project members have been engaged in it for 1–5 years, accounting for 64.3%, but only about 12% have been engaged in the software industry for more than 10 years, which is related to the high mobility of personnel in the software industry; The duration of projects is generally 1–3 years, accounting for about 67%. Generally, project teams contain fewer than 10 people. As can be seen from the positions held, developers account for the largest proportion of software development projects, followed by other personnel. Here, “other personnel” refers to relevant personnel who provide a project with business support or R&D support. These personnel also have a certain understanding of the project situation, and there are not many implementation consultants among them or even in practical operation.

Variable	Item	Frequency	Percentage	Average Value	Standard Deviation
Work	< 1 year	80	25.60%	2.23	0.97

experience	1–5 years	121	38.70%		
	5–10 years	72	23.00%		
	>10 years	40	12.80%		
Duration of projects	< 1 year	92	29.40%	2.22	1.06
	1–3 years	117	37.40%		
	3–5 years	47	15.00%		
	>5 years	57	18.20%		
Team size	Fewer than 5 persons	89	28.40%	2.31	1.12
	5–10 persons	110	35.10%		
	10–15 persons	41	13.10%		
	More than 15 persons	73	23.30%		
Position	Project (implementation)			3.59	1.20
	Supervisors	18	5.80%		
	Project managers	29	9.30%		
	Developers	121	38.70%		
	QC engineers	41	13.10%		
	Others	104	33.20%		

Table 6-1 Frequency Analysis of the Demographic Variables

According to the above analysis results, the numerical characteristics of the demographic variables reflect the distribution of the respondents, in which the mean represents the central tendency and the standard deviation represents the fluctuation.

The frequency analysis results of each variable show that the distribution basically meets the requirements of a sampling survey. For example, according to the survey results, project (implementation) director accounts for 5.8% of the respondents, project managers represent 9.3%, developers account for 38.7%, quality control personnel make up 13.1%, and other personnel constitute 33.2%. The results of this survey mainly represent the views of R&D personnel.

6.2 Evaluation of the Measurement Model

6.2.1 Reliability Test

The reliability of the measurement model is tested using Cronbach's alpha, which measure the internal consistency between variables, and composite reliability. When both the Cronbach's alpha and the composite reliability are greater than 0.7, the measurement model has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of "Cronbach's α after Deleting Item". When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the "CITC value". The CITC values of all measured items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of "customer requirement defects" has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.8, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Requirements not clearly defined	3.319	1.132	0.710	0.757	0.828
Unclear customers' requirements	3.275	1.104	0.753	0.736	
Customers' requirements constantly changing	3.866	1.051	0.593	0.810	
No consensus of customers' leaders	3.217	1.07	0.568	0.821	

Table 6-2 Measurement Model Evaluation of Customer Requirement Defects

It can be seen from Table 6-3 that the composite reliability of the six observation variables of "employees' negative characteristics" is 0.869, which is greater than 0.7,

and the internal consistency between the observation variables is high, indicating that the measurement model of “employees’ negative characteristics” has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of " Cronbach’s α after Deleting Item ". When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the “CITC value”. The CITC values of all measured items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of “employees’ negative characteristics” has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.8, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach’s α after Deleting Item	Cronbach’s α
Developers lack experience and have poor development habits	3.326	1.017	0.594	0.859	0.869
The level of developers is uneven	3.163	1.039	0.704	0.840	
The tasks of project personnel are complicated	3.450	1.094	0.716	0.837	
High employee mobility	3.224	1.057	0.716	0.838	
Lack of core technical talents	3.508	1.019	0.693	0.842	
Poor incentive mechanism for employees	3.345	1.054	0.579	0.862	

Table 6-3 Measurement Model Evaluation of Employees’ Negative Characteristics

It can be seen from Table 6-4 that the composite reliability of the five observation variables of “insufficient software testing” is 0.772, which is greater than 0.7. The internal consistency between the observation variables is high, indicating that the

measurement model of “insufficient software testing” has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of " Cronbach’s α after Deleting Item ". When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the “CITC value”. The CITC value of “many and disorderly ways of downloading software” in Table 6-4 is 0.423, which is less than 0.5 and needs to be deleted. The corresponding CITC values of other items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of “insufficient software testing” has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.7, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach’s α after Deleting Item	Cronbach’s α
Insufficient testing	3.409	1.146	0.633	0.697	0.772
Testing environment not allowed	3.406	0.970	0.543	0.731	
Complex site commissioning environment	3.508	1.019	0.588	0.717	
Application of advanced but immature technical proposal	2.866	1.007	0.545	0.730	
Many and disorderly ways of downloading software	2.856	1.084	0.423	0.771	

Table 6-4 Measurement Model Evaluation of Insufficient Software Testing

It can be seen from Table 6-5 that the composite reliability of nine observation variables with “inadequate project management” is 0.916, which is greater than 0.7, and the

internal consistency between observation variables is high, indicating that the measurement model with “inadequate project management” has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of “Cronbach’s α after Deleting Item”. When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the “CITC value”. The CITC values of all measured items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of “inadequate project management” has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.9, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach’s α after Deleting Item	Cronbach’s α
Poor product quality control	3.169	1.056	0.666	0.910	0.916
Poor cost control	2.917	0.93	0.600	0.913	
Difficult project management and coordination	3.080	0.959	0.651	0.910	
Lack of project management experience	2.958	0.988	0.701	0.907	
Insufficient prior risk review	3.272	1.065	0.786	0.901	
Lack of effective requirement change management	3.272	1.086	0.705	0.907	
Development process is not standardized or scientific enough	2.917	1.012	0.719	0.906	
Lack of effective knowledge management	2.920	1.076	0.724	0.906	
Lack of product planning	3.137	1.105	0.795	0.900	

Table 6-5 Measurement Model Evaluation of Inadequate Project Management

It can be seen from Table 6-6 that the composite reliability of six observation variables regarding the item “uncertain external environment” is 0.887, which is greater than 0.7, and the internal consistency between observation variables is high, indicating that the measurement model of “uncertain external environment” has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of “Cronbach’s α after Deleting Item”. When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the “CITC value”. The CITC values of all measured items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of “uncertain external environment” has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.8, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach’s α after Deleting Item	Cronbach’s α
Lack of project support from senior management	2.649	0.98	0.605	0.884	0.887
No harmonious relationship between leaders and employees of the company	2.339	0.899	0.688	0.87	
Excessive leadership intervention (project manager)	2.482	0.859	0.713	0.866	
The government or another institution has imposed restrictions on the project development	2.591	0.937	0.799	0.851	

New industry standards or laws and regulation have been issued recently	2.62	0.905	0.77	0.857	
Unpredictable market turmoil	2.84	0.944	0.65	0.876	

Table 6-6 Measurement Model Evaluation of Uncertain External Environment

It is apparent from Table 6-7 that the composite reliability of the five observation variables of "software failure" is 0.891, which is greater than 0.7, and the internal consistency between the observation variables is high, indicating that the measurement model of "software failure" has high reliability. Whether a measurement item needs to be deleted or not mainly depends on the value of "Cronbach's α after Deleting Item". When an item is deleted, the reliability coefficient will not increase significantly, indicating that the item needs to be retained and should not be deleted. In addition, the further deletion of measurement items also needs to consider the correlation between them, which is mainly determined according to the "CITC value". The CITC values of all measured items are greater than 0.5 and do not need to be deleted, which indicates that the correlation between the measurement items in the questionnaire is good. Therefore, the measurement model of "software failure" has a good reliability level. It can be seen that the reliability quality of the research data is high, and the values of its reliability coefficients are greater than 0.8, which meets the requirements of further analysis and satisfies the conditions of following in-depth research.

Measurement Item	Average Value	Standard Deviation	Corrected Item–Total Correlation	Cronbach's α after Deleting Item	Cronbach's α
Software product is complex and quickly updated	2.904	1.030	0.676	0.881	0.891
Hardware problem resulting in software function change	2.649	1.131	0.699	0.877	
Poor stability of software products	3.003	1.079	0.780	0.857	
Poor maintainability of software products	2.748	1.011	0.782	0.857	

Poor customer satisfaction with products	2.837	0.969	0.746	0.866	

Table 6-7 Measurement Model Evaluation of Software Failure

6.2.2 Exploratory Factor Analysis

Following the arguments and details in the preceding section, there are 34 effective measurement items. Before conducting the exploratory factor analysis, the validity of the samples is analyzed, obtaining the results shown in Table 6-8.

KMO Measure of Sampling Adequacy		0.938
Bartlett's Test of Sphericity	Approx. chi square	7463.569
	Freedom	561
	Significance	0.000

Table 6-8 KMO and Bartlett's Test Results

In Table 6-8, the coefficient result of the KMO test is 0.938, and the standard coefficient value range of the KMO test is between 0 and 1. The closer it is to 1, the better the validity of the questionnaire is. Only if this has a value greater than 0.6 are the premise requirements of factor analysis satisfied and only then can exploratory factor analysis be performed. According to the sphericity test, the significance is infinitely close to 0, so the null hypothesis is rejected. The data pass the Bartlett sphericity test ($P < 0.05$), indicating that the research data are suitable for further factor analysis.

Following the exploratory factor analysis steps, the total variance explained results are obtained as shown in Table 6-9.

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings			Rotation Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	14.267	41.962	41.962	14.267	41.962	41.962	5.373	15.802	15.802
2	2.623	7.715	49.677	2.623	7.715	49.677	4.438	13.052	28.854

3	2.166	6.371	56.048	2.166	6.371	56.048	3.866	11.371	40.225
4	1.567	4.61	60.657	1.567	4.61	60.657	3.805	11.191	51.417
5	1.197	3.521	64.178	1.197	3.521	64.178	2.943	8.655	60.072
6	1.064	3.131	67.309	1.064	3.131	67.309	2.461	7.238	67.309
7	0.972	2.859	70.168	-	-	-	-	-	-
8	0.783	2.302	72.47	-	-	-	-	-	-
9	0.748	2.201	74.671	-	-	-	-	-	-
10	0.688	2.023	76.694	-	-	-	-	-	-
11	0.63	1.852	78.547	-	-	-	-	-	-
12	0.588	1.73	80.276	-	-	-	-	-	-
13	0.523	1.538	81.814	-	-	-	-	-	-
14	0.505	1.486	83.3	-	-	-	-	-	-
15	0.498	1.466	84.766	-	-	-	-	-	-
16	0.457	1.344	86.111	-	-	-	-	-	-
17	0.424	1.248	87.359	-	-	-	-	-	-
18	0.411	1.209	88.568	-	-	-	-	-	-
19	0.391	1.151	89.718	-	-	-	-	-	-
20	0.356	1.047	90.765	-	-	-	-	-	-
21	0.327	0.963	91.728	-	-	-	-	-	-
22	0.305	0.898	92.626	-	-	-	-	-	-
23	0.296	0.872	93.498	-	-	-	-	-	-
24	0.277	0.815	94.313	-	-	-	-	-	-
25	0.262	0.769	95.083	-	-	-	-	-	-
26	0.254	0.746	95.829	-	-	-	-	-	-
27	0.234	0.689	96.518	-	-	-	-	-	-
28	0.219	0.645	97.163	-	-	-	-	-	-
29	0.205	0.601	97.764	-	-	-	-	-	-
30	0.18	0.529	98.293	-	-	-	-	-	-

31	0.156	0.46	98.753	-	-	-	-	-	-
32	0.155	0.455	99.208	-	-	-	-	-	-
33	0.141	0.415	99.623	-	-	-	-	-	-
34	0.128	0.377	100	-	-	-	-	-	-

Table 6-9 Total Variance Explained

It can be seen from Table 6-9 that six factors are extracted by the factor analysis, and the eigenvalues are greater than 1. These six factors' percentages of variance after rotation are 15.802%, 13.052%, 11.371%, 11.191%, 8.655% and 7.238% respectively, and the cumulative percentage after rotation reaches 67.309%.

The maximum variance rotation method (varimax) is used for rotation. The results of information extraction of factors for research items and the corresponding relationship between factors and research items are shown in Table 6-10.

Item	Factor (Loading Coefficient)						Common Degree (Common Factor Variance)
	1	2	3	4	5	6	
Requirements are not clearly defined	0.26	0.182	0.146	0.078	0.786	0.14	0.765
Unclear customers' requirements	0.226	0.158	0.142	0.048	0.81	0.211	0.799
Customers' requirements are constantly changing	-0.063	0.049	0.011	0.421	0.699	0.142	0.693
No consensus of customers' leaders	-0.026	0.237	0.149	0.151	0.681	0.031	0.567
Insufficient testing	0.299	0.123	-0.027	0.264	0.125	0.739	0.736
Testing environment not allowed	0.121	0.221	-0.048	0.324	0.27	0.625	0.634
Complex site commissioning environment	0.192	0.325	0.199	0.245	0.262	0.476	0.537
Application of advanced but immature technical proposal	0.27	0.423	0.214	-0.076	0.113	0.556	0.625
Development process is not standardized or scientific enough	0.434	0.527	0.171	0.104	0.081	0.426	0.694
Poor product quality control	0.326	0.565	0.151	0.116	0.145	0.302	0.575

Poor cost control	0.148	0.712	0.271	-0.003	0.172	0.077	0.637
Difficult project management and coordination	0.111	0.669	0.185	0.217	0.09	0.255	0.615
Lack of project management experience	0.297	0.615	0.095	0.209	0.317	0.076	0.626
Lack of effective requirements change management	0.219	0.595	0.186	0.401	0.196	0.091	0.644
Insufficient prior risk review	0.435	0.577	0.076	0.375	0.159	0.119	0.708
Lack of product planning	0.459	0.597	0.057	0.301	0.146	0.158	0.707
Lack of effective knowledge management	0.593	0.475	0.022	0.305	0.084	0.17	0.706
Poor incentive mechanism for employees	0.363	0.392	-0.009	0.492	0.141	0.095	0.557
The tasks of project personnel are complicated	0.312	0.212	0.039	0.695	0.228	0.053	0.682
The level of developers is uneven	0.472	0.139	0.09	0.622	0.164	0.144	0.685
High employee mobility	0.379	0.118	0.166	0.699	0.062	0.106	0.69
Lack of core technical talents	0.006	0.23	0.274	0.692	0.178	0.262	0.706
Developers lack experience and poor development habits	-0.047	0.132	0.458	0.579	0.174	0.317	0.696
No harmonious relationship between leaders and employees of the company	0.604	0.198	0.526	0.076	0.095	0.031	0.697
Excessive leadership intervention (project manager)	0.359	0.125	0.656	0.119	0.044	0.168	0.62
Lack of project support from senior management	0.516	0.203	0.437	0.192	-0.025	0.202	0.576
The government or another institution has imposed restrictions on the	0.242	0.13	0.841	0.047	0.096	0.034	0.796

project development							
New industry standards or laws and regulations have been issued recently	0.173	0.106	0.856	0.068	0.115	-0.005	0.791
Unpredictable market turmoil	0.072	0.178	0.771	0.198	0.159	-0.017	0.695
Software product is complex and quickly updated	0.572	0.358	0.091	0.366	0.021	0.219	0.647
Hardware problem resulting in software function change	0.684	0.252	0.177	0.24	0.018	0.104	0.632
Poor stability of software products	0.717	0.213	0.177	0.142	0.131	0.313	0.726
Poor maintainability of software products	0.722	0.217	0.325	0.083	0.155	0.177	0.736
Poor customer satisfaction with products	0.733	0.175	0.267	0.134	0.148	0.073	0.685

Table 6-10 Factor Loading Coefficients after Rotation

Table 6-10 reflects that the commonality of “unclear customers’ requirements” is the highest at a value of 0.799, and that of “complex site commissioning environment” is the lowest at a value of 0.537, which is greater than 0.4. The commonality values of all measurement items are higher than 0.4, meeting the requirements. It can be seen that the correlation between measurement items and factors is strong, showing that the effect of factor analysis is good and can effectively extract information.

