J. Software Engineering & Applications, 2009, 2: 173-194
doi:10.4236/jsea.2009.23025 Published Online October 2009 (http://www.SciRP.org/journal/jsea)

Research on Software Production Support Structure

Jiangping WAN[1,2]

[1]School of Business Administration, South China University of Technology, Guangzhou, China; [2]Institute of Emerging Industrialization Development South China Univ. of Tech., Guangzhou, China.
Email: csjpwan@scut.edu.cn

Received March 11th, 2009; revised July 10th, 2009; accepted 12th, 2009.

ABSTRACT

Firstly, it is found that process design is necessary for software process improvement after analyzing its complexity. Then, research methods and concepts framework are put forward, and the research content is also provided. The findings of research, including propositions of complexity of software process, the work program of complexity of software process improvement, software enterprise model and software production support structure are clarified. Finally, the demonstration, including mindbugs (cognitive barriers) in software process and the knowledge integration support structure of quality software production, is illustrated with case study. It is concluded that the research is useful for both software production and knowledge economy in the future.

Keywords**:** *Complexity, Mindbugs, Software Process, Work Program of Complexity, Software Enterprise Model, Interactive Management, Quality Software Production, Knowledge Integration*

1. Introduction

Since the computer software is a kind of logical product, its quality improvement is difficult and complex. Many researchers are trying to reduce the hardship and the cost as well. Nowadays, it is going to focus on the software process of software production. Software process is the set of tools, methods, and practices used to produce a software product. The objectives of software process improvement (SPI) are to produce products according to the plan while simultaneously improving the organization's capability to produce better products. It is clear that a fully effective software process must consider the relationships of all the required tasks, including the tools and methods used, the skill, training, and motivation of the people involved [1–2].

An economist, Howard Baetjer, commented on the software process as following [3]: as software likes all capital, it is concrete knowledge, and because that knowledge is initially dispersed, tacit, latent, and incomplete in large measure, software development is a social learning process. The process is a dialogue in which the knowledge on the software is brought together and embodied in the software. The process enables interaction between users and designers, between users and evolving tools, and between designers and evolving tool (technology). It is an iterative process in which the evolving tool itself serves as the medium for communication, with each new round of the dialogue eliciting more useful knowledge from the people involved. It is obvious that software process is also an organizational knowledge-intensive learning process and needed to be supported with knowledge management.

Warfield argued that normal problems involved local or occasionally intermediate logics, but complex problems involved deep logic. Since deep logic is generally absent from representations, or if present was often masked by being embedded in thicket-like prose, the consequence often was under-conceptualization and under-documentation, as well as poor communication. Just as Aristotle said that logic was measure to reach knowledge, and it was necessary to enhance organizational learning with process design for knowledge management. The variety of fundamental operations that can be carried out with ideas is quite limited. Almost everything that needs to be done can be conceived as 1) generating ideas, 2) clarifying ideas, 3) structuring ideas, 4) interpreting structures of ideas, and 5) amending ideas. The limited number of “idea actions” means that the variety of processes needed can also be quite limited. One only needs to get processes for clarifying ideas, structuring ideas, and interpreting the structures produced. Design consists primarily of three types of intellectual activity: conceptualization, choice, and documentation. The implementation of design is its most concrete phase, but the failure

This research was supported by Key Project of Guangdong Province Education Office (06JDXM63002), Soft Science project of Guangdong Province(2007B070900026), NSF of China (70471091), and QualiPSo (IST- FP6-IP-034763).

of any one of these three prior types will usually assure the failure of the implementation. Intelligence, analysis, and synthesis make up conceptualization. Communication and interpretation make up documentation [4].

Interactive Management is a system designed specifically and painstakingly for the purpose of helping people resolve complexity in organizations [5]. And 20 laws of complexity were put forward [6]. A science of complexity integrates all of the material in order to show both the theory of complexity and the empirical evidence that has been accumulated to show the validity of the theory. A principal outcome of the science of complexity is called the work program of complexity (WPOC). This WPOC consists of two main steps: discovery and resolution of complexity. Discovery consists of description and diagnosis. Resolution consists of design and implementation [7]. WPOC was applied by a large cross-functional team of Ford engineers and system developers in the mid-1990s as an enabler to create an enterprise-wide information system [8].

2. Software, Software Production and Their Complexity

Recall that this title of Brooks's article is “No Silver Bullet” [9]. Brooks's theme is that the very nature of software makes it highly unlikely that a silver bullet will be discovered that will magically solve all the problems of software production, let alone help to achieve software breakthroughs that are comparable to those that have occurred with unfailing regularity in the hardware field. He divides the difficulties of software into two Aristotelian categories: **essence**, the difficulties inherent in the intrinsic nature of software, and **accidents**, the difficulties that are encountered today but are not inherent in software product. He lists four, which he terms complexity, conformity, changeability, and invisibility. In the context of his article, Brooks uses the word complex in the sense complicated or intricate. In fact, the names of all four aspects are used in their non-technical sense.

Complexity. It is an inherent property of software. Brooks points out that complexity phenomena can be described and explained in disciplines such as mathematics and physics. In contrast, if software is simplified, the whole exercise is useless; the simplifying techniques of mathematics and physics work only because the complexities of those systems are accidents, not essence, as is the case with software products.

Conformity. The first type of conformity identified by Brooks, software acquires an unnecessary degree of complexity as it has to interface with an existing system. The second type identified by Brooks, where software acquires an unnecessary degree of complexity because of the misconception that software is the component that is the most conformable. In other words, it makes the complexity of software be escalated.

Changeability. The functionality of a system embodied in its software, and functionality changes are achieved through changing the software. Boorks points out that that changeability is a property of the essence of software and an inherent problem that cannot be surmounted.

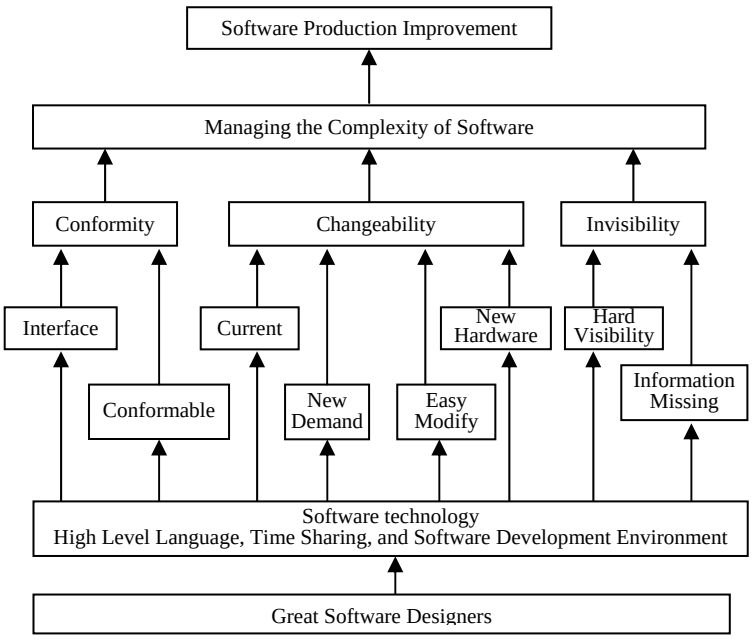

Figure 1. Approach of software production improvement by brook

Invisibility. The result of this inability to represent software visually not only makes software difficult to comprehend, it also severely hinders communication among software professionals which causes that there seems no alternative to handing a colleague a 150-page listing together with a list of modifications to be made.