6.2.3 Confirmatory Factor Analysis

After the above reliability test and exploratory factor analysis, the retained measurement indicators are divided into 6 factors and 34 measurement indicators/items. In order to further verify the rationality of PIS software failure scale, confirmatory factor analysis (CFA) will be carried out on the measured items. The convergent validity and discriminant validity of the scale were tested in the process of confirmatory factor analysis. 313 effective samples are selected in this paper, which is more than five times but less than 10 times the number of analysis items. They can basically be used for empirical analysis. Stable results can be obtained either by increasing the sample size or

by removing some analysis items. Using confirmatory factor analysis, the factor loading coefficient can be obtained, as shown in Table 6-11.

Factor (Latent Variable)	Measurement Item (Obvious Variable)	Non-standard Load Coef.	Std Error	z (CR Value)	p	Std Estimate
Factor 1	Requirements are not clearly defined	1	-	-	-	0.894
Factor 1	No consensus of customers' leaders	0.565	0.056	10.082	0	0.534
Factor 1	Customers' requirements are constantly changing	0.583	0.054	10.729	0	0.562
Factor 1	Unclear customers' requirements	1.007	0.049	20.753	0	0.923
Factor 2	Insufficient testing	1	-	-	-	0.718
Factor 2	Application of advanced but immature technical proposal	0.753	0.076	9.913	0	0.615
Factor 2	Complex site commissioning environment	0.813	0.074	11.051	0	0.69
Factor 2	Testing environment not allowed	0.969	0.087	11.084	0	0.692
Factor 3	Poor product quality control	1	-	-	-	0.69
Factor 3	Lack of effective knowledge management	1.157	0.089	12.959	0	0.783
Factor 3	Lack of product planning	1.285	0.092	13.925	0	0.847
Factor 3	Insufficient prior risk review	1.231	0.089	13.845	0	0.841
Factor 3	Development process is not standardized or scientific enough	1.062	0.084	12.664	0	0.764
Factor 3	Lack of effective requirement change management	1.112	0.09	12.378	0	0.745
Factor 3	Difficult project management and coordination	0.863	0.079	10.961	0	0.655
Factor 3	Lack of project management experience	0.978	0.082	12	0	0.721

Factor 3	Poor cost control	0.761	0.076	10.006	0	0.596
Factor 4	Poor incentive mechanism for employees	1	-	-	-	0.669
Factor 4	Developers lack experience and have poor development habits	0.923	0.091	10.142	0	0.64
Factor 4	Lack of core technical talents	1.037	0.093	11.209	0	0.718
Factor 4	Complicated tasks of project personnel	1.196	0.1	11.91	0	0.771
Factor 4	High employee mobility	1.155	0.097	11.915	0	0.771
Factor 4	Uneven level of developers	1.156	0.096	12.089	0	0.785
Factor 5	No harmonious relationship between leaders and employees of the company	1	-	-	-	0.717
Factor 5	Unpredictable market turmoil	1.06	0.086	12.273	0	0.724
Factor 5	New industry standards or laws and regulations have been issued recently	1.187	0.083	14.297	0	0.845
Factor 5	The government or another institution has imposed restrictions on the project development	1.268	0.086	14.725	0	0.873
Factor 5	Lack of project support from senior management	0.994	0.09	11.082	0	0.654
Factor 5	Excessive leadership intervention (project manager)	0.979	0.079	12.454	0	0.734
Factor 6	Software product is complex and quickly updated	1	-	-	-	0.722
Factor 6	Poor customer satisfaction with products	1.053	0.076	13.942	0	0.808
Factor 6	Poor maintainability of software products	1.165	0.079	14.773	0	0.857

Factor 6	Poor stability of software products	1.226	0.084	14.587	0	0.846
Factor 6	Hardware problem resulting in software function change	1.111	0.088	12.572	0	0.731

Table 6-11 Factor Loadings' Coefficients

Regarding the measurement relationship, for "no consensus of customers' leaders", measuring Factor 1, the absolute value of the standardized loading coefficient is 0.534 < 0.6, which means that the measurement relationship is weak. It can be considered necessary to remove this measurement relationship and repeat the analysis. For "customers' requirements are constantly changing", measuring Factor 1, the absolute value of the standardized loading coefficient is 0.562 < 0.6, which means that the measurement relationship is weak. This measurement relationship should be removed and the analysis performed again. For "poor cost control", measuring Factor 3, the absolute value of the standardized load coefficient is 0.596 < 0.6, which means that the measurement relationship is weak, so this measurement relationship should be removed and the analysis runs again. The requirements of the convergent validity test under normal circumstances are that the AVE value should be greater than 0.5 and the CR value should be greater than 0.7. After trial analyses of first removing the "application of advanced but immature technical proposal" in Factor 2 and then removing "difficult project management and coordination" in Factor 3, the AVE and CR index results and discriminant validity of the model are finally obtained as shown in Tables 6-12 and 6-13.

Factor	Average Variance Extraction (AVE) Value	Composite Reliability (CR) Value
Factor 1	0.835	0.91
Factor 2	0.520	0.762
Factor 3	0.600	0.912
Factor 4	0.532	0.871
Factor 5	0.578	0.890
Factor 6	0.628	0.893

Table 6-12 AVE and CR Index Results of the Model

Table 6-12 demonstrates that the AVE values corresponding to the six factors analysed in this paper are all greater than 0.5, and the CR values are all higher than 0.7, meaning that the analysis data have good convergent validity.

	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6
Factor 1	**0.914**					
Factor 2	0.467	**0.721**				
Factor 3	0.510	0.656	**0.774**			
Factor 4	0.446	0.613	0.707	**0.729**		
Factor 5	0.379	0.348	0.550	0.513	**0.760**	
Factor 6	0.409	0.559	0.769	0.637	0.631	**0.792**

Table 6-13 Discriminant Validity: Pearson Correlation and Square Root of AVE

Discriminant validity is generally evaluated by the square root of AVE, which must be greater than the correlation coefficient between this factor and other factors (Fornell and Larcker, 1981). It can be seen from the results in Table 6-13 that for Factor1, the square root value of AVE is 0.914, which is greater than the maximum value of correlation coefficient between Factor 1 and other factors of 0.510, indicating that the discriminant validity of this factor is good. Similarly, the square root values of AVE of Factor 2, Factor 3, factor 4, factor 5 and factor 6 are 0.721, 0.774, 0.729, 0.760 and 0.792 respectively, which are greater than their correlation coefficients corresponding to other factors, indicating that they all have good discriminant validity.

The measurement models of the six latent variables of the model studied in this paper have high reliability and validity, and there is a good measurement relationship between the observed variables and the corresponding latent variables. The results of the confirmatory factor analysis show that the factor structure resultant force of the latent variables and the relationship between the latent variables and their observed variables are accurate. Therefore, after deleting measurement items, the structural equation can be fitted.

6.3 Structural Model Evaluation

6.3.1 Evaluation of the Model's Explanatory Power

The model is established according to the relationships in the above hypothetical model, it is fitted with 313 effective sample data using SmartPLS software, the standardized regression coefficient and path coefficient of each path are calculated, and the significance of the path coefficients of the structural model is tested with the bootstrap algorithm (N 500) to understand the causal relationship between different variables. The significance test results of the model fitting and path coefficients are shown in Figure 6-3.

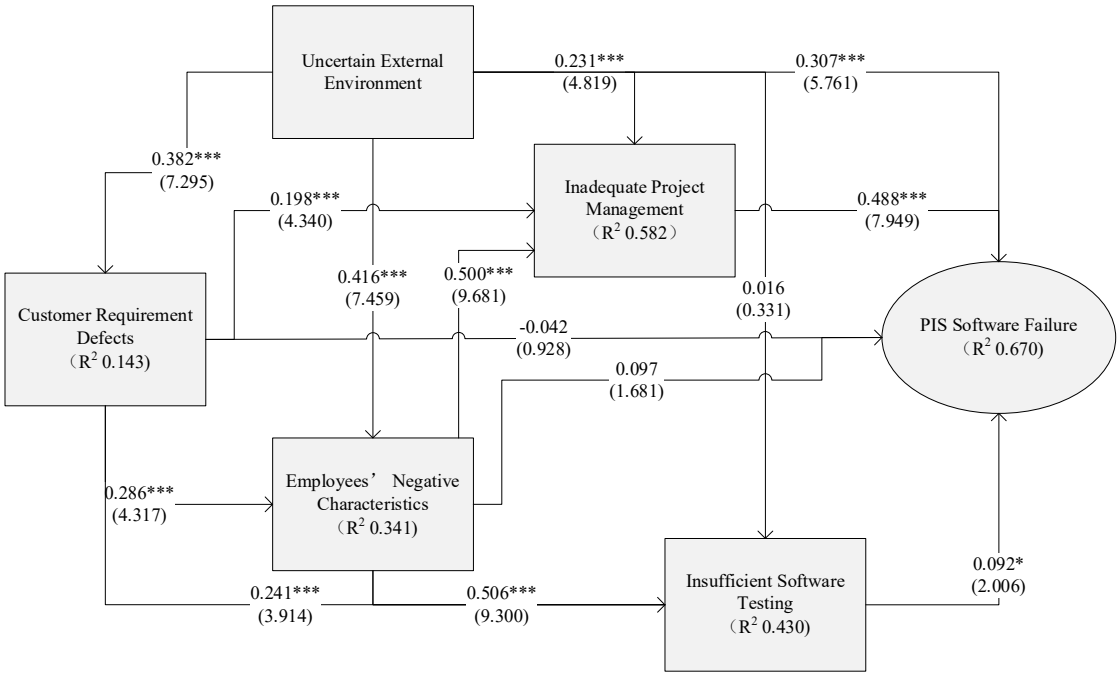


Figure 6-3 Path Coefficients and R^2 Values of the Research Model

In this paper, the explanatory power of the model is tested using the square value of complex correlation (R^2), and the degree of variance interpretation of the endogenous latent variables in the structural equation model is evaluated. As shown in Figure 6-3, the comprehensive impact R^2 on PIS software failure in the model reaches 0.670, explaining 67% of the PIS software failure variance; the comprehensive impact R^2 of the exogenous latent variables on inadequate project management is 0.582, explaining 52.8% of the variance of inadequate project management; the comprehensive influence

R^2 of the exogenous latent variables on insufficient software testing is 0.430, which explains 43% of the variance of insufficient software testing; the comprehensive impact R^2 of the exogenous latent variables on employees' negative characteristics is 0.341, which explained 34.1% of the variance of employees' negative characteristics; and the comprehensive impact R^2 of the exogenous latent variables on customer requirement defects is 0.143, which explains 14.3% of the variance of customer requirement defects. All the latent variables are fully explained, which ensures the reliability and accuracy of the research results.

6.3.2 Hypothesis Test

In this paper, the bootstrap algorithm (N 500) is used to test the significance of the path coefficients between latent variables. The null hypothesis, standardized path coefficient, T value and hypothesis test results obtained using the SmartPLS software are listed in Table 6-14.

It can be seen from Table 6-14 that the path coefficient between inadequate project management and PIS software failure is 0.488, and the T value is 7.949. Therefore, the impact of inadequate project management on PIS software failure is significant, and H1 is established at the statistically significant level of 0.001 per cent. The path coefficient between insufficient software testing and PIS software failure is 0.092, and the T value is 2.006. Therefore, the impact of insufficient software testing on PIS software failure is significant, and H2 is valid at the statistically significant level of 0.05 per cent. The path coefficient between employees' negative characteristics and PIS software failure is 0.097, and the T value is 1.681 < 1.96. Therefore, the impact of employees' negative characteristics on PIS software failure is not significant, and H3 is not supported. The path coefficient between employees' negative characteristics and insufficient software testing is 0.506, and the T value is 9.300. Hence, the impact of employees' negative characteristics on insufficient software testing is statistically significant, and H4 is established at the statistically significant level of 0.001 per cent. The path coefficient between employees' negative characteristics and inadequate project management is 0.500, and the T value is 9.681. Thus, the impact of employees' negative characteristics

on inadequate project management is significant, and H5 is established at the statistically significant level of 0.001 per cent. The path coefficient between customer requirement defects and PIS software failure is -0.042, and the T value is 0.928 < 1.96. Therefore, the impact of customer requirement defects on PIS software failure is not significant, and H6 is not supported. The path coefficient between customer requirements defects and inadequate project management is 0.198, and the T value is 4.340. Therefore, the impact of customer requirements defects on inadequate project management is significant, and H7 is established at the statistically significant level of 0.001 per cent. The path coefficient between customer requirements defects and employees' negative characteristics is 0.286, and the T value is 4.317. Therefore, the impact of customer requirement defects on employees' negative characteristics is significant, and H8 is established at the statistically significant level of 0.001 per cent. The path coefficient between customer requirement defects and insufficient software testing is 0.241, and the T value is 3.914. Therefore, the impact of customer requirement defects on insufficient software testing is significant, and H9 is valid at the statistically significant level of 0.001 per cent. The path coefficient between the uncertain external environment and PIS software failure is 0.307, and the T value is 5.761. Hence, the impact of an uncertain external environment on PIS software failure is significant, and H10 is established at the statistically significant level of 0.001 per cent. The path coefficient between an uncertain external environment and the inadequate project management is 0.231, and the T value is 4.819. Therefore, the impact of an uncertain external environment on inadequate project management is significant, and H11 is established at the statistically significant level of 0.001 per cent. The path coefficient between an uncertain external environment and employees' negative characteristics is 0.416, and the T value is 7.459. Consequently, the impact of an uncertain external environment on employees' negative characteristics is significant, and H12 is valid at the statistically significant level of 0.001 per cent. The path coefficient between an uncertain external environment and insufficient software testing is 0.016, and the T value is 0.331 < 1.96. Therefore, the impact of an uncertain external environment on

insufficient software testing is not significant, and H13 is not supported. The path coefficient between an uncertain external environment and customer requirement defects is 0.382, and the T value is 7.295. Thus, the impact of an uncertain external environment on customer requirement defects is significant, and H14 is valid at the statistically significant level of 0.001 per cent.

Hypothesis	**Proposition**	**Path Coefficient**	**T Value**	**Hypothesis Supported or Not**
H1	Inadequate project management is positively related to PIS software failure.	0.488	7.949***	Supported
H2	Insufficient software testing is positively correlated with PIS software failure.	0.092	2.006*	Supported
H3	Employees' negative characteristics are positively correlated with PIS software failure.	0.097	1.681	**Not Supported**
H4	Employees' negative characteristics are positively correlated with insufficient software testing.	0.506	9.300***	Supported
H5	Employees' negative characteristics are positively correlated with inadequate project management.	0.500	9.681***	Supported
H6	Customer requirement defects are positively correlated with PIS software failure.	-0.042	0.928	**Not Supported**
H7	Customer requirement defects are positively related to inadequate project management.	0.198	4.340***	Supported
H8	Customer requirement defects are positively correlated with employees' negative characteristics.	0.286	4.317***	Supported
H9	Customer requirement defects are positively correlated with insufficient software testing.	0.241	3.914***	Supported

H10	An uncertain external environment is positively related to PIS software failure.	0.307	5.761***	Supported
H11	An uncertain external environment is positively related to inadequate project management.	0.231	4.819***	Supported
H12	An uncertain external environment is positively correlated with the negative characteristics of employees.	0.416	7.459***	Supported
H13	An uncertain external environment is positively correlated with insufficient software testing.	0.016	0.331	**Not Supported**
H14	An uncertain external environment is positively correlated with customer requirement defects.	0.382	7.295***	Supported

Table 6-14 Structural Model

Note: *p<0.05; **p<0.01; ***p<0.001.

According to the hypothesis test results in Table 6-14, the relationships between an uncertain external environment, inadequate project management, insufficient software testing and PIS software failure are significant, showing that these are factors that directly affect PIS software failure. The relationship between customer requirement defects and PIS software failure is not significant, which indicates that customer requirement defects cannot lead directly to PIS software failure and that their role is realized indirectly through the impact on employees' negative characteristics, inadequate project management and insufficient software testing; this result also indicates that customer requirement defects are the driving cause of PIS software failure. The relationship between employees' negative characteristics and PIS software failure is also not significant, indicating that employees' negative characteristics will not directly cause the failure of PIS software and that their role is indirectly realized through the increase in inadequate project management and insufficient software testing caused by employees. It can be inferred that customer requirement defects and employees' negative characteristics do not directly cause the failure of PIS software but lead to an enhancement of inadequate project management and an increase in insufficient software

testing, which further lead to the failure of PIS software.

6.3.3 Influence Effect of Factors

The action path and influence effect of various factors on PIS software failure, according to the processing results of the SmartPLS software and the hypothesis test conclusions, are as shown in Table 6-15.

		Direct Effect	Indirect Effect	Total Effect
PIS Software Failure	← Customer Requirements Defects	-0.042	0.230	0.187
PIS Software Failure	← Employees' Negative Characteristics	0.097	0.290	0.388
PIS Software Failure	← Insufficient Software Testing	0.092		0.092
PIS Software Failure	← Inadequate Project Management	0.488		0.488
PIS Software Failure	← Uncertain External Environment	0.307	0.347	0.654

Table 6-15 Total Effects on PIS Software Failure

Notes: 1) The significance level is 0.05;

2) The standardized parameters are given in the table, and the direct effect is the path coefficient in the model.

The following conclusions can be drawn from Table 6-15:

(1) The uncertain external environment has the greatest total effect on the failure of PIS software (0.654), indicating that industry standards, government policies, market turbulence and the support of senior leaders are the most important to the on-board PIS of rail transit. The external environment is the primary consideration in the development of PIS software. Paying attention to industry standards, laws and regulations, government policies and the support of senior leaders is the most significant way to improve the failure rate of PIS software. By improving the market research ability, becoming familiar with industry trends and improving the professional level and attention of senior leaders, we can provide high-quality environmental conditions for PIS software development and achieve customers' satisfaction.

(2) The total effect of inadequate project management on PIS software failure (0.488) is greater than the total effect of employees' negative characteristics on PIS software failure (0.388), indicating that inadequate project management is also a significant

factor affecting PIS software failure. We should invest in strengthening the management of the software development team and play a joint role. The direct effect of employees' negative characteristics on PIS software failure is low and not significant, which shows that employees' negative characteristics cannot directly affect PIS software failure, but they can affect PIS software failure through inadequate project management and insufficient software testing. A company can increase its investment in human resources, the company system and the team culture through employee training, knowledge management and system construction.

(3) The existing literature has pointed out that customer requirements are one of the most important factors affecting PIS software failure. However, this thesis finds that the total effect of customer requirement defects on PIS software failure is 0.187, which does not support the results of the existing literature. The results show, though, that customer requirements define the action paths from defects to PIS software failure. They constitute an indirect factor that has an impact on PIS failure through employees' negative characteristics, inadequate project management and insufficient software testing. In this way, the three direct factors are enhanced. The previous semi-structured interviews conducted in Company A show that many employees believe that insufficient software testing is an important factor causing PIS software failure. Through empirical analysis, the study determines that the influence degree of this factor is relatively low. The total effect on PIS software failure is only 0.092, and the significance level is low: it is statistically significant only at the level of 0.05 per cent, which may be related to the item design and participants' understanding of the questionnaire survey.

6.4 Summary

Based on the research model and questionnaire data collection constructed in the previous chapter, this chapter first analyses the data processing model and undertakes a preliminary evaluation of the sample size. Then, the analysis results of the structural equation model are evaluated from the two levels of the measurement model and the structural model, and the model is modified accordingly to form the influencing mechanism model of PIS software failure. Finally, the total effect of PIS software

failure is analysed, and the corresponding explanations and conclusions are presented.

Chapter 7. Conclusion and Prospects

This chapter introduces the main conclusions, including the research findings and contributions, based on which the implications for practice are proposed for managerial reference. The research limitations and further research directions are also provided at the end of this paper.

7.1 Main Conclusions

Due to the fact that the customer complaint rate regarding on-board PISs ranks in the top three among all systems in rail carriages, while the total system value accounts for less than 1% of the total carriage price, it is found from analysing the opening item lists of 10 projects' FAI reports from Company A that most of the customer complaints about PISs are caused by software problems. Therefore, based on the semi-structured interviews, this paper uses Nvivo12 to identify and analyse the factors causing PIS software failure and determines the influencing factors of PIS software failure through induction and condensation; then, it summarizes the interview results and the existing literature research results, builds the influencing factor system of PIS software failure and uses the DEMATEL method to analyse the influencing factors systematically to establish the influence degree, affected degree, cause degree and centrality of each factor. Based on the above analysis, the theoretical model between customer requirement defects, employees' negative characteristics, insufficient software testing, inadequate project management, an uncertain external environment and PIS software failure is constructed. The structural equation model based on partial least square method is applied to analyse the survey data of 341 questionnaires (313 valid questionnaires) received through WeChat, mobile phone and network link within the specified time period.

The research results show that the uncertain external environment, inadequate project management and insufficient software testing are the direct factors affecting the failure of PIS software; customer requirement defects indirectly affect the failure of PIS software by affecting the employees' negative characteristics, inadequate project

management and insufficient software testing; and employees' negative characteristics affect the failure of PIS software through inadequate project management and insufficient software testing. Specifically, the main research findings of this paper are as follows:

(1) This paper qualitatively describes the failure of on-board PIS software through semi-structured interviews with employees of Company A, and uses the Nvivo12 software for qualitative research to find the main factors affecting the failure of PIS software. The interview data are coded to form 484 initial concepts and 34 categories, which are further refined into seven main categories, specifically product characteristics, weak technical support, leadership response behaviour, an imperfect company management system, company employee characteristics, inadequate project management and insufficient customer support. Then, through qualitative research, it is determined that PIS software products have four problems: a complex and quickly updated system, poor stability, poor maintainability and a disorderly downloading mode. The differences of respondents' perception of problems in their positions and work experience are compared and analysed. At the same time, it is found that the root causes of PIS software failure of Company A mainly include weak technical support, inadequate project management, insufficient customer support, imperfect employee characteristics and the management system of the company. Finally, the influencing factors of many PIS software failures are condensed into two aspects – key factors and situational factors – so that managers can focus on these key influencing factors and lay the foundation for a follow-up study of the influence mechanism.

(2) Through the reading of the literature and sorting of data on software project development risk management and risk factor identification, combined with the potential risk analysis in the project implementation cycle and the investigation of project experts, this paper combs the failure of PIS software in seven dimensions: customer requirements, technical factors, project management, the company system, employee characteristics, the leadership style and the external environment. Combined with the questionnaire survey results from Company A, an influencing factor system of

PIS software failure is constructed. Then, the expert consultation method, questionnaire survey method and DEMATEL method are used to evaluate various factors regarding the influence degree, affected degree, centrality and cause degree respectively. The cause degrees of customer requirements, the company system, employee characteristics, the leadership style and the external environment are all greater than zero, and these are the cause factors; meanwhile, the cause degrees of technical factors, project management and product characteristics are less than zero, and these are the result factors. The centrality of project management is 28.109, which is the highest of all the dimensions, indicating that it is very important to software development. The influence degree and affected degree are the highest. This dimension is in a very important position and is a phased result factor. Its behaviour is restricted by other dimensions to a certain extent. The cause degree and influence degree of customer requirements are positive, and the centrality is large, indicating that the requirements of customers, as the demanders of PIS software, have a central impact on PIS software failure, but this item's impact role cannot ignore the important position of project management. The external environment has the highest cause degree, indicating that it is the direct cause of PIS software failure. Therefore, customer requirements and the external environment are the two most important dimensions that affect the function failure and poor sales of PIS software.

(3) Through the semi-structured interviews conducted with respondents from Company A, the extensive literature reviews and the quantitative analysis of influencing factors combined with the DEMATEL method, it is found that inadequate project management, employees' negative characteristics and insufficient software testing are the direct factors leading to PIS software development failures. Customer requirement defects and an uncertain external environment will affect inadequate project management, employees' negative characteristics and insufficient software testing and then further play a key role in affecting software failure. On this basis, the influence mechanism model of PIS software failure is constructed. Further, taking the uncertain external environment as the starting point, the research model containing customer requirement

defects, employees' negative characteristics, inadequate project management and insufficient software testing is constructed. It provides a theoretical basis for exploring the failure influence mechanism of PIS software from a new perspective.

(4) This paper finds two kinds of influence paths leading to PIS software failure. The first is the direct influence path. An uncertain external environment, inadequate project management and insufficient software testing are the direct influence factors that directly cause PIS software failure. The second is the indirect influence path. Customer requirement defects and employees' negative characteristics are the indirect influence factors of PIS software failure. The former indirectly affects the failure of PIS software by influencing the employees' negative characteristics, inadequate project management and insufficient software testing, while the latter affects the failure of PIS software through inadequate project management and insufficient software testing. The existing literature has demonstrated that most software errors (50%–71%) are normally related to the customer requirement defects (Kumar and Kumar, 2011, Muhammad Naeem Ahmed et al., 2013, Ali, 2015). The research results of this paper show that the total effect of customer requirement defects on PIS software failure is only 0.187, which demonstrates the particularity of the rail transit industry in China, a rapidly developing economy. At the same time, it makes clear the action path from customer requirement defects to PIS software failure. It is not a direct effect but affects PIS software failure through employees' negative characteristics, inadequate project management and insufficient software testing, further strengthening the role of the direct influencing factors.

(5) The integrated perspective based on the questionnaire survey, interviews and literature review can reveal the impact mechanism of PIS software failure more comprehensively. Emerging from the process of coding the data, the research identifies the mechanisms that influence PIS software failure; combining these results with insights from risk management theory enables a more complete explanation of software failure. PIS software failure depends on many internal and external factors; and these internal and external factors also affect each other. Therefore, the research presented in

this paper can reveal the influence mechanism and failure path of PIS software failure more comprehensively.

7.2 Main Contributions

The main contribution of the research lies in identifying the mechanism and action pathways that influence software failure. This was achieved by drawing on and proposing, as a second contribution, a research model/framework based on the existing literature review and semi-structured interviews. In addition, the process of the scale development, the design and selection of variables in the research also provide a good reference for future studies. Finally, the mixed-method research approach that this research adopts (and which related studies typically do not) has enhanced the reliability and validity of the study's findings, potentially providing some inspiration for future research. The contributions are explained in detail from following three aspects:

(1) The research has identified the influencing factors of PIS software failure, and clarified the causal relationship between the influencing factors systematically through a research method that combines qualitative and quantitative analysis.

Based on three-stage coding process, the influencing factors of PIS software failure are summarized and condensed through semi-structured interviews that are refined into seven main categories. These categories, specifically, are product characteristics, weak technical support, leadership response behaviour, an imperfect company management system, employee characteristics, inadequate project management and insufficient customer support. Based on the semi-structured interviews and literature review, this paper constructs an influencing factor model of PIS software failure. The DEMATEL method is applied to analyse the influencing factors of the constructed PIS software failure model quantitatively to determine the influence degree, affected degree, centrality and cause degree, clarifying that customer requirements, company system, employee characteristics, the leadership style and the external environment are the causal factors. These factors determine whether the PIS software functions can be realized, while technical factors, project management and product characteristics are the result factors, which then affect the failure of PIS software.

(2) Based on this causal relationship, this study has put forward a research model/framework and identified the mechanism and action pathways that impact PIS software failure for a more in-depth and comprehensive analysis. Although there have been a large number of papers, empirical analyses and case studies that examine the relationship between customer demand and software failure (Loconsole, 2004, Kumar and Kumar, 2011, Ali, 2015, Muhammad Naeem Ahmed et al., 2013), the fundamental mechanism of how various influencing factors affect software failure is still a major theoretical and practical problem that has not been solved in the field of information systems. By putting these influences and pathways forward as key factors that explain software failure, this research helps to fill a gap in the existing literature. According to the cause degree and centrality of factors, the influencing factors are divided into cause and result factors, and the failure of PIS software is an important premise. This paper constructs a research model involving an uncertain external environment, customer requirement defects, employees' negative characteristics, insufficient software testing, inadequate project management and PIS software failure, providing a theoretical basis for exploring the mechanism of PIS software failure from a new perspective. Through a questionnaire survey and structural equation model analysis, the influence mechanism of PIS software failure is investigated, the action coefficients are obtained. SmartPLS processing results and hypothesis test conclusions reveal the action path and influence effect of various factors on PIS software failure.

(3) Some efforts have been done on the development of the scale including the design and selection of the variables in the research, which provides a reference for subsequent research.

A scale of high validity and reliability is the most important factor in the process of empirical research. The scale development of this study referred to Hinkin's six-stage process idea and the questionnaire survey was distributed twice. First a small-scale questionnaire pilot test was conducted and then the initial measurement items corresponding to different variables/constructs were adjusted and purified. In this way, a relative perfect scale and questionnaire were formed for the second survey in a

large-scale distribution. The result of empirical analysis proves the effectiveness and feasibility of such an improvement method of scale development and measurement items/variables.

Besides, the mixed-method research approach in this study enhances the reliability and validity of the study's findings, and also gives some inspiration to the research idea, which is the limited use of the approach for IS software failure studies in rolling stocks industry.

7.3 Managerial Implications and Empirical Application

The identification of influencing factors and the study of their action pathways provides solutions that software enterprises, especially those within rail transit industry, can draw on to reduce the incidents of software failure and effectively improve the competitiveness of enterprises. The research result has comprehensively managerial implications which promote software project success rate as well as empirical application meaning which demonstrates how to stimulate a business model innovation of an enterprise.

7.3.1 Managerial Implications for Software Success Promotion

The research findings of this paper reveal the influencing factors of PIS software failure and the action mechanism of these factors on software failure. Therefore, the practical application value of the research mainly includes the following three aspects:

(1) The impact of customer requirement defects on PIS software failure is not a direct impact but an impact on other aspects of the enterprise that, in turn, cause PIS software failure indirectly. Therefore, senior managers, including enterprise CIOs, need to be aware of the following points in the process of PIS software development: customer requirements are of strategic significance, but this strategic value is not directly reflected. It is realized indirectly through enterprise management, employees' ability, software testing and so on. Therefore, in the process of PIS software development and application, enterprises should actively seek opportunities to realize customer requirements, for example determining how to reduce the cost of products or services, how to differentiate their products or services through technology innovation and so on.

They should also actively seek customers' support for enterprise products, and integrate customer requirements with the selection and employment, training and development, promotion of enterprise developers and so on so that the customer demand value can be transformed into an important part of the enterprises' core competence to enhance the competitiveness and sustainability.

(2) Inadequate project management plays a more important role in the development and application of PIS software than other factors, whether in a dynamic or in a stable environment. Therefore, in the process of PIS software application, senior managers, information system managers and business managers of enterprises should shift the focus from software development or implementation to the construction, improvement and optimization of the project management system to help enterprises build up the ability to respond quickly to internal and external changes when they happen. At the same time, senior managers, information system managers and business managers of enterprises need to make it clear that, in the process of cultivating different project management capabilities, there are different resource requirements. To cultivate and improve a certain project management capability better and faster, it is necessary to obtain capabilities and technical means from employees who have a great impact on such a capability.