Brooks considers the three major breakthroughs in software technology, namely, high level language, time sharing, and software development environment, but stresses that they solved only accidental, and not essential difficulties. For Brooks, the greatest hope of a major breakthrough in improving software production lies in training and encouraging great designers. It can be illustrated in Figure 1.

In J. N. Warfield's words [4], we could manage the complexity of software product through process design. Processes reduce personal mindbugs (cognitive barriers) primarily through the way in which information is sequenced. If members of groups are given rein to choose the topic of their discussion at random, and if several members speak at the same time, sensible discussion that leads to some organized product is hard to obtain. But if the process is designed so that the subject is broken down into a series of carefully designed questions which are presented under computer control for group discussion and resolution, the dialog becomes focused and the products of the dialog can be aggregated and organized with ease. Such a design will meet with group approval, provided the group is assured that important subjects will not thereby be excluded, and that their contributions will ultimately be incorporated on all matters perceived by group members to be relevant.

3. Research Methods

The complexity of SPI was studied through investigation on spot and literature reading. There are three problems as following: 1) What is the theory foundation of WPOC being applied to SPI? 2) How to design the WPOC of SPI? What kind of knowledge technology is necessary? 3) How to combine WPOC of SPI with the business goals of software enterprise?

The concept framework of study is established through learning Warfield's complexity theory, software engineering and quality management. The theory hypothesis is put forward that WPOC can be applied to SPI, and then the propositions of complexity of software process are educed. The propositions are theory foundation of WPOC of SPI. Basing on Warfield' WPOC, WPOC of SPI is designed. So do software enterprise model based on Microsoft's enterprise model and Infosys' knowledge management. Finally, software product support structure is established and its rationality and validity are illustrated by demonstration study [10].

The research concept framework is illustrated in Figure 2. It is established according to generating ideas, clarifying ideas, structuring ideas, interpreting structures of ideas and amending ideas. At first, the goal of software quality improvement is essential to SPI, and SPI is abstract to the theory of SPI, and the theory of SPI is structural to WPOC of SPI, and WPOC of SPI is organizational to software enterprise model and at last software enterprise model is commercial to market competition by business operation [10–18].

4. Research Content

The logical structure of the research content is illustrated in Figure 3. The contents which dashed frames represented are involved in the research. The research finds that four software essence aspects are similar to Warfield's complexity and Warfield's complexity theory can be applied to SPI to conclude five propositions of software developmental process and seven propositions of software support process whose rationality and validity are proved by successful practices of software enterprises.

Software process model regulates not only every phase task and activity in details, hierarchical sequences among tasks and activities, but also establishes the order and

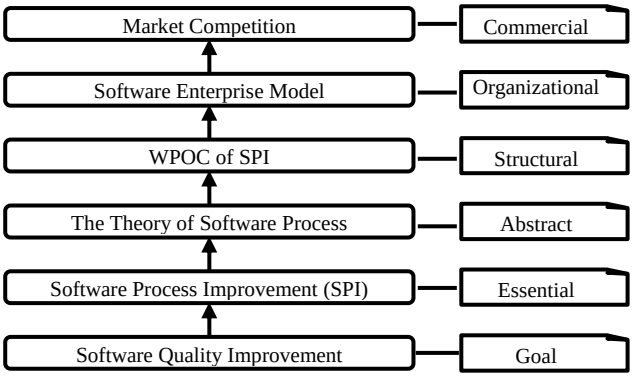

Figure 2. Diagram of concept framework

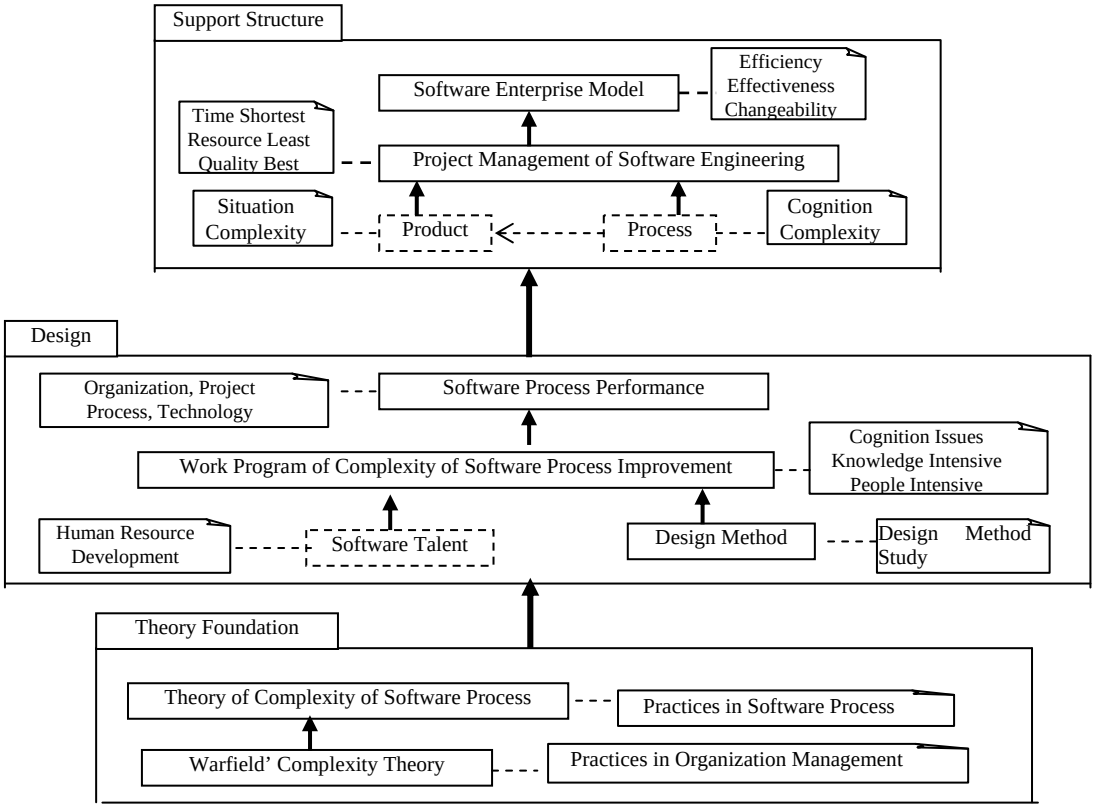

Figure 3. Diagram of the research content

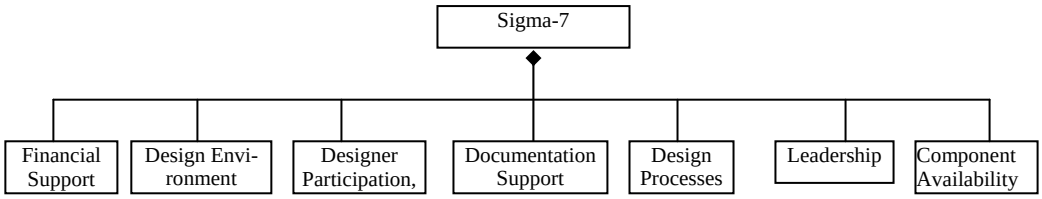

Figure 4. Class diagram of sigma-7

restrictions in every phase of software development and evolution and specifies the guideline from one phase to another. Meanwhile, it also gives the restrictions of the policy which should be followed in software development.