(3) PIS software failure is an enterprise-level concept that includes not only the technical application ability of the information technology department and the information technology project team but also the product application ability of senior managers and business departments. Therefore, the effect of PIS software application depends not only on the information technology department and the information technology project team but also largely on the product application ability of the senior managers and business departments. Hence, in the process of PIS software development and application, enterprises' departments need to cooperate with each other to improve their application ability to help enterprises create and maintain a sustainable competitive advantage.According to the research results and conclusions stated above, managerial implications to avoid PIS software failure can be put forward regarding the following

two aspects:

(1) Pay attention to the analysis of internal and external factors of PIS software project development to ensure the success of PIS software project and avoid a high failure rate. On the one hand, the internal factors of projects should be controlled by focusing on the two factors of project management and enterprise employees. In particular, project managers should focus on cultivating their “soft ability” and need to have strong adaptability when dealing with the uncertainty of the external environment. On the other hand, the external factors of the project are also very important, mainly consisting of the external environment and customer requirements. For example, in terms of customers, managers should always be aware of the importance of regular and clear communication with customers and the provision of timely feedback information and pay attention to benign interaction.

(2) Be fully familiar with and use the theoretical methods of risk management and project management maturity to lower the failure rate of software development projects. Through the project risk management process, it is possible to utilize the Monte Carlo risk simulation model and other methods rationally to make the development process of PIS software more scientific and accurate. It is also possible to evaluate and improve the software development process and software development capability through the maturity model of project management and to increase the success rate of software projects from the perspective of project evaluation and strategic planning. Paying regular attention to, optimizing and improving project risk management technologies and applying project management maturity to the enterprise PIS software project management can reduce the failure rate of PIS software.

7.3.2 Empirical Application of the Platform-Based Model

Most of the existing studies had the common idea that incorrect or poor requirement management plays the most important role in software failure. However, when focusing on the specific industrial area of rolling stocks, the influencing factor mechanism shows new findings: an uncertain external environment and inadequate project management are the two key factors. The importance of customer requirement defects is lowered to a

considerable degree. In this way, the business model innovation from the traditional customer-tailored model to the supplier-leading model has become a new empirical tendency under the external new technology development of digital technology as well as big data. (1) Traditional customer-tailored business model: Different customers, especially carriage builders and end-users, have different requirements. The traditional project management method (Figure 7-1 below) enables each project manager to form project team composed of different functions, including R&D, procurement, finance, quality, production and logistics strictly, with respect to a single customer's special requirements. One project manager with a team normally takes care of one project and sometimes also take care of two or even more projects, which would produce a problem of priority when resources crash. Accordingly, for each project, the customers' requirements are different and the products, including hardware and software, are customized. Limited by the project development period and budget, knowledge management and experience from previous projects can be difficult to pass on to new projects. All these put projects at risk, especially the risk of software failure.

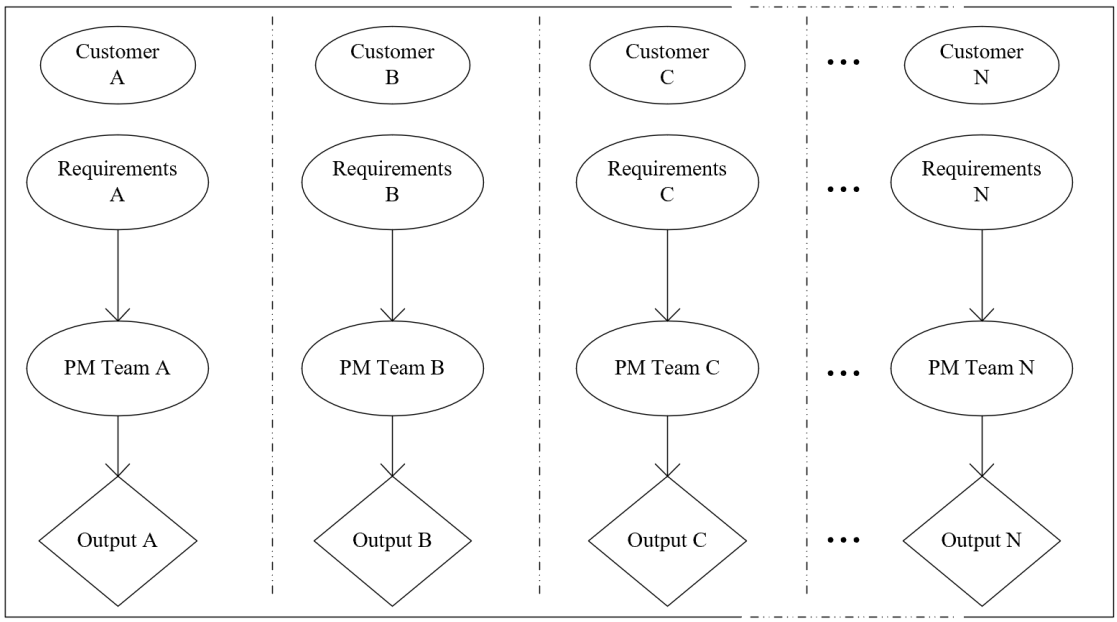


Figure 7-1 Traditional Customer-Tailored Business Model

(2) The new technology of digitalization and big data enable, on the one hand, the platform model to combine all the stakeholders and collect all their ideas and demands and, on the other hand, the product engineers to design the software as modules and the

project manager to develop terminals and system through different configurations by combining models and specific characteristics (Figure 7-2 -> Figure 7-3 -> Figure 7-4).

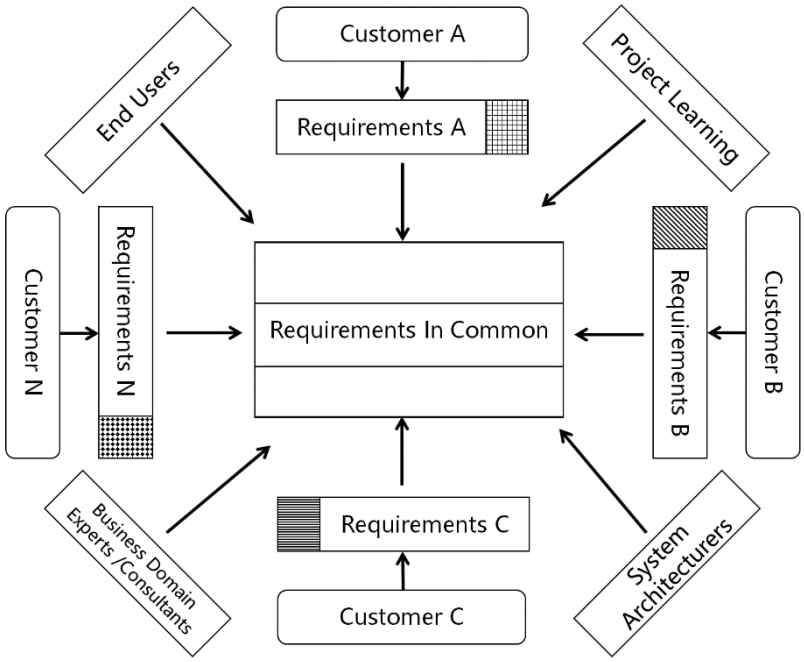


Figure 7-2 Requirement Input Diversity of the Supplier-Leading Business Model

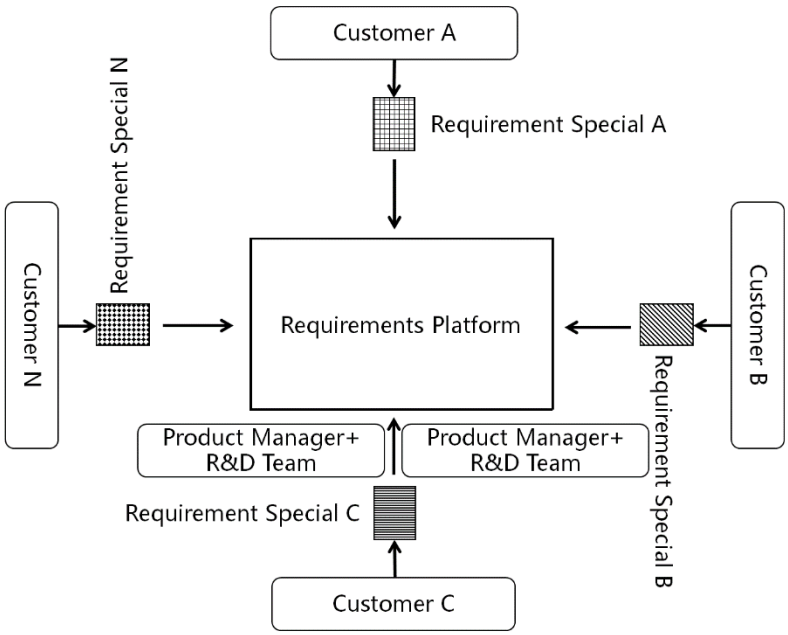


Figure 7-3 Requirement Platform of the Supplier-Leading Business Model

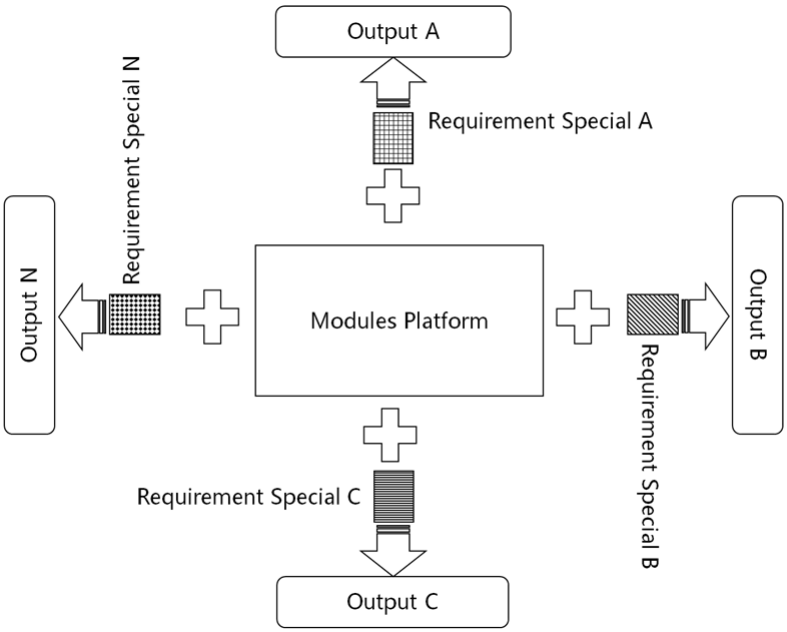


Figure 7-4 Output Platform of the Supplier-Leading Business Model

The business model innovation from the customer-tailored to the supplier-leading model emphasizes the active role of PIS suppliers and shows that it is a scientific option since it is well known that suppliers are much more professional than customers in PIS development technology. The supplier could use big data from all the stakeholders of the system with the support of digital technology to gather information on the external environment, including industry standards, government policies and so on, to reduce the environmental uncertainty risk. The new business model also upgrades the project management to raise the software success rate. The traditional linear top-down project management structure (Figure 7-1) is replaced with a platform module plus customization for different customers (Figure 7-4), the latter of which has the advantage of maximizing the returns with minimal resources. The R&D team can focus its efforts on the “customization” and “specialization”.

7.4 Research Limitations and Future Research Directions

This paper uses a new method, the “structural equation modelling method”, to analyse the mechanisms that influence PIS software failure: this method has some innovation compared with traditional factor analysis or regression analysis. As exploratory research, this paper constructs and verifies the research framework and empirical model of PIS

software failure caused by an uncertain external environment, customer requirement defects, employees' negative characteristics, inadequate project management and insufficient software testing and obtains meaningful conclusions. However, there are still some deficiencies in the construction of the measurement model, data collection and research design, which need to be solved in further research. The main research limitations of this paper include the following:

(1) Since this paper combs the influencing factors of PIS software failure based on the three-stage coding process, the case company selected for the study is the author's work unit, and the study has exploratory characteristics. Research is undertaken on software failure, project failure and risk factors at home and abroad, but the theoretical analysis and empirical testing of its influencing mechanism are still very limited. Scholars have not reached a consensus on the measurement of software failure's influencing factors, customer needs and software failure. Although this thesis tries to develop a measurement model of the above latent variables with high reliability and validity by means of a literature review and expert interviews, because the relevant investigation is limited to the individual interview method, does not carry out large-scale sample investigation and is based on the analysis of small-scale pilot test data, innovative empirical research was successfully carried out using 313 actual survey data. However, considerable effort is still required to build a widely accepted scale.

(2) On the extraction of the influencing factors causing PIS software failure, although this thesis reviews many relevant studies and materials to ensure validity, due to the diversity of measurement indicators, there may be imperfections when selecting which indicators to use to measure the dependent variable factors, and some measurement indicators need to be discussed further. For the measurement of PIS software failure, this paper uses indicators such as software product characteristics and users' satisfaction with software products. Although we add time factors and comparison with competitors in the design process of measurement items, and reduce measurement errors through multiple measurement items, the use of many objective data could make the measurement of PIS software failure more accurate.

(3) This paper uses the method of key information providers to collect data. Although this method has its own advantages, deficiencies exist in comprehensiveness of data collection using this method. The research data in this paper mainly represent the respondents' perception of the application status of PIS software. Although they are directly related to the development, management and application of PIS software, due to the limitations of data collection methods, it is difficult to understand fully the application status of PIS software in enterprises.

(4) This paper has a cross-sectional instead of a longitudinal study. Therefore, the sample data studied represent the basic situation of enterprise PIS software application at a certain point in time, which makes it difficult to clarify the causal relationship between customer requirement, project management, employee characteristics, software testing and PIS software failure. Although through normative theoretical deduction and typical case analysis, the relationships between the above variables are clarified, and they are further verified through data analysis and hypothesis testing, this approach does not clarify the causal relationship between the influencing factors of PIS software failure. Therefore, future research can collect and analyse longitudinal data to reflect fully the time lag and long-term impact on enterprise PIS software failure and clarify the causal relationship between them.

Based on the limitations of the existing research and the needs of practical application, the study proposes the following further research directions:

(1) The measurement model of PIS software failure and an uncertain external environment and the further verification of the model of PIS software failure influence the mechanism from the perspective of integration. The development and application of PIS software are affected by many internal and external factors. In addition to the variables involved in the research model in this paper, future research should include the corporate culture, values and other factors to upgrade the existing research model. In this way, more survey data from different cultural backgrounds, different regions and so on can be collected and fitted to refine the measurement model of PIS software failure further to verify the effectiveness of the impact mechanism model of PIS software

failure.

(2) The influence mechanism of PIS software failure is studied based on dynamic environment. Although this paper considers the impact of environmental uncertainty on PIS software failure and finds its action path and importance degree, it does not determine the different impact mechanisms of PIS software failure in both a dynamic and a stable environment, nor does it study the relationship between dynamic adjustment of environmental factors to other factors in relation to PIS software failure. The current market environment has entered the dynamic stage, and the uncertainty of the external environment has become the consensus of the academic community. Therefore, based on the uncertainty of the external environment, its role in the failure process of PIS software can be investigated. Exploring the failure of PIS software in a dynamic environment is a research direction with important theoretical value and practical significance.

(3) The cultivation of enterprises' core technology application ability and employee development ability, which include software development capability and software testing capability, constitute the key factor determining the application effect of PIS software. It plays a very important role in the process of PIS software development and application. Especially in the dynamic environment, the core application capability of enterprise technology has strong flexibility and it can continuously enhance the competitiveness of enterprises by continuously developing and updating a PIS that supports the competitive strategy and responds to the customer demands. Therefore, determining how to develop and cultivate the core application ability of enterprise technology is a research topic of important practical significance. Although this paper has made a preliminary study of the relevant factors, there are still many fields worthy of in-depth research, such as the cultivation mode and cultivation strategy of enterprises' core technology application ability based on learning theory.

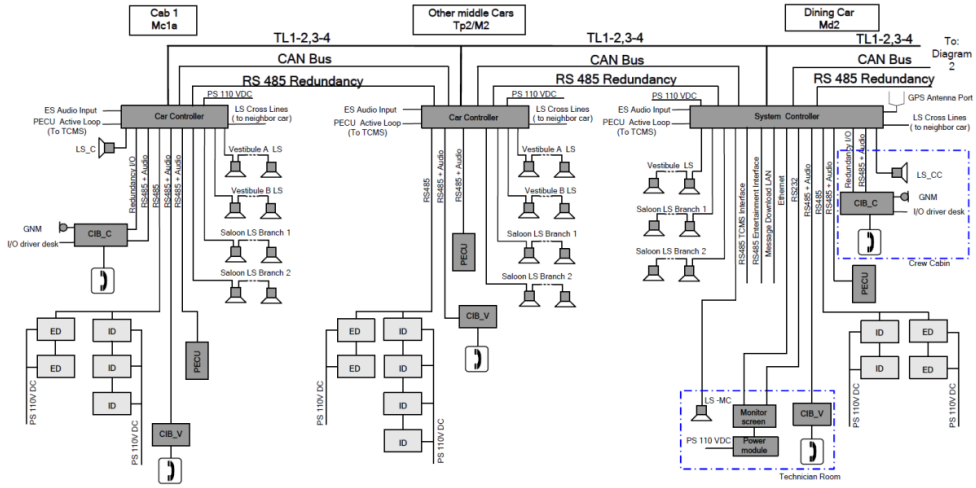
7.5 Summary

The author draws the conclusions in the last chapter, listing the research findings and the contributions from both the theoretical and the practical perspective, based on which

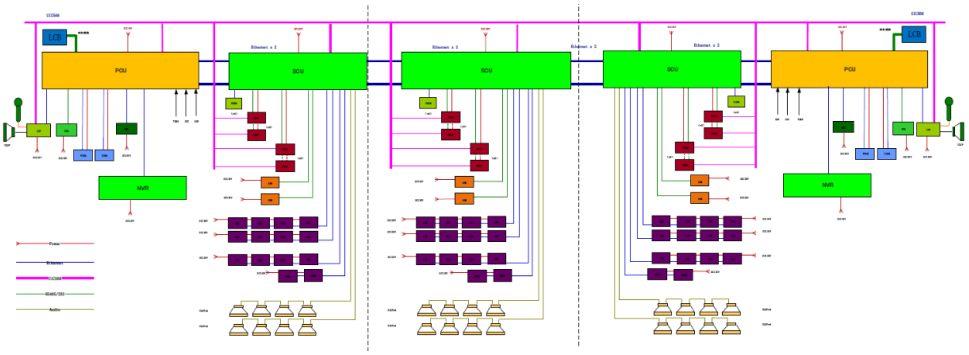
a business model innovation with platform-based managerial application is suggested. The research limitations and further research directions are also suggested at the end of this paper.

Appendix

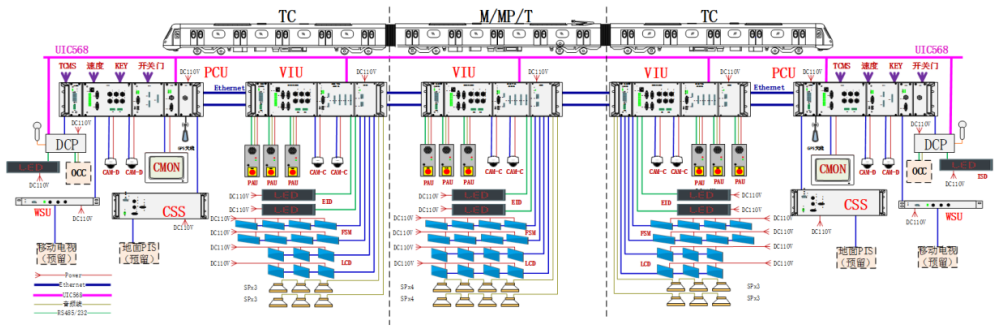
Appendix 1-1 Traditional Architecture of On-Board PIS



Appendix 1-2 Upgraded Architecture of On-Board PIS



Appendix 1-3 Full Network Architecture of On-Board PIS



Appendix 1-4 Failures of On-Board PIS Software and Their Impact on Traffic Operation

No.	Software Failure Phenomenon	Subsystem	Impact on the Passengers	Impact on Operation Personnel	Impact on Operation Company	Impact on Cars	the Customers' Complaints
1	The station announcement function cannot be realized.	Public Address System	Passengers are unable to obtain effective station information, leading them to get on and off at wrong stations, which affects the passengers' travelling efficiency and complaints, resulting in the passengers' wrong boarding and alighting, affecting travel efficiency and causing complaints.	It is necessary to appease the passengers in time and handle their complaints, increasing the workload.	It may need to compensate the passengers, affect the goodwill of the operator, bear major financial losses and be dissatisfied with the PA system supplier.	Suspended	Yes
2	Passenger emergency intercom function failure.	Public Address System	Unable to contact the driver in an emergency, missing the best time to stop loss, resulting in passenger property loss or death.			Suspended	No
3	The PIS display shows that Chinese and English are out of sync, garbled codes and * * * donkey * * * similar insulting words are displayed.	Information System	Passengers are unable to obtain effective station information, leading them to get on and off at wrong stations, which affects the passengers' travelling efficiency and complaints.			Suspended	Yes
4	The PIS display shows that the train speed is 2,000 km per hour, and the speed display is not updated.	Information System	Passengers are unable to obtain effective station information, leading them to get on and off at wrong stations and affecting the ride service experience, and even miss the trip, causing complaints.			Suspended	Yes
5	Car number error on external display during train coupling.	Information System	Passengers are unable to obtain effective station information, leading them to get on and off at wrong stations affecting the ride service experience, and even miss the trip, causing complaints.			Suspended	Yes
6	Part cameras fail.	Surveillance System	The occurrence of an emergency cannot be traced, and the accidental damage caused to the passenger's personal property cannot be recovered, resulting in the passenger's dissatisfaction with the operator.	Unable to play back and query image/voice information locally.		To repair failed components	Yes
7	Loss of camera monitoring data.	Surveillance System	The occurrence of an emergency cannot be traced, and the accidental damage caused to the passenger's personal property cannot be recovered, resulting in the passenger's dissatisfaction with the operator.			Suspended	Yes
8	The functions of the driver's main operation interfaces, such as touch screen and DACU, cannot be realized or are realized incorrectly.	PIS	The normal experience of the passengers cannot be realized, or the passengers are misled.	It is necessary to appease the passengers in time and handle their complaints, increasing the workload.		Suspended	Yes
9	The communication error with TCMS interface leads to the error of the whole PIS.	PIS	Passengers are unable to obtain effective station information, leading them to get on and off at wrong stations, which affects the passengers' travelling efficiency and complaints.			Suspended	Yes

Appendix 3-1 Interview Guide

This part contains an interview outline for members of Company A's software development projects. The interview methods include telephone interview, e-mail inquiry and face-to-face interview.

1) Do you usually work with software development?

2) If so, what is the biggest challenge of software development in your opinion? Please give examples as much as possible.

3) The data show that the proportion of software problems encountered in the development of mechanical and electrical products is far greater than that of hardware. Does your work experience support this judgement?

4) The literature tells us that more than 70% of software problems are related to poor requirement management. What do you think is the potential cause of software development failure? Please list the causes of software problems as far as possible.

5) Summarize your analysis of several reasons. If you are asked to rank these reasons, which three reasons do you think are the most important?

6) What do you think should be done to adress these causes of software development failure? Please make a presentation from the company level, team level and member level.

7) How do you usually evaluate the results of software development? What are the important indicators?

8) What other ideas or suggestions do you have for software development?

Appendix 4-1 Extracts from the Interview Materials

1. Factor Identification of PIS Software Failure

1) The software product system is complex and quickly updated

Extracts from some interview materials:

A2: Some problems that can't be solved may be difficult to solve. We don't want to try again and change it into another one. As a result, unexpectedly it was also very difficult.

B3: Because our PIS system is a large system, unlike developing a small device alone without communication with others. The on-board PIS is a large system, which needs to communicate with ATC, TCMs and trackside, including all kinds of equipment.

B7: If there are too many things interacting with PIS.

2) Poor stability of software products

Extracts from some interview materials:

A4: The engineer goes to the site to change. After the change, many problems cannot be found and are unpredictable. After running, it may bring some new series of problems. In this way, it feels that your system is very unstable and constantly reports problems.

B5: If it's like us, it basically depends on the feeling of the software developers themselves. Many times, it depends on the feeling; that is, I feel that there should be no problem when I write like this. Then I run it on the platform without any problem. It may not be the case when I go to the site. The on-site environment is much more complex than our company's platform.

3) Poor maintainability of software products

Extracts from some interview materials:

A3: If you modify or add new functions to other people's code, you will spend more time getting familiar with the code than writing new code yourself. Time is often not allowed, because a new code needs to be retested, and it is best to change it in the original code. This is the fastest, but there will be other problems, because some places you don't understand and it is easy to go wrong.

A5: The software formats are basically different. It is rare to have the same board. There is a difference between the same hardware. Some people like to pack it together, and some people like to download it one by one. The more complex it is, the easier it will be to make mistakes after sales. If

you make mistakes, it will be easy to report faults and problems. Sometimes it's good to brush it again.

B3: For example, I've been working on the old line 12 and line 13 some time ago. I don't even have a program. How can I do it?

B3: The service cycle is too long. It takes 15 years for the carriage. A chip can't be supplied continuously for 15 years.

B5: General electronic products don't have such a long service, because they usually are used for 1–2 years. If they break down, they will be replaced with new ones. Industries like ours are special. 10 years later, they still let you change the software to meet their functional needs.

4) Software product download mode is disorderly

Extracts from some interview materials:

A5: The download methods of software are not unified, and everyone's habits are different ... Some people like to pack together, while others like to download one by one. The more complex it is, the easier it will be to make mistakes after sales. And it will be easy to get report faults and problems.

B4: All our software is downloaded in different ways. There are legacy problems ... A large number of software were chaotic, and some were lost, just before PDM was applied for documentation.

2. Cause analysis of PIS software failure

1) Weak technical support

Extracts from some interview materials:

A2: There are two reasons. One is that the production of the products was stopped. Maybe it was in the middle and late stages when we selected it. Considering that it is stable and reliable, the service life may be relatively short; there are many unsolvable problems in the previous one, which can only be changed in the next project.

A3: What I want to say is related to time, including the development cycle and test cycle, which are limited not only to in our company but also to in the carriage. Especially in the later stage, when our platform is complete, all the functions we should do have been completed at home. There is no

problem. When we need to get on the bus for debugging, it is difficult to control the time, and the efficiency on site is very low.

A4: If you modify or add new functions to other people's code, you will spend more time getting familiar with the code than writing new code yourself.

B3: After having a relatively stable product, based on this hardware, the hardware will always be like this and will not move. It's just that for different needs of each project. It's OK to change the software a little, unlike us changing every day. Today, I finally developed some new things on this device, which are easy to use, but they have been changed in the next project.

B5: Many aspects of software can be tested through some software. Some professional tests can detect many problems. It is not like us basically depending on the software developers themselves. Many times, it depends on the feeling; that is, I feel that there should be no problem when I write like this, and then I run it on the platform without any problem. It may not be the case when I go to the site. The on-site environment is much more complex than our company's platform. After clarifying the requirements, you should have a principled design for the whole framework of your software. When I am developing software, it is necessary to divide my software into several parts and do some structured design.

B8: It may take 1 or even 2 years to build a module, but once this module is built, it will be used for 8 or 10 years. It's not that you do one project today and the next tomorrow.

C3: In fact, the best state is to modularize our core code without changing it. After the new project comes out, you can directly use the module. This has the lowest probability of problems, and the code accumulated in 10 years and 8 years is unlikely to have problems. What has changed is some logic between various projects.

2) Inadequate project management

Extracts from some interview materials:

A1: We had no experience before. Now when we decide that it is a Chinese character, we cut off two bytes in advance to avoid splicing errors and making jokes. The customer said you were fast or slow.

A4: I have never participated in the inter-departmental review ... What's more, we may not have

fully identified some customers' previous needs, and some requirements may not have been fully identified.

B2: The current software control is still relatively poor. There may have been a release process in the past, which is to go through a formal release procedure after verification on the platform, right? To send the program ... This test condition, right? After adding some function, you can only go to the site to test them. Second, there are a lot of such costs that the customers won't pay, will they? It causes cost and normally two engineers are required for the site work.

B3: We must have a simple understanding of our whole PIS system and some PIS-related systems. In this way, your program, including the program framework you built, will be considered more, and the later program expansion will be better

B4: Sometimes the customer pushes very hard. For example, the Hanyang Tram project was a big one. The customer urged him to do it quickly. He wanted to get on the bus the next day when he went on a business trip, and then nobody took care of it for half a year ... At least the project manager made clear what he needed to do first. To do a project successfully requires more reviews. If there is no review, it is basically in the state of each person getting on with his own job.

3) Insufficient customer support

Extracts from some interview materials:

A1: The demand is not clear. For common things, such as the display scrolling, the customer requires a good visual effect, but there is no quantitative value. I don't know how many times a second. A value may be adjusted according to my experience. The customer says you are fast or slow. When demands cannot be quantified, it is a great challenge.

B2: For example, the display interface of Changzhou Line 1 dynamic map and the display interface of the TV set are different because everyone's aesthetics are different. Our company produced three sets of schemes at that time, and then the customer has different leaders. Maybe the first leader finished reading the scheme, and he said it was OK. And then it is read by a higher-level leader. He said the scheme was too ugly and had to be changed again.

B3: We must have a simple understanding of our whole PIS system and some PIS-related systems. In this way, your program, including your program framework, will be considered more,

and the later program expansion will be better.

C1: Sometimes the customer's business changes so fast that the developed products or functions fail in a short time ... There are too many factors affecting the verification on site because the environment is more changeable on site.

D2: Customers often complain that they don't want to see all kinds of information irrelevant to their business in an interface of the system, because it will increase their visual fatigue, but customers of other business departments must keep this information.

4) Company employee characteristics

Extracts from some interview materials:

A1: And to modify other people's codes. Some engineers leave their jobs. It may take a long time to get familiar with their codes.

A4: Engineers write the software with a strong personality; that is, the software written by each engineer is not necessarily the same ... Each engineer works in his own way, and there is no core. You can coordinate with everyone to see where the problem is or whose problem I judge.

B7: In compiling and decoding software, it is impossible to avoid mistakes when a person writes a function and tens of thousands of lines of code.

B8: Our company is in Changzhou. First, it has something to do with the staff in Changzhou because our current staff is not the top people in the industry or the very well-known universities, doctor's or master's students with high academic qualifications. Most of them are undergraduates. Of course, some of them graduated from 211 or 985 universities (first-class universities in China), but there is still a lack of personnel ability.

5) The company's management system is not perfect

Extracts from some interview materials:

B1: Nowadays, software lacks knowledge management, accumulation of experience or inheritance of experience, which depends on the individuals. A good software team should have its own knowledge base, which is internal ... In this area of software, I think initiative and enthusiasm are the most important. Enthusiasm and initiative, in fact, need to be stimulated. Our reward policy for this area is quite backward. The person who writes software does not have a sense of engaging

activity, and he does not have a sense of achievement after a project is successful.