It is necessary to combine WPOC of SPI with software project management to realize the business goals of enterprise. The enterprise model can provide support to business operation by project because the concrete software process is determined by product including service in software engineering project management [10,18].

5. Propositions of Complexity of Software Process

The software process can be classified to software development process (SDP) and software support process (SSP) according to M. Porter's value chain model. Five complexity propositions of SDP are put forward through applying principle of generic design to seven principles of software engineering by B. W. Boehm [19]. Finally, seven complexity propositions of SSP are educed through applying principle of generic design to six principles of SPI by W. S. Humphrey [1,10].

5.1 Sigma-N Concepts

The Sigma represents the idea of integration [4], and the N represents factors to be integrated in order to achieve success. It is illustrated in Figure 4.

To make this clear, supposing that in some particular design situation, context makes it clear that financial support and component availability are readily available, but the critical factors of leadership, design environment, designer participation, documentation support and design processes have not been integrated. Then we would say

that we are dealing with a potential Sigma-5 situation. It has been found in practice that the Sigma-7 and Sigma-5 concepts are the most significant, with other numerical values such as Sigma-1 and Sigma-2 representing situations that are normally ineffective in dealing with complex issues.

5.2 Five Complexity Propositions of Software Development Process

The basic principles of software engineering by B. W. Boehm are in the following [19]:

Principle of plan (BP1): Manage using a phased life-cycle plan.

Principle of review (BP2): Perform continuous validation.

Principle of control (BP3): Maintain disciplined product control.

Principle of technology (BP4): Use modern programming practices.

Principle of visualization (BP5): Maintain clear accountability for result.

Principle of performance (BP6): Use better and fewer people.

Principle of process (BP7): Maintain a commitment to improve the process.

The relationship of six principles of software engineering with Sigma-5 are illustrated in Figure 5.

The principles that have related with design environment include BP4, BP6 and BP7.

The principles that have related with designer participation include BP4, BP6 and BP7.

The principles that have related with documentation support include BP1, BP2, BP3, BP4, BP5, BP6 and BP7.

The principles that have related with design processes include BP1, BP2, BP3, BP4, BP5, BP6 and BP7.

The principles that have related with leadership include BP1, BP3, BP4, BP6 and BP7.

There are two ubiquitously critical factors among Sigma-5: documentation support and design processes. It is necessary for software engineering to design processes of reducing the cognitive barriers (mindbugs) to improve intelligence activities' efficiency and quality, and visualize intellectual activities for management.

There are three ubiquitously principles among the seven principles of software engineering: technology principle, process principle, and performance principle. It is necessary for software engineering to learn new technology and make continuous improvement because of intelligence-intensive teamwork.

Five propositions of complexity of SDP are as follow (Figure 6): 1) Proposition of capability (SDP1). SDP is required continuously organizational learning in order to improve SDP capability of organization; 2) proposition of quality (SDP2). SDP is required to have high quality products and services through efficient cooperation; 3) Proposition of process (SDP3). SDP is required to adopt carefully design processes in order to reduce personal cognitive burden and improve group cognitive activities effectively and efficiently; 4) Proposition of documentation (SDP4).

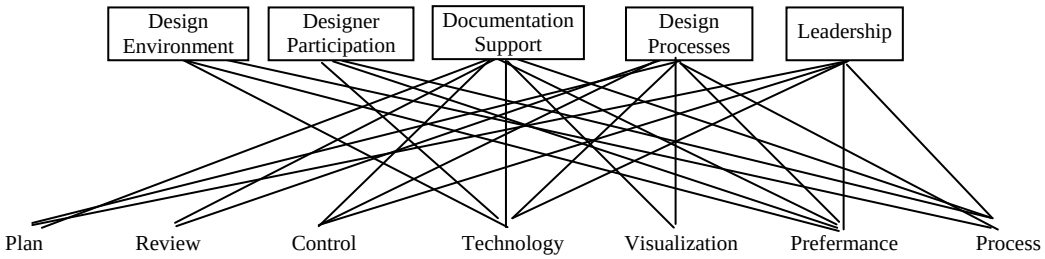

Figure 5. Relationship seven principles with sigma-5

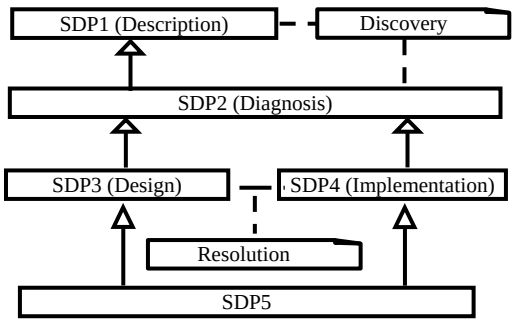

Figure 6. Five propositions of complexity of SDP

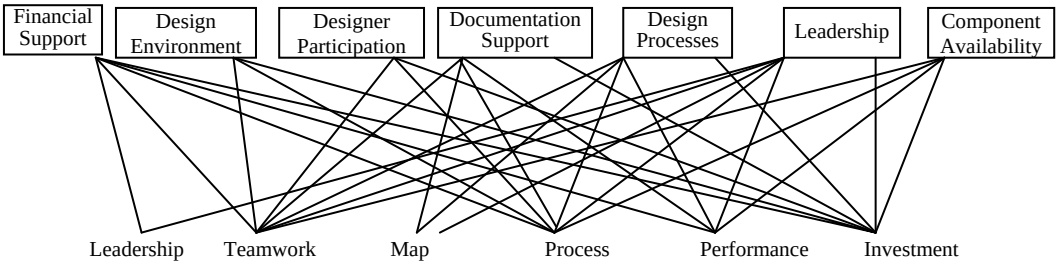

Figure 7. Relationship six principles of SPI with sigma-7

SDP is required to make documentation to communicate and interpret to visualize activities of SDP in order to control them; 5) Proposition of technology (SDP5). SDP is required to use modern software technology. The SDP1 (capability) and SDP2 (quality) are mostly based on behavioral science and goals of software engineering, but SDP3 (process), SDP4 (documentation) and SDP5 (technology) are mostly based on deep logic and software formal technology, their relationship are illustrated in Figure 6 (The hollow triangle represents inherited relation, e.g. if the child is similar to his parent, the child inherits his parent's characters. Here is to embody the proposition. e. g. the SDP1 is embodied in SDP2, SDP2 is embodied in both SDP3 and SDP4, and both SDP3 and SDP4 are embodied in SDP5. The hierarchy of concept is analyzed with object technology here, the proposition is a kind of abstract concept, and the embodying can be considered as implementation of lower abstraction). It is necessary to apply WPOC to software engineering with the propositions of SDP. Discovery is needed by SDP1 (description) and SDP2 (diagnosis), and resolution is needed by SDP3 (design) and SDP4 (implementation). WPOC must apply modern software technology according to SDP5.

5.3 Seven Complexity Propositions of Software Support Process

The basic principles of SPI by Watts S. Humphrey are in the following [1]:

Principle of leadership (HP1). Major changes to the software process must start at the top. Senior management leadership is required to launch the change effort and to provide continuing resources and priority.