B8: Product planning, right? It may take 1 or even 2 years to build a module, but once this module is built, it will take 8 or 10 years. It's not that you do one project today and the next one tomorrow. Even God cannot finish it. This module is the software modularization, and the package is even bound with the hardware. This platform is relatively like our set-top box. Now it is relatively stable after a few years.

C1: If it is found at the beginning of the project that our technology is unable to do this, and the software can't do it, should we consider outsourcing?

C3: It's worth setting up this framework or letting us programmers train and participate in software architecture training.

3. Role of company leaders

Extracts from some interview materials:

B1: Some functions of current software development often are that he will do what he is taught by the project manager.

B3: Sometimes when we review the interface protocol, Xue Union will put forward some opinions, and we will add his opinions to our basis.

B4: I feel that, under such circumstances, nothing makes sense. Anyway, our leaders are quite reasonable. I told them that basically, if he felt reasonable, he would agree. If he said no, he would also explain the reasons. It's OK.

B7: I didn't urge the project manager at that time, so I had to brace myself to do it.

C2: We can find them if we don't know whether they are in the technical layer or which function. They can all answer. They will tell me everything they know. This is no problem.

Appendix 5-1 Questionnaire A for the Pilot Test

Statistical Table of the Importance Degree of the Risk Evaluation Index for On-Board Passenger Information System Software Development

The following indicators are from project management professional (PMP), scientific research papers and practical projects of some Fortune 500 enterprises.

1. Demand risk*

1.1 Unclear definition of requirements*

1	2	3	4	5
Very unimportant				Very important

1.2 Insufficient demand research*

1	2	3	4	5
Very unimportant				Very important

1.3 Changing demand*

1	2	3	4	5
Very unimportant				Very important

1.4 Lack of effective requirement change management*

1	2	3	4	5
Very unimportant				Very important

1.5 Failure to reach an agreement when disputing needs*

1	2	3	4	5
Very unimportant				Very important

1.6 Demand gilding: the real needs of users collected are raised innocently by R&D personnel*

1	2	3	4	5
Very unimportant				Very important

2. Technical risk

2.1 Software design defects*

1 2 3 4 5

Very unimportant Very important

2.2 Adopt an advanced but immature technical scheme*

1 2 3 4 5

Very unimportant Very important

2.3 The development facilities are not in place or the development environment is in disorder*

1 2 3 4 5

Very unimportant Very important

2.4 The interface with the third-party system is not smooth*

1 2 3 4 5

Very unimportant Very important

2.5 Software iteration management confusion*

1 2 3 4 5

Very unimportant Very important

2.6 The development process is not standardized or scientific*

1 2 3 4 5

Very unimportant Very important

3 Progress risk

3.1 Unreasonable project progress schedule*

1 2 3 4 5

Very unimportant Very important

3.2 Lack of effective monitoring of progress*

1 2 3 4 5

Very unimportant Very important

3.3 Project management and coordination are difficult*

1 2 3 4 5

Very unimportant Very important

3.4 Project resources are not in place*

1 2 3 4 5

Very unimportant Very important

3.5 The quality assurance/control system has not been effectively implemented*

1 2 3 4 5

Very unimportant Very important

3.6 Lack of senior management support*

1 2 3 4 5

Very unimportant Very important

4. Cost risk

4.1 The bid price is far lower than the controlled bidding price*

1 2 3 4 5

Very unimportant Very important

4.2 The construction budget cost exceeds the bidding budget cost*

1 2 3 4 5

Very unimportant Very important

4.3 The financial cost exceeds the budget*

1 2 3 4 5

Very unimportant Very important

4.4 Pay no attention to the management and control of quality cost and construction period cost*

1 2 3 4 5

Very unimportant Very important

5. Personnel risk

5.1 Lack of project management experience

1 2 3 4 5

Very unimportant Very important

5.2 Lack of experience of project technicians

1 2 3 4 5

Very unimportant Very important

5.3 Change of key project personnel*

1 2 3 4 5

Very unimportant Very important

5.4 Demand deviation caused by insufficient user cooperation*

1 2 3 4 5

Very unimportant Very important

5.5 The project personnel are not in place in time or the employee mobility is high*

1 2 3 4 5

Very unimportant Very important

5.6 Friction within the development team*

1 2 3 4 5

Very unimportant Very important

6 External risk

6.1 The contract is unjust or compliant*

1 2 3 4 5

Very unimportant Very important

6.2 Governments or other agencies have imposed restrictions on the project development*

1 2 3 4 5

Very unimportant Very important

6.3 New industry standards or laws and regulations have been issued recently*

1 2 3 4 5

Very unimportant Very important

6.4 Unpredictable market turbulence*

1 2 3 4 5

Very unimportant Very important

7. Other factors than you can think of (you can enter more than one)*

Appendix 5-2 Questionnaire B for the Large-Scale Survey

Dear Madam/Sir,

Hello! First, thank you very much for taking the time out of your busy schedule to participate in this questionnaire survey. The purpose of this research is to investigate the main factors affecting the failure of on-board PIS software and to provide a scientific basis for improving the performance of PIS software development. Your answer is very important for the successful completion of this study! There is no right or wrong answer. Please fill in the questionnaire according to your actual work experience and reflect your true views. This research is anonymous. All research data will only be used for academic research and will be strictly confidential. The research results will only show comprehensive data and will not involve any personal information. If necessary, we can provide the research results to your company for reference in operation and management. We thank you again for your support and participation in this research and wish you a nice day!

Completion Instructions:

1. It is required that the completing personnel are engineering and technical personnel related to on-board PIS software development, project managers or personnel participating in/understanding on-board PIS projects;

2. The investigation is aimed at the rail transit on-board PIS software development project, and the software system is customized according to the requirements of specific customers;

3. To study the effectiveness, regardless of whether the project performance is good or bad, please truthfully provide information about a recently completed or ongoing on-board PIS software project (it is better not to select a project with particularly good performance).

Part I Project Background and Basic Personal Information

1. How long have you been engaged in the on-board PIS software business?

A. Less than 1 year B. 1–5 years C. 5–10 years D. More than 10 years

2. Please provide the duration of any on-board PIS project in which you have participated or are participating.

A. Less than 1 year B. 1–3 years C. 3–5 years D. More than 5 years

3. Number of on-board PIS project team members:

A. Fewer than 5 people B. 5–10 people C. 10–15 people D. More than 15 people

4. Your position in the project:

A. Project (implementation) director

B. Project manager

C. Developer

D. Quality control personnel

E. Others_____

The main factors affecting the failure of on-board PIS software are investigated later, with a total of 36 items. Each item adopts the 5-point scoring method: 1 = very non-compliant, 5 = very compliant.

Part II Investigation of the Key Influencing Factors of On-Board PIS Software Failure

No.	Description	Very noncompliant	Noncompliant	Uncertain	Compliant	Very compliant
1	Requirements are not clearly defined	1	2	3	4	5
2	Unclear requirements	1	2	3	4	5
3	Changing requirements	1	2	3	4	5
4	Inconsistent opinions of customer leaders	1	2	3	4	5
5	Insufficient testing	1	2	3	4	5

6	Site test environment not allowed	1	2	3	4	5
7	Advanced but immature technical proposal adopted	1	2	3	4	5
8	Development process is not standardized or scientific enough	1	2	3	4	5
9	Hardware problem resulting in software function change	1	2	3	4	5
10	Product quality control not in place	1	2	3	4	5
11	Cost is out of control	1	2	3	4	5
12	Difficult project management and coordination	1	2	3	4	5
13	Insufficient project management experience	1	2	3	4	5
14	Lack of effective requirement change management	1	2	3	4	5
15	Inadequate prior risk review	1	2	3	4	5
16	No good planning for products	1	2	3	4	5
17	Poor knowledge management	1	2	3	4	5
18	Lack of incentive mechanism for employees	1	2	3	4	5
19	Basically no technology outsourcing	1	2	3	4	5
20	Developers are inexperienced and have poor development habits	1	2	3	4	5
21	No core technical talents	1	2	3	4	5
22	The tasks of project personnel are complicated	1	2	3	4	5
23	The level of developers is uneven	1	2	3	4	5
24	High employee mobility	1	2	3	4	5
25	Complex site commissioning environment	1	2	3	4	5
26	Software product is complex and quickly updated	1	2	3	4	5
27	Many and disorderly ways of downloading software	1	2	3	4	5

Part III Investigation of Situational Factors of On-Board PIS Software Failure

No.	Description	Very noncompliant	Noncompliant	Uncertain	Compliant	Very compliant
1	No harmonious relationship between leaders and employees of the company	1	2	3	4	5
2	Excessive leadership intervention (project manager)	1	2	3	4	5
3	Lack of support from senior management	1	2	3	4	5
4	The government or another institution has imposed restrictions on the project development	1	2	3	4	5
5	New industry standards or laws and regulations have been issued recently	1	2	3	4	5
6	Unpredictable market turmoil	1	2	3	4	5

Part IV investigation of the Actual Situation of On-Board PIS Software Failure

No.	Description	Very noncompliant	Noncompliant	Uncertain	Compliant	Very compliant
1	Poor stability of software products	1	2	3	4	5
2	Poor maintainability of software products	1	2	3	4	5
3	Poor customer satisfaction with products	1	2	3	4	5

References

1998. *IEEE Standard for Passenger Information System for Rail Transit Vehicles (1477-1998),* USA, USA: IEEE.

2008a. *A guide to the project management body of knowledge (PMBOK® guide),* Newtown Square, Pa, Project Management Institute.

2008b. *ISO/IEC 12207:2008(E) IEEE Std 12207-2008 - Redline: Systems and software engineering -- Software life cycle processes - Redline*, IEEE.

2009. Trustworthy Software Development Processes. *Trustworthy Software Development Processes: International Conference on Software Process, ICSP 2009 Vancouver, Canada, May 16-17, 2009 Proceedings*.

A. EBAD, S. 2016. Influencing Factors for IT Software Project Failures in Developing Countries — A Critical Literature Survey. *Journal of software,* 11**,** 1145-1153.

ABERBACH, J. D. & ROCKMAN, B. A. 2002. Conducting and Coding Elite Interviews. *APSC,* 35**,** 673-676.

AGARWAL, N. & RATHOD, U. 2006. Defining 'success' for software projects: An exploratory revelation. *International Journal of Project Management,* 24**,** 358-370.

AHMEDSHAREEF, Z., HUGHES, R. & PETRIDIS, M. 2014. Exposing the influencing factors on software project delay with actor-network theory. *Electronic journal of business research methods,* 12**,** 132-146.

AL-AZZANI, S., AL-NATOUR, A., AVGERIOU, P., ALI BABAR, M., BAHSOON, R., BENGHAZI, K., BOSCH, J., BUCHGEHER, G., CHUNG, L., COPLIEN, J. O., CLELAND-HUANG, J., CZAUDERNA, A., DíAZ, J., EELES, P., ELORANTA, V.-P., FRIEDRICHSEN, U., GALSTER, M., GARBAJOSA, J., HARCOMBE, S., HOPKINS, R., ISOTTA-RICHES, B., KOSKIMIES, K., GARRIDO, J. L., MIRAKHORLI, M., NOGUERA, M., PéREZ, J., RANDELL, J., REENSKAUG, T., RICO, A., VAN DER VEN, J. S., STAL, M., WEINREICH, R. & YAGüE, A. 2014. List of Contributors. *Agile Software Architecture.* Boston: Morgan Kaufmann.

ALAMI, A. 2016. Why Do Information Technology Projects Fail? *Procedia computer science,* 100**,** 62-71.

ALI, A. 2015. Software Requirements Management. *International Journal of Advanced Research in Artificial Intelligence,* 4**,** 64-68.

ALTER, S. 1979. Implementation Risk Analysis. *TIMS Studies in Management Sciences,* 13**,** 103-119.

ANDA, B. C. D., SJOBERG, D. I. K. & MOCKUS, A. 2009. Variability and Reproducibility in Software Engineering: A Study of Four Companies that Developed the Same System. *IEEE transactions on software engineering,* 35**,** 407-429.

ANDERSSON, C. & RUNESON, P. 2007. A Replicated Quantitative Analysis of Fault Distributions in Complex Software Systems. *IEEE transactions on software engineering,* 33**,** 273-286.

APRIL H. REED & KNIGHT, L. V. 2011. Major virtual project risk factors. *Journal of Information Technology Managemetn,* XXII.

ARNUPHAPTRAIRONG, T. 2011. Top Ten Lists of Software Project Risks: Evidence from the Literature Survey. *The International MultiConference of Engineering and Computer Scientists,* 1.

ASSOCIATION, T. C. U. R. T. 2021. Statistics and Analysis Report of 2020 URT. 3/30**,** 66.

ATKINSON, D. J. 1999. A Case Study Exploration of Groupware Supported Workflow. *Proc. 10th Australasian Conference on Information Systems.*

BANNERMAN, P. L. 2014. A Reassessment of Risk Management in Software Projects. Cham: Springer International Publishing.

BARKI, H., RIVARD, S. & TALBOT, J. 1993. Toward an Assessment of Software Development Risk. *Journal of management information systems,* 10**,** 203-225.

BARRETT, P. 2007. Structural equation modelling: Adjudging model fit. *Personality and individual differences,* 42**,** 815-824.

BARTKO, J. J. 1966. The Intraclass Correlation Coefficient As A Measure Of Reliability *Pshchological Reposrts,* 19(1)**,** 3-11.

BENTLER, P. M. & CHOU, C.-P. 1987. Practical Issues in Structural Modeling. *Sociological methods & research,* 16**,** 78-117.

BOEHM, B. W. 1989. *Software risk management,* Washington, DC, IEEE Computer Society Press.

BOEHM, B. W. 1991. Software risk management: principles and practices. *IEEE software,* 8**,** 32-41.

BROWN, W. J. 1998. *AntiPatterns : refactoring software, architectures, and projects in crisis,* New York ;, Wiley.

BUTLER, T. & FITZGERALD, B. 2001. The relationship between user participation and the management of change surrounding the development of information systems: A European perspective. *Journal of end user computing,* 13**,** 12-25.

ČELAN, M., KLEMENČIČ, M., MRGOLE, A. L. & LEP, M. 2017. Bus-stop Based Real Time Passenger Information System - Case Study Maribor. *IOP Conf. Ser.: Mater. Sci. Eng.* IOP Publishing.

CERPA, N. & VERNER, J. 2009. Why did your project fail? *Communications of the ACM,* 52**,** 130-134.

CHARETTE, R. 1992. Software engineering risk analysis and management. *Computer standards and interfaces,* 14**,** 180.

CHARETTE, R. N. 2005. Why software fails. *IEEE spectrum,* 42**,** 36-43.

CHEN, J.-C. & HUANG, S.-J. 2009. An empirical analysis of the impact of software development problem factors on software maintainability. *The Journal of systems and software,* 82**,** 981-992.

CHIN, W. W., MARCOLIN, B. L. & NEWSTED, P. R. 2003. A Partial Least Squares Latent Variable Modeling Approach for Measuring Interaction Effects: Results from a Monte Carlo Simulation Study and an Electronic-Mail Emotion/Adoption Study. *Information systems research,* 14**,** 189-217.

CHITTISTER, C. & HAIMES, Y. Y. 1994. Assessment and management of software technical risk. *IEEE transactions on systems, man, and cybernetics,* 24**,** 187-202.

CHITTISTER, C. G. & HAIMES, Y. Y. 1996. Systems integration via software risk management. *IEEE transactions on systems, man and cybernetics. Part A, Systems and humans,* 26**,** 521-532.

CHOW, T. & CAO, D.-B. 2008. A survey study of critical success factors in agile software projects. *Journal of Systems and Software,* 81**,** 961-971.