Principle of teamwork (HP2). Ultimately, everyone must be involved. Software engineering is a team effort, and anyone who does not participate in improvement will miss the benefits and may even inhibit progress.

Principle of map (HP3). Effective change requires a goal and knowledge of the current process. To use a map, you must know where you are.

Principle of process (HP4). Change is continuous. Software process improvement is not a one-shot effort; it involves continual learning and growth.

Principle of performance (HP5). Software process changes will not be retained without conscious effort and periodic reinforcement.

Principle of investment (HP6). Software process improvement requires investment. It takes planning, dedicated people, management time, and capital investment.

The relationship six principles of SPI with Sigma-7 are illustrated in Figure 7.

The principles that have related with financial support include HP1, HP2, HP4, HP5 and HP6.

The principles that have related with design environment include HP2, HP4 and HP6.

The principles that have related with designer participation include HP2, HP4 and HP6.

The principles that have related with documentation support include HP2, HP3, HP4, HP5 and HP6.

The principles that have related with design processes include HP2, HP3, HP4, HP5 and HP6.

The principles that have related with leadership include HP1, HP2, HP3, HP4, HP5 and HP6.

The principles that have related with component availability include HP2, HP4, HP5 and HP6.

There are four ubiquitously critical factors among Sigma-7: financial support, documentation support, design processes and leadership. It is necessary for SPI to design processes through documentation with financial support.

There are four ubiquitously principles among the six principles of SPI: team principle, process principle, performance principle and investment principle. It is necessary for SPI to organize teamwork with carefully design processes through both excellent leadership and enough investment to get satisfactory performance.

Seven propositions of complexity of SSP (Figure 8) are in the following: 1) Proposition of capability (SSP1). SSP is required to make continuously organizational learning in order to improve software process capability of organization; 2) Proposition of performance (SSP2). SSP will not be retained without conscious effort and periodic reinforcement; 3) Proposition of leadership (SSP3). SPI must start from the top. Senior management leadership is required to launch and participate in SSP; 4) Proposition of process (SSP4). SSP is required to adopt carefully design processes in order to reduce personal cognitive burden

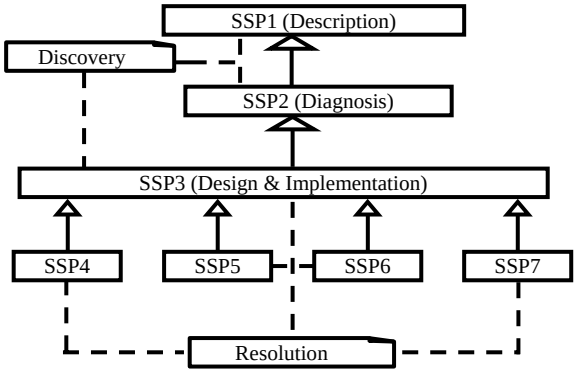

Figure 8. Seven propositions of complexity of SSP

and improve group cognitive activities effectively and efficiently; 5) Proposition of documentation (SSP5). SSP is required to make documentation to communicate and interpret for visualizing activities of SPI to control them; 6) Proposition of teamwork (SSP6). Everyone must be involved. SSP is a team effort, and anyone who does not participate in SPI will miss the benefits and may even inhibit progress; 7) Proposition of investment (SSP7). SSP is required to be projected, assigned full time stuffs, managed time and to be invested.

Watts S. Humphrey said that the key topics to focus on once the decision has been made to invest in process improvement are in the following [1]: 1) To improve the software process, someone must work on it (SSP3, SSP6, SSP7); 2) Unplanned process improvement is wishful thinking (SSP4, SSP5); 3) Automatic of a poorly defined process will process poorly defined results(SSP1, SSP2, SSP3); 4) Improvements should be made in small, tested steps(SSP1, SSP2, SSP3); 5) Train, train, train(SSP1, SSP3, SSP7).

According to a systematic survey of CMM experience and results by SEI (Software Engineering of Institute), there are seven empirical principles of successful SPI in the following [2]:

EP1. Management control is tightly related with the successful SPI (SSP2, SSP4, SSP5).

EP2. To establish software process group, which includes technicians, correctly, and assign SPI responsibility clearly and enough (SSP4, SSP6).

EP3. To take SPI carefully and retain experienced specialists and refer outside seriously (SSP1, SSP7).

EP4. To estimate the progress and cost of engineering correctly (SSP1, SSP2, SSP4, SSP7).

EP5. Managers consider seriously the influences, which come from software organizational culture, external business environment and internal environment, on SPI (SSP1, SSP2, SSP3).

EP6. To conduct operation of software correctly (SSP4, SSP5, SSP6, SSP7).

EP7. To establish software quality plan perfectly (SSP1, SSP2, SSP3, SSP4, SSP5, SSP6, SSP7).

6. Work Program of the Complexity of Software Process Improvement

The WPOC of SPI consists of two phases (Figure 9): discovery and resolution [10].

Discovery includes two steps: 1) Describing the software process, the basic activities of software process are described and their mindbugs are identified and classified. It is necessary to consider the mindbugs, because software process is usually a complexity group cognitive process. Object technology, a kind of knowledge technology, is useful since it conforms to human cognitive habits and can reduce cognitive burden effectively. It suggests that the basic activities of software process are firstly described with object technology to understand enough the architecture and concretely activities in details, the mindbugs of software process are identified and classified for diagnosing software process. 2) Diagnosing the software process, Software Capability Maturity Model (SW-CMM) by SEI is an effective method and tool. SW-CMM can be analyzed with object technology to understand its essence that software process performance is the core of SW-CMM and SW-CMM can be used to illuminate performance structure and target of software process.

Resolution also consists of two steps. 1) Designing SPI, three popular SPI models are IDEAL model (process model includes Initial, Diagnosis, Establishment, Action and Learning five steps and is a de fact implementation methodology of SW-CMM) by the HP corporation and SEI, three-phrase method (product model includes Understanding, Assessing, and Packaging three steps) by