CHYXX.COM. 2021. *2020 年中国铁路客车行业发展现状：拥有铁路客车 7.6 万辆，动车组 31340 辆 [图]* [Online]. https://www.chyxx.com/industry/202109/972158.html. [Accessed].

COMMISSION, T. N. D. A. R. 2004. the Medium- and Long-Term Railway Network Plan.

COMMISSION, T. N. D. A. R. 2008. the New Medium- and Long-Term Railway Network Plan.

CORLEY, K. G. & GIOIA, D. A. 2004. Identity Ambiguity and Change in the Wake of a Corporate Spin-Off. *Administrative science quarterly,* 49**,** 173-208.

CSISZáR, C. & NAGY, E. 2017. Model of an integrated air passenger information system and its adaptation to Budapest Airport. *Journal of air transport management,* 64**,** 33-41.

DE BAKKER, K., BOONSTRA, A. & WORTMANN, H. 2010. Does risk management contribute to IT project success? A meta-analysis of empirical evidence. *International Journal of Project Management,* 28**,** 493-503.

DINMOHAMMADI, F., ALKALI, B., SHAFIEE, M., BéRENGUER, C. & LABIB, A. 2016. Risk Evaluation of Railway Rolling Stock Failures Using FMECA Technique: A Case Study of Passenger Door System. *Urban rail transit,* 2**,** 128-145.

DOHMEN, D. C. 2017. Modelling IT systems for public transport companies: the domain model ITTC. *Transportation Research Procedia,* 25**,** 1846-1864.

DOLL, W. J. 1985. Avenues for Top Management Involvement in Successful MIS Development. *MIS quarterly,* 9**,** 17-35.

DU, S., KEIL, M., MATHIASSEN, L., SHEN, Y. & TIWANA, A. 2007. Attention-shaping tools, expertise, and perceived control in IT project risk assessment. *Decision Support Systems,* 43**,** 269-283.

DWIVEDI, Y. K., WASTELL, D., LAUMER, S., HENRIKSEN, H. Z., MYERS, M. D., BUNKER, D., ELBANNA, A., RAVISHANKAR, M. N. & SRIVASTAVA, S. C. 2015. Research on information systems failures and successes: Status update and future directions. *Information systems frontiers,* 17**,** 143-157.

EASTMONEY.COM. 2021. *2020年中国铁路行业发展现状及经营情况分析疫情影响铁路建设运营目标完成情况* [Online]. https://baijiahao.baidu.com/s?id=1696987035100184591&wfr=spider&for=pc. [Accessed].

EBERLEIN, E. & GRBAC, Z. 2013. RATING BASED LÉVY LIBOR MODEL. *Mathematical finance,* 23**,** 591-626.

ELZAMLY, A. & HUSSIN, B. 2015. Modelling and evaluating software project risks with quantitative analysis techniques in planning software development. *Journal of computing and information technology,* 23**,** 123-139.

ETEMADI, V., BUSHEHRIAN, O. & ROBLES, G. 2022. Task assignment to counter the effect of developer turnover in software maintenance: A knowledge diffusion model. *Information and software technology,* 143**,** 106786.

FAIRLEY, R. 1994. Risk management for software projects. *IEEE software,* 11**,** 57-67.

FENTON, N. E. & OHLSSON, N. 2000. Quantitative analysis of faults and failures in a complex software system. *IEEE transactions on software engineering,* 26**,** 797-814.

FORNELL, C. & LARCKER, D. F. 1981. Evaluating structural equation models with unobservable variables and measurement error. *Journal of marketing research***,** 39-50.

GAMAGE, N. B. J. 2017. The risk factors affecting to the software quality failures in Sri Lankan Software industry.

GAULD, R. 2007. Public sector information system project failures: Lessons from a New Zealand hospital organization. *Government information quarterly,* 24**,** 102-114.

GLASS, R. L. 2001. Extreme programming: The good, the bad, and the bottom line. *IEEE software,* 18**,** 112.

GOPARAJU PURNA, S., AYESHA, F. & SANGHAMITRA, P. 2011. Soft factors affecting the performance of software development teams. *Team Performance Management,* 17**,** 187-205.

GROTTKE, M., DONG SEONG, K., MANSHARAMANI, R., NAMBIAR, M., NATELLA, R. & TRIVEDI, K. S. 2016. Recovery From Software Failures Caused by Mandelbugs. *IEEE transactions on reliability,* 65**,** 70-87.

GROUP, S. 2013. The Chaos Manifesto 2012. 64.

GROUP, T. S. 2004. The 3rd Quarter Research Report.

GUO, L. 2016. Literature Project.

GUO, Y., YU, W., CHEN, Z. & ZOU, R. 2020. Impact of high-speed rail on urban economic development: An observation from the Beijing-Guangzhou line based on night-time light images. *Socio-economic planning sciences,* 72**,** 100905.

GUPTA, S. K., GUNASEKARAN, A., ANTONY, J., GUPTA, S., BAG, S. & ROUBAUD, D. 2019. Systematic literature review of project failures: Current trends and scope for future research. *Computers & industrial engineering,* 127**,** 274-285.

HAIMES, Y. Y., KAPLAN, S. & LAMBERT, J. H. 2002. Risk Filtering, Ranking, and Management Framework Using Hierarchical Holographic Modeling. *Risk analysis,* 22**,** 383-397.

HAMILL, M. & GOSEVA-POPSTOJANOVA, K. 2009. Common Trends in Software Fault and Failure Data. *IEEE transactions on software engineering,* 35**,** 484-496.

HAN, W.-M. & HUANG, S.-J. 2007. An empirical analysis of risk components and performance on software projects. *The Journal of systems and software,* 80**,** 42-50.

HANNIKAINEN, M., LAITINEN, A., HAMALAINEN, T., KAISTO, K. & LESKINEN, K. 2001. Architecture of a passenger information system for public transport services. IEEE.

HAQ, S. U., GU, D., LIANG, C. & ABDULLAH, I. 2019. Project governance mechanisms and the performance of software development projects: Moderating role of requirements risk. *International journal of project management,* 37**,** 533-548.

HARIED, P. & RAMAMURTHY, K. 2009. Evaluating the success in international sourcing of information technology projects: The need for a relational client-vendor approach. *Project Management Journal,* 40**,** 56-71.

HE, D., LU, X. & YAO, X. 2006. P2P and Mobile Agent Based Interactive Communication in the Digital Passenger Information System. IEEE.

HEEKS, R. 2002. Information systems and developing countries: Failure, success, and local improvisations.

HEEKS, R. 2006. Health information systems: Failure, success and improvisation.

HEISKANEN, A., NEWMAN, M. & EKLIN, M. 2008. Control, trust, power, and the dynamics of information system outsourcing relationships: A process study of contractual software development. *The Journal of Strategic Information Systems,* 17**,** 268-286.

HIGUERA, R. P. & HAIMES, Y. Y. 1996. *Software Risk Management.* Carnegie Mellon University.

HIGUERA, R. P. D., A. J, WALKER, J. A, WILLIAMS, R. C 1994. Team Risk Management: A New Model for Customer-Supplier Relationships. *Software Engineering Institute, Carnegie Melton*

University.

HIJAZI, H., ALQRAINY, S., MUAIDI, H. & KHDOUR, T. 2014. Risk factors in software development phases. *European Scientific Journal (Kocani),* 10**,** 213.

HINKIN, T. R. 1995. A Review of Scale Development Practices in the Study of Organizations. *Journal of management,* 21**,** 967-988.

HUANG, S.-J. & HAN, W.-M. 2008. Exploring the relationship between software project duration and risk exposure: A cluster analysis. *Information & management,* 45**,** 175-182.

IEEE 2001. *IEEE Std 1540-2001: IEEE Standard for Software Life Cycle Processes - Risk Management*.

INSTITUTE, B. S. 2017. PD ISO/TS 22163:2017: Railway applications. Quality management system. Business management system requirements for rail organizations: ISO 9001:2015 and particular requirements for application in the rail sector. British Standards Institute.

INSTITUTE, T. P. I. R. 2021. Prospects for the Development Trend of China's Urban Rail Transit Market During the 14th Five-Year Plan Period.

JACKSON, D. L. 2003. Revisiting Sample Size and Number of Parameter Estimates: Some Support for the N:q Hypothesis. *Structural equation modeling,* 10**,** 128-141.

JIANG, J. & KLEIN, G. 2000. Software development risks to project effectiveness. *The Journal of systems and software,* 52**,** 3-10.

JIANG, J. J., KLEIN, G. & DISCENZA, R. 2001. Information system success as impacted by risks and development strategies. *IEEE transactions on engineering management,* 48**,** 46-55.

JINGQIN SU & MIAO CUI 2011. Interview Question Design for Exploratory and Confirmative Cse Study: Theory and Cases. *Chinese Journal of Management,* 8**,** 1428-1437.

JINZHI 2014. 没有需求就没有软件. *电脑世界*.

JONES, C. 1994. *Assessment and control of software risks,* Englewood Cliffs, N. J, Yourdon.

JORGENSEN, M. 1999. Software quality measurement. *Advances in engineering software (1992),* 30**,** 907-912.

JORGENSEN, M. 2014. Failure factors of small software projects at a global outsourcing marketplace. *The Journal of systems and software,* 92**,** 157-169.

KALLE LYYTINEN, B. H. 1987. Information Systems Failures-a Survey and Classification of the Empirical Literature. *Oxford Surveys in Information Technology,* 4**,** 257-309.

KAROLAK, D. W. 1998. Software Engineering Risk Management: A Just-in-Time Approach. *IEEE*.

KERZNER, H. 2017. *Project management : a systems approach to planning, scheduling, and controlling,* Hoboken, New Jersey, John Wiley & Sons, Inc.

KHAN, A. A., BASRI, S. & DOMINC, P. D. D. 2014. A Proposed Framework for Communication Risks During RCM in GSD. *Procedia, social and behavioral sciences,* 129**,** 496-503.

KHAN, H. H. & MALIK, M. N. 2017. Software Standards and Software Failures: A Review With the Perspective of Varying Situational Contexts. *IEEE access,* 5**,** 17501-17513.

KHAN, M. N. A., KHALID, M. & HAQ, S. U. 2013. Review of Requirements Management Issues in Software Development. *International journal of modern education and computer science,* 5**,** 21-27.

KLIEM, R. L. 2000. Risk management for business process reengineering projects. *Information systems management,* 17**,** 71-73.

KLINE, R. B. 2016. *Principles and practice of structural equation modeling,* New York, The Guilford

Press.
KONTIO, J. 2001. *Software Engineering Risk Management: A Method, Improvement Framework, and Empirical Evaluation.* Helsinki University of Technology.
KORU, A. G., EMAM, K. E., ZHANG, D., LIU, H. & MATHEW, D. 2008. Theory of relative defect proneness: Replicated studies on the functional form of the size-defect relationship. *Empirical software engineering : an international journal,* 13**,** 473-498.
KUMAR, S. & KUMAR, T. 2011. STUDY THE IMPACT OF REQUIREMENTS MANAGEMENT CHARACTERISTICS IN GLOBAL SOFTWARE DEVELOPMENT PROJECTS: AN ONTOLOGY BASED APPROACH. *International Journal of Software Engineering & Applications,* 2**,** 107-107.
KUNERT, S. & VON DER WETH, R. 2018. Failure in Projects. Cham: Springer International Publishing.
LAWRENCE H. PUTNAM & MYERS, W. 1992. Measures for Excellence: Reliable Software on Time, Within Budget. *Englewood, NJ: Prentice-Hall*.
LEDERER, A. L. & MENDELOW, A. L. 1988. Information systems planning: Top management takes control. *Business horizons,* 31**,** 73-78.
LEFFINGWELL, D. & WIDRIG, D. 2000. *Managing software requirements : a unified approach,* Reading, Mass. ;, Addison-Wesley.
LEHTINEN, T. O. A., MäNTYLä, M. V., VANHANEN, J., ITKONEN, J. & LASSENIUS, C. 2014. Perceived causes of software project failures – An analysis of their relationships. *Information and Software Technology,* 56**,** 623-643.
LEJLA, T. & NIJAZ, B. 2020. COMPARISON OF STRUCTURAL EQUATION MODELLING AND MULTIPLE REGRESSION TECHNIQUES FOR MODERATION AND MEDIATION EFFECT ANALYSIS. *Sarajevo business and economics review,* 38**,** 29-50.
LI, X., LIU, S., CAI, W. & FENG, S. 2009. The Application of Risk Matrix to Software Project Risk Management. IEEE.
LIN, F., WU, P. & XU, Y. 2021. Investigation of Factors Influencing the Construction Safety of High-Speed Railway Stations Based on DEMATEL and ISM. *Advances in civil engineering,* 2021**,** 1-12.
LINBERG, K. R. 1999. Software developer perceptions about software project failure: a case study. *The Journal of systems and software,* 49**,** 177-192.
LIU, Z. 2000. Discussion on Software Failure Mode, Impact and Hazard Analysis. *Electronic Product Reliability and Environmental Testing***,** 26-29,15.
LOCONSOLE, A. 2004. Empirical Studies on Requirement Management Measures.
LóPEZ, C. & SALMERON, J. L. 2012. Monitoring Software Maintenance Project Risks. *Procedia technology,* 5**,** 363-368.
LU XINYUAN, ZHANG JINLONG & TAO, C. 2006. 企业信息化及风险管理实证分析与研究——以湖北省问卷调查为实证研究. *科研管理,* 27**,** 77-86.
LUCAS, H. C. 1975. *Why Information Systems Fail,* New York, Columbia University Press.
LUHMANN, N. 1993. *Risk : a sociological theory,* Berlin, De Gruyter.
LYNCH, T. & GREGOR, S. 2004. User participation in decision support systems development: Influencing system outcomes. *European journal of information systems,* 13**,** 286-301.
LYYTINEN, K., MATHIASSEN, L. & ROPPONEN, J. 1996. A Framework for software risk management.

Journal of information technology, 11**,** 275-285.

MACHUCA-VILLEGAS, L., GASCA-HURTADO, G. P., PUENTE, S. M. & TAMAYO, L. M. R. 2021. An instrument for measuring perception about social and human factors that influence software development productivity. *Journal of Universal Computer Science,* 27**,** 111-134.

MAO, J.-Y., LEE, J.-N. & DENG, C.-P. 2008. Vendors' perspectives on trust and control in offshore information systems outsourcing. *Information & Management,* 45**,** 482-492.

MARCH, J. G. & SHAPIRA, Z. 1987. Managerial Perspectives on Risk and Risk Taking. *Management science,* 33**,** 1404-1418.

MCFARLAN, F. W. 1981. Portfolio Approach to Information Systems. *Harvard business review,* 59**,** 142-151.

MCLEOD, L. & MACDONELL, S. G. 2011. Factors that affect software systems development project outcomes: A survey of research. *ACM Comput. Surv.,* 43**,** 1-56.

MENEZES JR, J., GUSMãO, C. & MOURA, H. 2019. Risk factors in software development projects: a systematic literature review. *Software quality journal,* 27**,** 1149-1174.

MENG, Y., TIAN, X., LI, Z., ZHOU, W., ZHOU, Z. & ZHONG, M. 2020. Exploring node importance evolution of weighted complex networks in urban rail transit. *Physica A,* 558**,** 124925.

MEREDITH, J. R. & SURESH, N. C. 1986. Justification techniques for advanced manufacturing technologies. *International journal of production research,* 24**,** 1043-1057.

MIZUNO, O., KIKUNO, T., INAGAKI, K., TAKAGI, Y. & SAKAMOTO, K. 2000. Statistical analysis of deviation of actual cost from estimated cost using actual project data. *Information and software technology,* 42**,** 465-473.