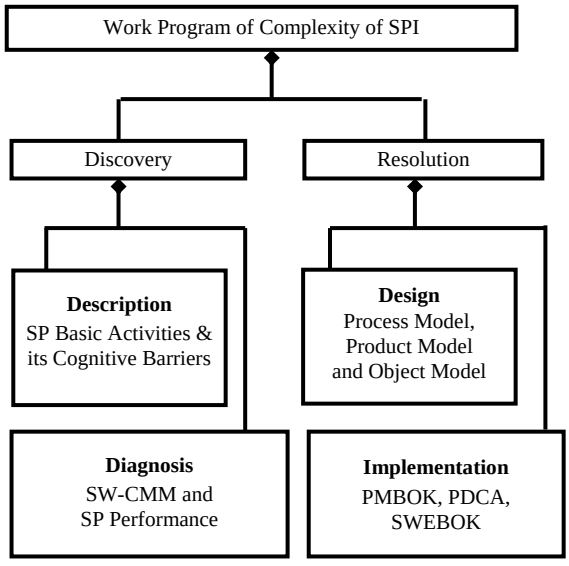

Figure 9. Work program of complexity of SPI

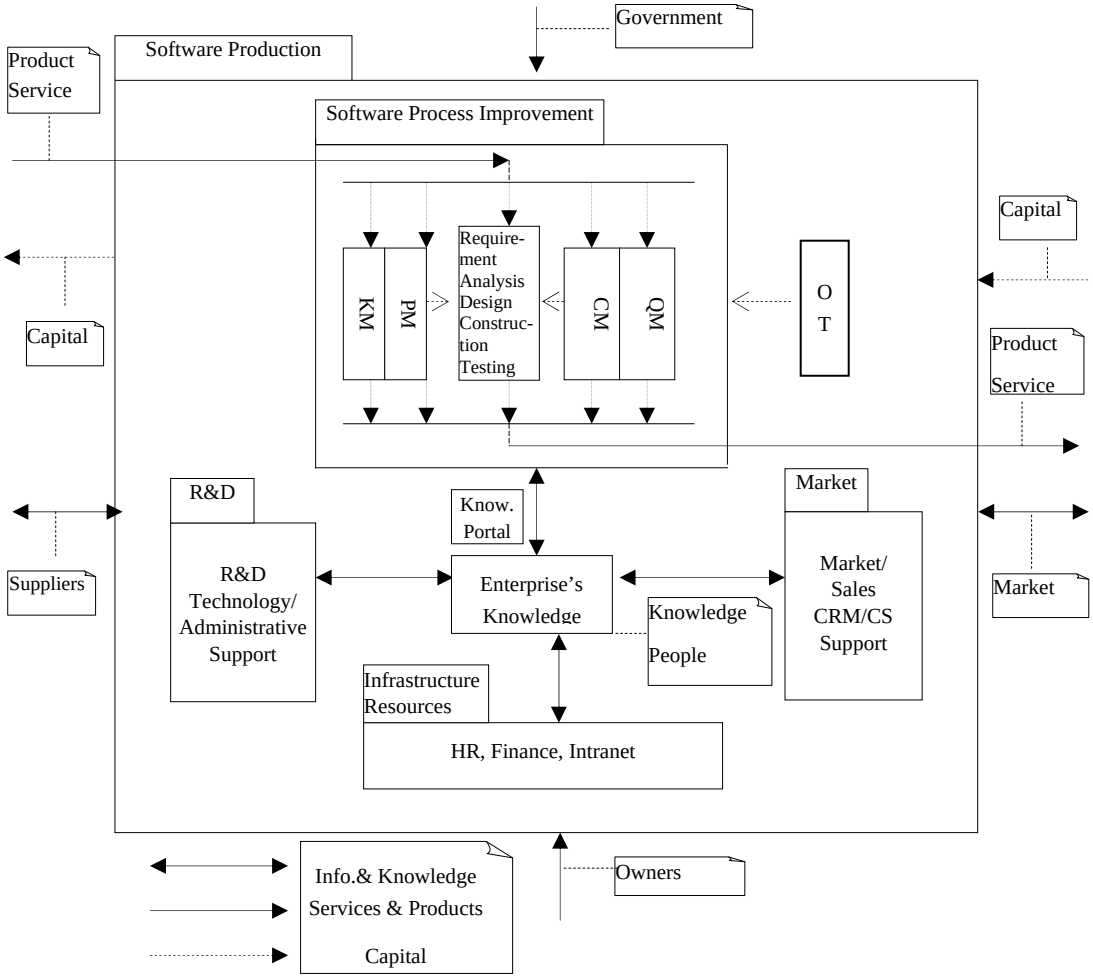

Figure 10. Knowledge enabling enterprise model for software production

NASA, and Rational Unified Process (object model includes Inception, Elaboration, Construction and Transition four phases and nine core process workflows, which are business modeling, requirement, analysis and design, implementation, testing, project management, configuration, change management and environment etc.). It is important to understand the essence that there are both merits and demerits in three models. When designing software processes, it is necessary to actively think about real contexts and issues in order to apply the models in a creative way, not in a rote way. 2) Implementation SPI. The first important knowledge is project management, which is often achieved with half effort to do SPI according to project management in real world. The second is method of quality management used in traditional industry, e.g. PCDA cycle by W. E. Deming, which is used in SPI by NEC do Brazil (NDB), but the traditional industry is quite different from software industry, which is knowledge-intensive and brain-intensive, so IEEE SWEBOK (Software Engineering Body of Knowledge) is necessary. These are knowledge technologies needed to implement SPI.

7. Knowledge Enabling Enterprise Model for Software Production

A model is an abstract representation of reality expressed in terms of some formalism. An enterprise model may be used for various purposes such as facilities design, systems architecture, organization, simulation, optimization, performance measurement and benchmarking. Here it will be used to describe and understand the operational of a software organization [13,18].

A knowledge enabling model for software enterprise with generic structure is illustrated in Figure 10. The notation of UML is used. The outer boundary frames business operation. Inputs are services and products, capital, information and knowledge. Outputs are services and products, capital, information and knowledge. Governmental regulations and other frame conditions may be regarded as control. Owners and external organizations

comprise the mechanism. Software production box includes market, sale, Customer Relationship Management (CRM) and Customer Service (CS). Research and Development (R&D), technological and administrative support functions, infrastructure resources (any kind, e.g. Human Resources and Intranet etc.), organizational knowledge portal (the knowledge keys to successfully software project), as well as SPI. The market is goal, while sale is means and CRM & CS are assurance. R&D is the core element, technological and administrative supporting are interfaces and infrastructure resources. The inner box, software process improvement (development), includes software process (inception, elaboration, construction and transition, them are not illustrated in the Figure 10) and core workflows and activities supporting. The core workflows are in the following: requirement, analysis, design, construction and testing. All of the development activities are architecture based (OT, object technology), including configuration (change) management (CM), project management (PM, plan, control, cooperation and standard etc.), quality management (QM, including analysis and assurance), knowledge management (KM) in SPI, Knowledge portal is facility for knowledge accumulating, sharing and communication in organization scope, KM should be supported by HR, Finance, Intranet etc.

8. Software Production Support Structure

Basing on interactive management support structure [5,12], software production support structure can be illustrated in Figure 11.

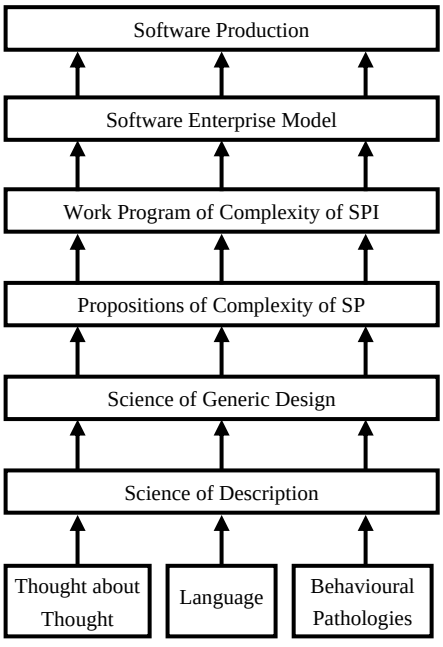

Figure 11. Software production support structure

It consists of boxes that contain text and arrows that connect the boxes. The arrows are all oriented upward. They show the directional flow of support**.** If there is a directed path from some box to some higher level box, the one below supports the one at the other end of the path. This is the basic rule for understanding this support structure. The word "support" is the key point. It means that you will find necessary information in the lower box for the development of what appears above. In this instance, you can see, for example, that software production support structure is supported by two sciences and one theory and one process and one model. The two sciences are all supported by these three principal components: 1) **Thought about thought** (which J. N. Warfield called "second-order thought"); 2) **Language**; 3) **Behavioural pathologies** (which is the information about human limitations, individually, in small groups, and in larger organizations; the information coming from the research in psychology, sociology, and organizational studies).