MUHAMMAD NAEEM AHMED, K., MUHAMMAD, K. & SAMI UL, H. 2013. Review of Requirements Management Issues in Software Development. *International Journal of Modern Education and Computer Science,* 5**,** 21-27.

NA, K.-S., SIMPSON, J. T., LI, X., SINGH, T. & KIM, K.-Y. 2007. Software development risk and project performance measurement: Evidence in Korea. *Journal of Systems and Software,* 80**,** 596-605.

NATOVICH, J. 2003. Vendor Related Risks in IT Development: A Chronology of an Outsourced Project Failure. *Technology Analysis & Strategic Management,* 15**,** 409-419.

NELSON, R., RYAN 2007. IT PROJECT MANAGEMENT: INFAMOUS FAILURES, CLASSIC MISTAKES, AND BEST PRACTICES. *MIS Quarterly Executive,* 16**,** 67-78.

NEVES, S. M., DA SILVA, C. E. S., SALOMON, V. A. P., DA SILVA, A. F. & SOTOMONTE, B. E. P. 2014. Risk management in software projects through Knowledge Management techniques: Cases in Brazilian Incubated Technology-Based Firms. *International journal of project management,* 32**,** 125-138.

NIZAM, A. 2022. Software Project Failure Process Definition. *IEEE access,* 10**,** 34428-34441.

NONAKA, I. & KONNO, N. 1998. The concept of "Ba": Building a foundation for knowledge creation. *California Management Review***,** 40-54.

NUNDLALL, C. & NAGOWAH, S. D. 2021. Task allocation and coordination in distributed agile software development: a systematic review. *International Journal of Information Technology (Singapore),* 13**,** 321-330.

NUNNALLY 1967. *Psychometric theory*.

OSTRAND, T. J. & WEYUKER, E. J. 2002. The Distribution Of Faults In A Large Industrial Software System. *Proc ACM/International Symposium on Software Test and Analysis,* 2002: 55**,** 55-64.

PALLANT, J. 2016. *SPSS survival manual : a step by step guide to data anlysis using IBM SPSS,* Maidenhead, Open University Press, McGraw-Hill.

PAN, G. & PAN, S. L. 2006. Examining the coalition dynamics affecting IS project abandonment decision-making. *Decision Support Systems,* 42**,** 639-655.

PARK, C.-H. & KIM, Y.-G. 2003. Identifying key factors affecting consumer purchase behavior in an online shopping context. *International journal of retail & distribution management,* 31**,** 16-29.

PINTARD, L., FABRE, J.-C., KANOUN, K., LEEMAN, M. & ROY, M. 2013. Fault injection in the automotive standard ISO 26262: an initial approach. *Dependable Computing*. Springer.

PINTO, J. K. & MANTEL, S. J. 1990. The causes of project failure. *IEEE transactions on engineering management,* 37**,** 269-276.

POLITIS, I., PAPAIOANNOU, P., BASBAS, S. & DIMITRIADIS, N. 2010. Evaluation of a bus passenger information system from the users' point of view in the city of Thessaloniki, Greece. *Research in transportation economics,* 29**,** 249-255.

PONTAKORN SONCHAN, S. R. 2014. Top Twenty Risks in Software Projects: A Content Analysis and Delphi Study. *The 11th Conference on Electrical Engineering/Electronics, Computer, Telecommunication and Information Technology***,** 1-4.

POZGAJ, Z., SERTIC, H. & BOBAN, M. Effective requirement specification as a precondition for successful software development project. Proceedings of the 25th International Conference on Information Technology Interfaces, 2003. ITI 2003., 16-19 June 2003 2003. 669-674.

PRESSMAN, R. S. 2005. *Software engineering : a practitioner's approach,* Boston, Mass. ;, McGraw-Hill.

PRESSMAN, R. S. & MAXIM, B. R. 2015. *Software engineering : a practitioner's approach,* New York, McGraw-Hill Education.

PRICE, J. L. 1997. Handbook of organizational measurement. *International journal of manpower,* 18**,** 305-558.

RALPH JONKERS, R. V. R., GILBERT SILVIUS 2015. The Relationship between Information Systems Strategy and the Perception of Project Success. *International Journal of Information Technology Project Management*.

RAZ, T., SHENHAR, A. J. & DVIR, D. 2002. Risk management, project success, and technological uncertainty. *R & D management,* 32**,** 101-109.

REDMILL, F. 1998. Managing risk: methods for software systems development. Elaine M. Hall. Published by Addison Wesley Longman, Harlow, Essex, U.K., SEI Series in Software Engineering, 1998. ISBN: 0-201-25592-8, 374 pages. Price: U.K. £31.95, Hard Cover. *Softw. Test. Verif. Reliab.* New York: John Wiley & Sons, Ltd.

REEVES, J. D., EVELEIGH, T., HOLZER, T. H. & SARKANI, S. 2013. Risk Identification Biases and Their Impact to Space System Development Project Performance. *Engineering management journal,* 25**,** 3-12.

REIFER, D. J. 2001. Software management's seven deadly sins. *IEEE software,* 18**,** 12-15.

ROCKART, J. F. & CRESCENZI, A. D. 1984. Engaging Top Management in Information Technology. *MIT Sloan management review,* 25**,** 3.

RODRIGUEZ-REPISO, L., SETCHI, R. & SALMERON, J. L. 2007. Modelling IT projects success: Emerging methodologies reviewed. *Technovation,* 27**,** 582-594.

SAMANTRA, C., DATTA, S., MAHAPATRA, S. S. & DEBATA, B. R. 2016. Interpretive structural modelling of critical risk factors in software engineering project. *Benchmarking : an international journal,* 23**,** 2-24.

SARIGIANNIDIS, L. & CHATZOGLOU, P. D. 2014. Quality vs risk: An investigation of their relationship in software development projects. *International journal of project management,* 32**,** 1073-1082.

SAVOLAINEN, P., AHONEN, J. J. & RICHARDSON, I. 2012. Software development project success and failure from the supplier's perspective: A systematic literature review. *International Journal of Project Management,* 30**,** 458-469.

SHIM, J. T., SHEU, T. S., CHEN, H.-G., JIANG, J. J. & KLEIN, G. 2010. Coproduction in successful software development projects. *Information and Software Technology,* 52**,** 1062-1068.

SHITAO, Y., XINGLI, Z., LIN, Y., YUANMING, G. & BIN, Z. 2006. Study on the model-based development approach for the electronically controlled system of a high-pressure common-rail diesel engine. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering,* 220**,** 359-366.

SHOU, Y. & YING, Y. 2005. Critical failure factors of information system projects in Chinese enterprises. IEEE.

SIJUN, B. 2002. 关于项目后评价的探讨. *管理工程学报,* 16**,** 5-7.

SIMãO FILHO, M., PINHEIRO, P. R. & ALBUQUERQUE, A. B. 2017. Verbal Decision Analysis Applied to the Prioritization of Influencing Factors in Distributed Software Development. Cham: Springer International Publishing.

SOUSA, M. J. C. & MOREIRA, H. M. A survey on the Software Maintenance Process. 1998. IEEE, 265-274.

STAPIC, Z., DE-MARCOS, L., STRAHONJA, V., GARCIA-CABOT, A. & GARCIA LOPEZ, E. 2016. Scrutinizing Systematic Literature Review Process in Software Engineering. *TEM Journal,* 5**,** 104-116.

STOICA, R. & BROUSE, P. 2013. IT Project Failure: A Proposed Four-Phased Adaptive Multi-Method Approach. *Procedia Computer Science,* 16**,** 728-736.

STOICA, R. & BROUSE, P. 2014. An Adaptive and Preemptive Theory for Improving Systemic IT Project Failure Trends ("AdaPIT" Theory). *Procedia computer science,* 28**,** 247-256.

SUBRAMANIAN, G. H., JIANG, J. J. & KLEIN, G. 2007. Software quality and IS project performance improvements from software development process maturity and IS implementation strategies. *The Journal of systems and software,* 80**,** 616-627.

SUHARTANTO, D., KUSDIBYO, L., CHEN, B., DEAN, D. & SETIAWATI, L. 2020. Predicting consumer behaviour in tourism industry: Comparing Structural Equation Modelling (SEM) and multiple regression. *IOP conference series. Materials Science and Engineering,* 830**,** 32090.

SUN, X. N., WANG, X. G. & MAI, Y. Y. 2013. The Structural Equation Model for Public Evaluation of

the Transfer Efficiency of Rail Transit P&R Facilities. *Applied mechanics and materials,* 368-370**,** 2027-2033.

SURUGIU, M. C., GHEORGHIU, R. A. & BARNA, O. V. 2017. Environmental Optimization of Urban Travel Using Mobile Devices Detection. *Procedia engineering,* 181**,** 853-860.

TAIPALUS, T., SEPPäNEN, V. & PIRHONEN, M. 2020. Uncertainty in information system development: Causes, effects, and coping mechanisms. *The Journal of systems and software,* 168**,** 110655.

TAMBURRI, D. A., PALOMBA, F. & KAZMAN, R. 2021. Success and Failure in Software Engineering: A Followup Systematic Literature Review. *IEEE transactions on engineering management,* 68**,** 599-611.

TAYLOR, H. 2007. Outsourced IT Projects from the Vendor Perspective: Different Goals, Different Risks. *Journal of Global Information Management,* 15**,** 1-6,13-27.

THIEME, C. A., MOSLEH, A., UTNE, I. B. & HEGDE, J. 2020a. Incorporating software failure in risk analysis—Part 2: Risk modeling process and case study. *Reliability engineering & system safety,* 198**,** 106804-18.

THIEME, C. A., MOSLEH, A., UTNE, I. B. & HEGDE, J. 2020b. Incorporating software failure in risk analysis – Part 1: Software functional failure mode classification. *Reliability engineering & system safety,* 197**,** 106803-13.

THOMAS, J. & MENGEL, T. 2008. Preparing project managers to deal with complexity: Advanced project management education. *International journal of project management,* 26**,** 304-315.

THUMMADI, B. V., SHIV, O., BERENTE, N. & LYYTINEN, K. 2011. Enacted software development routines based on waterfall and agile software methods: Socio-technical event sequence study.

TIBAUT, A., KAUČIČ, B. & REBOLJ, D. 2012. A standardised approach for sustainable interoperability between public transport passenger information systems. *Computers in industry,* 63**,** 788-798.

TRENDOWICZ, A., OCHS, M., WICKENKAMP, A., MüNCH, J., ISHIGAI, Y. & KAWAGUCHI, T. 2008. An Integrated Approach for Identifying Relevant Factors Influencing Software Development Productivity. Berlin, Heidelberg: Springer Berlin Heidelberg.

TSUNODA, M., MONDEN, A., MATSUMOTO, K., OHIWA, S. & OSHINO, T. 2021. Analysis of Work Efficiency and Quality of Software Maintenance Using Cross-Company Dataset. *IEICE transactions on information and systems,* E104.D**,** 76-90.

VAN GENUCHTEN, M. & KOOLEN, H. 1991. On the use of software cost models. *Information & management,* 21**,** 37-44.

VERNER, J. M. & ABDULLAH, L. M. 2012. Exploratory case study research: Outsourced project failure. *Information and software technology,* 54**,** 866-886.

VITHARANA, P. 2015. Defect propagation at the project-level: results and a post-hoc analysis on inspection efficiency. *Empirical software engineering : an international journal,* 22**,** 57-79.

VRHOVEC, S. L. R., HOVELJA, T., VAVPOTIČ, D. & KRISPER, M. 2015. Diagnosing organizational risks in software projects: Stakeholder resistance. *International journal of project management,* 33**,** 1262-1273.

WALLACE, L. & KEIL, M. 2004. Software project risks and their effect on outcomes. *Communications of the ACM,* 47**,** 68-73.

WALLACE, L., KEIL, M. & RAI, A. 2004a. How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model. *Decision sciences,* 35**,** 289-321.

WALLACE, L., KEIL, M. & RAI, A. 2004b. Understanding software project risk: a cluster analysis. *Information & management,* 42**,** 115-125.

WALSHAM, G. 1993. *Interpreting information systems in organizations,* Chichester, Wiley.

WANG, H., YU, F. R. & WANG, H. 2017. A cognitive control approach to interference mitigation in communications-based train control (CBTC) co-existing with passenger information systems (PISs). *EURASIP journal on wireless communications and networking,* 2017**,** 1-13.

WANG, N. 2009. *企业信息技术应用能力的理论和实证研究*. 博士，东南大学.

WANSTROM, L. 2002. *Sample Sizes for Two-Group Second Order Latent Growth Curve Models.* Stockholm University.

WARD, J., DANIEL, E. & PEPPARD, J. 2008. BUILDING BETTER BUSINESS CASES FOR IT INVESTMENTS. *MIS quarterly executive,* 7**,** 1-15.

WHITNEY, K. M. & DANIELS, C. B. 2013. The Root Cause of Failure in Complex IT Projects: Complexity Itself. *Procedia Computer Science,* 20**,** 325-330.

WILSON, M. & HOWCROFT, D. 2002. Re-conceptualising failure: social shaping meets IS research. *European journal of information systems,* 11**,** 236-250.

WYSOCKI, R. K., KAIKINI, S. & SNEED, R. 2014. *Effective project management : traditional, agile, extreme,* Indianapolis, Indiana, Wiley.

XUE SIXIN, J. G. 2004. Project Management for Software. *China Machine Press,* 4**,** 36-56.

YAN, B. 2020. Improvement of the Economic Management System Based On the Publicity of Railway Transportation Products. *Intelligent Automation and Soft Computing,* 26(3)**,** 539-547.

YAO, X. Y., REN, C. H., HE, D. Q. & TANG, H. 2013. Research on passenger information system based on train security detection sensor network.

YE, W. 2006. *Failure Factors Analysis of Small- and Medium-sized Software Projects Based on PLS Method.* Master, Zhejiang University.

YEO, K. T. 2002. Critical failure factors in information system projects. *International journal of project management,* 20**,** 241-246.

YIP, S. W. L. & LAM, T. A software maintenance survey. 1994. IEEE Comput. Soc. Press, 70-79.

YOUNG, R. & JORDAN, E. 2008. Top management support: Mantra or necessity? *International journal of project management,* 26**,** 713-725.

YOUNG, R. & POON, S. 2013. Top management support—almost always necessary and sometimes sufficient for success: Findings from a fuzzy set analysis. *International journal of project management,* 31**,** 943-957.

YU, X. 2012. Design and Optimization on Train-Ground Wireless Communication of Passenger Information System of Urban Rail Transit. *Applied Mechanics and Materials,* 236-237**,** 964.

ZAHID, A. H., HAIDER, M. W., FAROOQ, M. S., ABID, A. & ALI, A. 2018. A Critical Analysis of Software Failure Causes From Project Management Perspectives. *VFAST Transactions on Software Engineering***,** 113-119.

ZENG WU-YI & BING-YI, H. 2005. 调查问卷的可信度和有效度分析. *统计与信息论坛,* 20**,** 11-15.

ZHANG, L., AKIFUJI, S., KAWAI, K. & MORIOKA, T. 2006. Comparison between Test Driven Development and waterfall development in a small-scale project. *Extreme Programming And Agile Processes In Software Engineering, Proceeding,* 4044**,** 211-212.

ZHANG QING & HUANG, L. 2003. IT 投资价值评价研究综述. *外国经济与管理,* 25**,** 35-40.

ZHIVICH, M. & CUNNINGHAM, R. K. 2009. The Real Cost of Software Errors. *IEEE security & privacy,* 7**,** 87-90.

ZHOU DEQUN, F. Z. 2010. *系统工程概论*, 科学出版社.

崔耀东, 周., 廖文和 2001. 制造业信息系统应用评价研究. *系统过程,* 19（6）**,** 64-70.