The date 350 B.C. corresponded roughly to the period of Aristotle's work in logic, in which he developed the first formal statement of deductive logic, along with the concept of categorizing topics so that it would be possible to work with and compare collections of ideas, as well as with the individual ideas themselves. Prof. Warfield has described the pattern that evolved over the many centuries that brought thought about thought into our view today, so that those ideas find their ways into SPI, where they help people resolve complexity of SPI. The idea is applied to Chinese enterprise on the condition that Warfield's theory must be combined with Chinese traditional culture and thought, such as Change of Book and "Shili-Wuli-Renli" etc.

A Science of Generic Design [4]: This science focused upon the use of information coming from a high- quality description to develop a set of options from which an alternative design could be chosen for implementation. It focuses upon the design of socio-technical systems. Software engineering, which is also socio- technical systems, can be applied the science of generic design. The propositions of SDP and SSP are educed and become the theory foundation of SPI. This theory can guide the design of WPOC of SPI (structural presentation).

When working on a software project, software engineers apply principles based on the foundation to software developing processes and the product through forming specific and pragmatic methods and tools [18]. It is necessary for complexity interactive cognition processes because that SPI should be combined with software project management, which is needed to conform to enterprise business goals. Software enterprise model can be considered as an application model of e-business for software enterprise [10, 15, 18, 20].

9. Demonstration

Demonstration includes two parts as following: 1) Prof. J. N. Warfield has tried to identify and name each distinctive origin of one or more behaviorally related symptoms. Mindbugs (cognitive barriers) according to SW- CMM are studied. The research finds that the most common mindbugs of SPI is misattribution of consensus. 2) The knowledge integration support structure of quality software production is illustrated with case study.

9.1 Mindbugs in Software Process

9.1.1 The Identification and Classification of Mindbugs

Warfield have tried to identify and name each distinctive origin of one or more behaviorally-related symptoms (as "mindbugs" to bring the language in line with contemporary computer languages) [21]. So far, twenty-five mindbugs have been identified, which are grouped into four categories: Mindbugs of Minsinterpretation: those where concepts are misconstrued or misattributed, because of faulty interpretation, Type M; Mindbugs of Clanthink: those where concepts are very widely perceived to be correct, but are demonstrably incorrect, Type C; Mindbugs of Habit: those which involve ingrained behavior, evinced with essentially no conscious thought, Type H; and Mindbugs of Error: just plain mistakes, Type E. It can be summarised in Table1 (Where a Mindbugs is at least assigned to more than one type, and several types are separately acknowledged).

Table 1. The identification and classification of mindbugs

M	C	H	E
M1, Misinterpretation of Linguistic Adequacy of Natural Language , C1	C1, Misinterpretation of Linguistic Adequacy of Natural Language, M1	H1, Indistinguished Affinity to Unstructured Discussion, C6, E3	E1, Susceptibility to the Fad of the Month, H6
M2**,** Misinterpretation of Linguistic Adequacy of Object Languages, C2	C2, Misinterpretation of Linguistic Adequacy of Object Languages, M2	H2, Adversity to Budgeting for Interface Expenses, C7, E4	E2, Unawareness of Imputed Structure, H10
M3, Confusing Prestige with Authoritativeness	C3, Misconstruing Technology as Science (and vice versa), M4	H3, Affinity to All-Encompassing Dichotomies	E3, Indistinguished Affinity to Unstructured Discussion, C6,H1
M4, Misconstruing Technology as Science (and vice versa), C3	C4, Insensitivity to Conceptual Scale	H4, Leaping to Misassociation	E4, Adversity to Budgeting for Interface Expenses, C7, H2
M5, Misconstruing Structural Incompetence as Innate Incompetence	C5, Insensitivity to the Presence and Origins of Human Fallibility	H5, Insensitivity to Role Distinction	E5, Mistaken Sense of Uniqueness
M6, Misattribution of Consensus	C6, Indistinguished Affinity to Unstructured Discussion, H1,E3	H6, Susceptibility to the Fad of the Month, E1	E6, Mistaken Sense of Similarity
M7, Misconstruing Persistence as Validity	C7, Adversity to Budgeting for Interface Expenses, H2, E4	H7, Insensitivity to the Significance of Information Flow Rates	E7, Misconstruing Philosophy as Ideology(and vice versa)
		H8, Adversity to Deep Though	E8, Misassignment of Relative Saliency
		H9 Failure to Distinguish among Context, Context, and Process	E9, Irresponsible Propagation of Underconceptualized Themes
		H10, Unawareness of Imputed Structure, E2	E10, Unawareness of the Cumulative Impact of Many Collocated Mindbugs

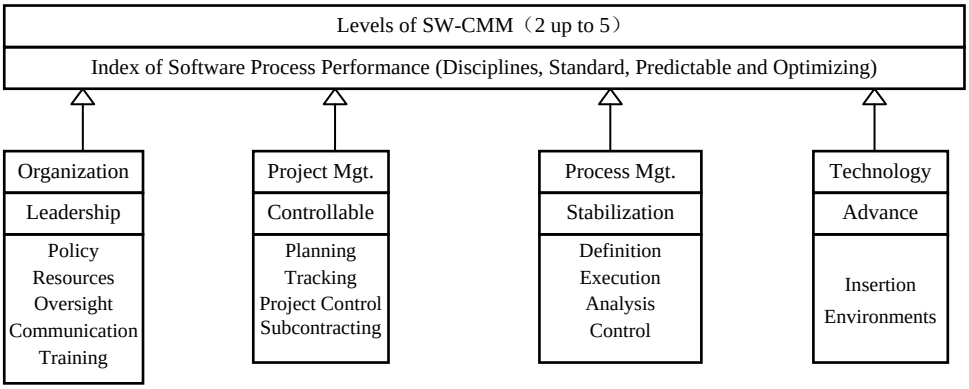

Figure 12. Class diagram of index system of software process performance

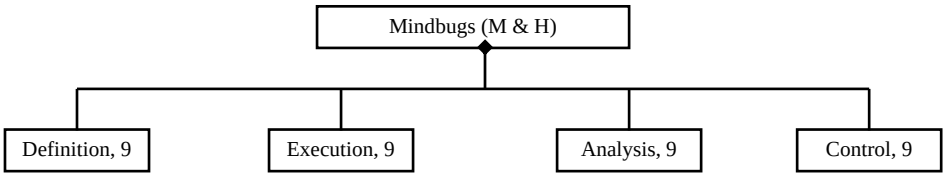

Figure 13. Class diagram of mindbugs of process management of SPI

Table 2. The reliability of mindbugs questionnaire

Mindbugs	P1	P2	P3	P4	P5	P6	P7	P8	P9
Reliability	0.8447	0.8816	0.8570	0.8701	0.8882	0.9190	0.9031	0.9185	0.8934

9.1.2 Framework of Software Capability Maturity Model

SPI actions are divided into 15 categories, which are further grouped into 4 major topics, namely organization, project management, process management and technology. They are summarised in Figure 12 (The four views are integrated into one figure to illustrate that the four views is a whole) [1].

9.1.3 Research Design

The five levels Likert scale is taken. The process management of SPI is selected referring to key areas of SW-CMM. The process management of SPI involves process definition, process execution, data gathering and analysis, and process control. The process definition provides a standardized framework for task implementation, evaluation, and improvement. Process execution defines the methods and techniques used to produce quality products. Analysis deals with the measurements, including software products and processes, and uses this data. Process control concerns the establishment of mechanisms to assure the performance of the defined process and process monitoring and adjustment where improvements are needed [1].

The major mindbugs of SPI are mindbugs of M and mindbugs of H through literature review and investigation on spot. The overlap mindbugs of M and H are removed. There are nine mindbugs in the following to be selected for study: Confusing Prestige with Authoritativeness (M3, P1), Misconstruing Structural Incompetence as Innate Incompetence (M5, P2), Misattribution of Consensus (M6, P3), Misconstruing Persistence as Validity (M7, P4), Leaping to Misassociation (H4, P5), Insensitivity to Role Distinction (H5, P6), Insensitivity to the Significance of Information Flow Rates (H7, P7), Adversity to Deep Though (H8, P8), and Failure to Distinguish among Context, Context, and Process (H9, P9). The first four mindbugs are in the type of minsinterpretation, and others of habit. The Pi (I=1, 9) is taken for easy documentation, It can be illustrated in Figure 13.

9.1.4 Statistic Analysis

The questionnaire survey was carried in Guangdong Province of China, including Center of BSS&OSS Support of China Telecom Corporation Ltd. Guangzhou Research Institute, Zhongshan Mobile Telecom Corporation, China Industry and Commercial Bank Guangzhou Branch, Asia Info et al. The number of the total questionnaire is 59, and there are 41 effective ones.

According to the above nine kinds of mindbugs, reliability analysis result is illustrated in Table 2. P6 is the top one (0.9190) and P1 is at the bottom (0.8447), but all

are bigger than 0.8. The internal consistency of questionnaire is very good.

Table 3 illustrates the following information:

a) The minimum of mindbugs are all 1, except P8 (1.25). The maximum of first five mindbugs are all 5. The other mindbugs are in the following respectively: the P6 is 4.00, and the P7 is 4.75, and the P8 is 4.00, and the P9 is 4.75.

b) The maximum mean is P3 (3.4634), and the minimum mean is P6 (2.7966), the average of the means is about 3. The descending order is as follows: P3 (3.4634), P4(3.2561), P2(3.2012) and P1(3.1890). Every type M of mindbugs is above the type H of mindugs.

One-sample T Test is taken for every hypothesis (There is some kind of mindbugs in process management of SPI). Test Value sets 3 (basing on descriptive statistics) and 95% confidence interval of the difference is taken, The T test is passed. The result is illustrated in Table 4.

a) The mean of the mindbugs of M is 3.2774, and the mean of the mindbugs of H is 2.9707.

b) In the process definition, the mean of mindbugs of M is 3.2256 and the mean of the mindbugs of H is 2.9707. In the process execution, the mean of the mindbugs of M is 3.2500 and the mean of the mindbugs of H is 2.9951. In the process analysis, the mean of the mindbugs of M is 3.3049 and the mean of the mindbugs of H is 2.9707. In the process control, the mean of the mindbugs of M is 3.3293 and the mean of the mindbugs of H is 2.9463. The biggest of mindbugs of M is the process control, and the following is process execution. The biggest of mindbugs of H is the process execution, and the following are both process definition and process analysis. The mindbugs of M is more serious than the Mindbugs of H. These are consistent with the results of both literature review and field study

It is necessary to research on SPI in the view of cognition. The research finds that the most common mindbugs of SPI is Misattribution of Consensus (M6), and the finding is consistent with literature review and investigation on spot. It is suggested to resolve the mindbugs in Type M, including M6 in the first place, then framebreak and remodel to resolve the mindbugs of Type H in SPI.

Table 5 illustrates the following information:

Table 3. The descriptive statistics of mindbugs

Mindbugs	N	Minimum	Maximum	Mean	Std. Deviation
Confusing Prestige with Authoritativeness (P1)	41	1.00	5.00	3.1890	.9300
Misconstruing Structural Incompetence as Innate Incompetence (P2)	41	1.00	5.00	3.2012	.9000
Misattribution of Consensus (P3)	41	1.00	5.00	3.4634	.8112
Misconstruing Persistence as Validity (P4)	41	1.00	5.00	3.2561	.7593
Leaping to Misassociation (P5)	41	1.00	5.00	3.0122	.9202
Insensitivity to Role Distinction (P6)	41	1.00	4.00	2.7866	.9962
Insensitivity to the Significance of Information Flow Rates (P7)	41	1.00	4.75	3.0427	.9697
Aversity to Deep Though (P8)	41	1.25	4.00	3.0000	.9066
Failure to Distinguish among Context, Context, and Process (P9)	41	1.00	4.25	3.0122	.8695

Table 4. One-sample T test

	Test Value = 3				95% Confidence	
	t	df	Sig. (2-tailed)	Mean Difference	Interval of the Difference	
Mindbugs					Lower	Upper
Confusing Prestige with Authoritativeness(P1)	1.301	40	.201	.1890	-.1045	.4826
Misconstruing Structural Incompetence as Innate Incompetence (P2)	1.432	40	.160	.2012	-8.2866E-02	.4853
Misattribution of Consensus (P3)	3.658	40	.001	.4634	.2074	.7195
Misconstruing Persistence as Validity (P4)	2.160	40	.037	.2561	1.644E-02	.4958
Leaping to Misassociation (P5)	.085	40	.933	1.220E-02	-.2782	.3026
Insensitivity to Role Distinction (P6)	-1.372	40	.178	-.2134	-.5278	.1010
Insensitivity to the Significance of Information Flow Rates (P7)	.282	40	.780	4.268E-02	-.2634	.3488
Aversity to Deep Though (P8)	.000	40	1.000	.0000	-.2861	.2861
Failure to Distinguish among Context, Context, and Process (P9)	.090	40	.929	1.220E-02	-.2623	.2867

Table 5. Compare minsinterpretation mindbugs with habit mindbugs

	N	Minimum	Maximum	Mean	Std. Deviation
Process Definition (M)	41	1.00	5.00	3.2256	.8020
Process Definition (H)	41	1.00	4.40	2.9707	.8241
Process Execution (M)	41	1.00	5.00	3.2500	.8422
Process Execution (H)	41	1.00	4.20	2.9951	.8781
Process Analysis (M)	41	1.00	5.00	3.3049	.7876
Process Analysis (H)	41	1.20	4.20	2.9707	.8301
Process Control (M)	41	1.00	5.00	3.3293	.8449
Process Control (H)	41	1.20	4.00	2.9463	.8025
Means (M)	41	1.00	5.00	3.2774	.7514
Means (H)	41	1.20	4.10	2.9707	.7943

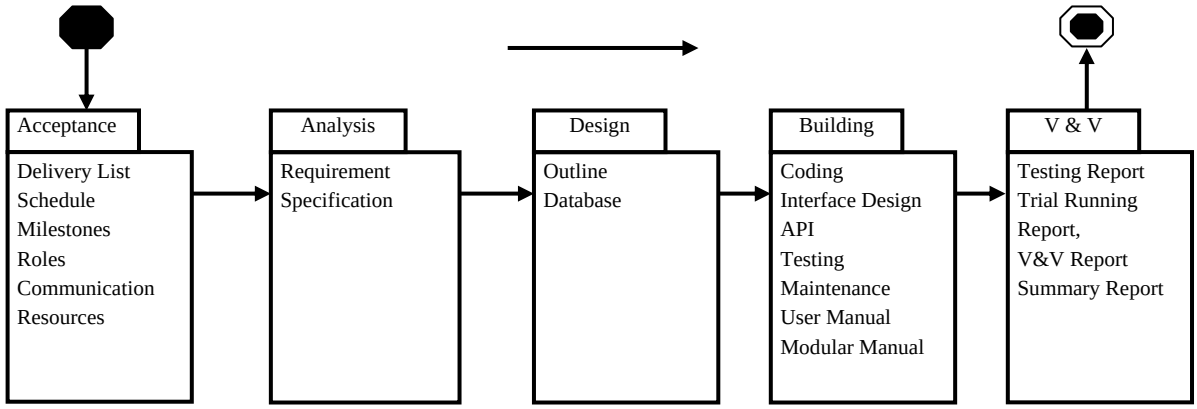

Figure 14. Software project process model of XXX corp

9.2 Case Study

XXX Corp. was established in 1995 and is a high-tech enterprise of Guangzhou. Its mindbugs in software processes, software project organization, and keys to software project management are analyzed. Finally, knowledge integration support structure of quality software production is put forward.

9.2.1 Primary Activities of Software Process

Software project usually is carried through five steps in the following: (1) Acceptance commission; (2) Analyzing requirement; (3) Design outline; (4) Building; (5) Verification and validation, It can be illustrated in Figure 14.

(1) Acceptance commission. Including pre-sales, investigation, requirement investigation, submitting requirement investigation report, feasibility study report, assignment software development tasks; (2) Analyzing requirement. Requirement specification, which is V&V to be effective, is submitted. It is both the baseline of software engineering management and the most importance gist for check and acceptance. (3) Design outline. Including outline design and database design. (4) Building. Detailed design based outline design according to coding specification, interface design specification and API specification etc. (5) Verification and Validation. Check and accept the system, including a series of testing, trial running and evaluation, submitting testing analysis report, trial running report, check and accept report and project development summary report.

9.2.2 Mindbugs in Software Process

Because of the programmers failing to rationally organizing the development team, there are problems in the following: (1) Working following feeling in the first place, working following the operator's scattered ideas in the end (no well-done requirement description, Type M); (2) Think hard, working while thinking, including writing documentation (no well-done project plan, Type C & H); (3) Think hard without guidance, it is often to discover that the system couldn't be integrated at last (no well-done process specification, Type H & E); (4) It often lose money in business when the project is finished (no well-done milestones, four types). The programmers often feel weariness, distress, at sea and no achievement (complexity, mindbugs). The phenomena are very com-

mon among software enterprises in China.

9.2.3 Software Project Organization

Developing organization scientifically is the foundation of highly efficient and stable project operating. In accordance with the CMM model, the company established SEPG (Software Engineering Process Group, similar to the Technical Group), SQA (Software Quality Assurance) and SEG (Software Engineering Group), and this led to form a balanced system including the legislation, supervision and law enforcement and realize the organization theme of the software process improvement performance model. It is illustrated in Figure 15. By 2002, the company had developed relatively complete and formal documents to manage software processes, and set up a knowledge management center.

9.2.3.1 Technical Group

In the software development processes, SEG has to obtain large amount of technical resources which relies on the day-to-day accumulation and constantly bringing in external technical resources. At the same time, by establishing development standards and norms, and completing technical proposal verification and technical personnel evaluation as well as the evaluation of researches which do not belong to any R&D development team, the development team's working efficiency can be assured and duplication of enterprises investment can be avoided. Consequently, the technical resources can be fully used. It can be illustrated in Figure 16.

The characteristics of the technical group include: 1) involving the functions of SEPG; 2) never directly undertaking the entire project tasks; 3) mastering the core technology, managing the technical capability of enterprises, and achieving processes management and technology introduction.

The functions of the technical group include: 1) technical standards: drawing up technical standards and specifications to form a unified internal technical language which is conducive to initial learning, technology communication and mutual supports among the teams. 2) technology accumulation: drafting technical plans, ensuring there are plans to introduce new technologies and organizational research and development so as to realize technology sharing as well as assessment and enhancement of the technological capability.3) knowledge resuing: managing the achievement of technology and knowledge base to ensure maximum reuse of knowledge and shortening the development cycle. 4) the quality of personnel: organizing technology training, technology communication and technology assessment to speed up the improvement of the quality of personnel. 5) technology quality: organizing technical evaluation and proposal verification to ensure the quality of technology.

The positions of the technical group include: 1) Technical manager is the person who is in charge of technology planning, technology assessment and the development of technology management system. 2) System analyst is responsible for technical analysis, program review and project assessment. 3) R & D engineer is responsible for the development of public components. 4) File Manager manages product base and shares modules base, knowledge base and database.

Many companies are not sizeable enough to establish the group, but these duties are so important that they must be managed by somebody who can be substituted by virtual teams composed of the department manager, file managers, some technical experts. However, the results are often not so satisfactory.

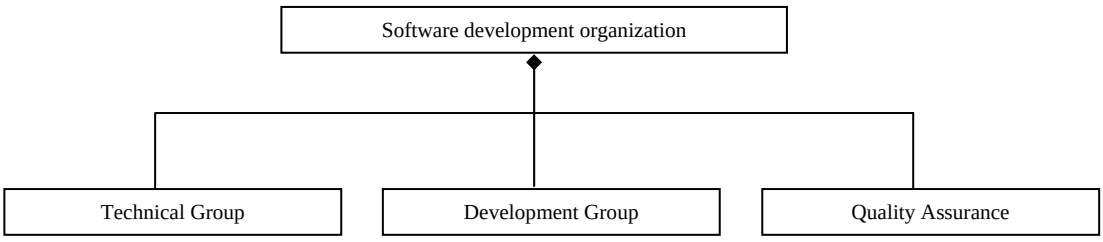

Figure 15. Class diagram of software development of XXX corp

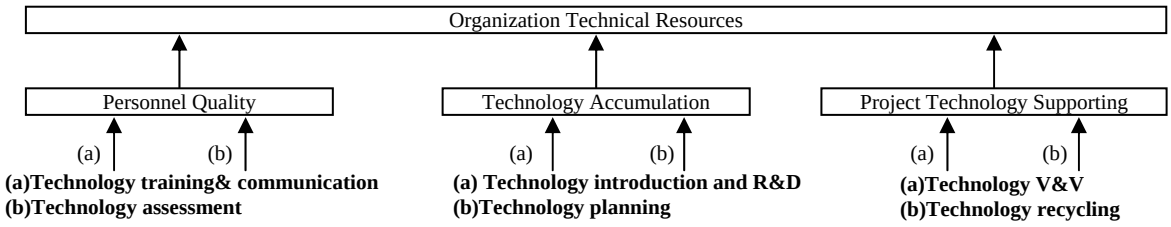

Figure 16. Structure diagram of software technology group of XXX corp

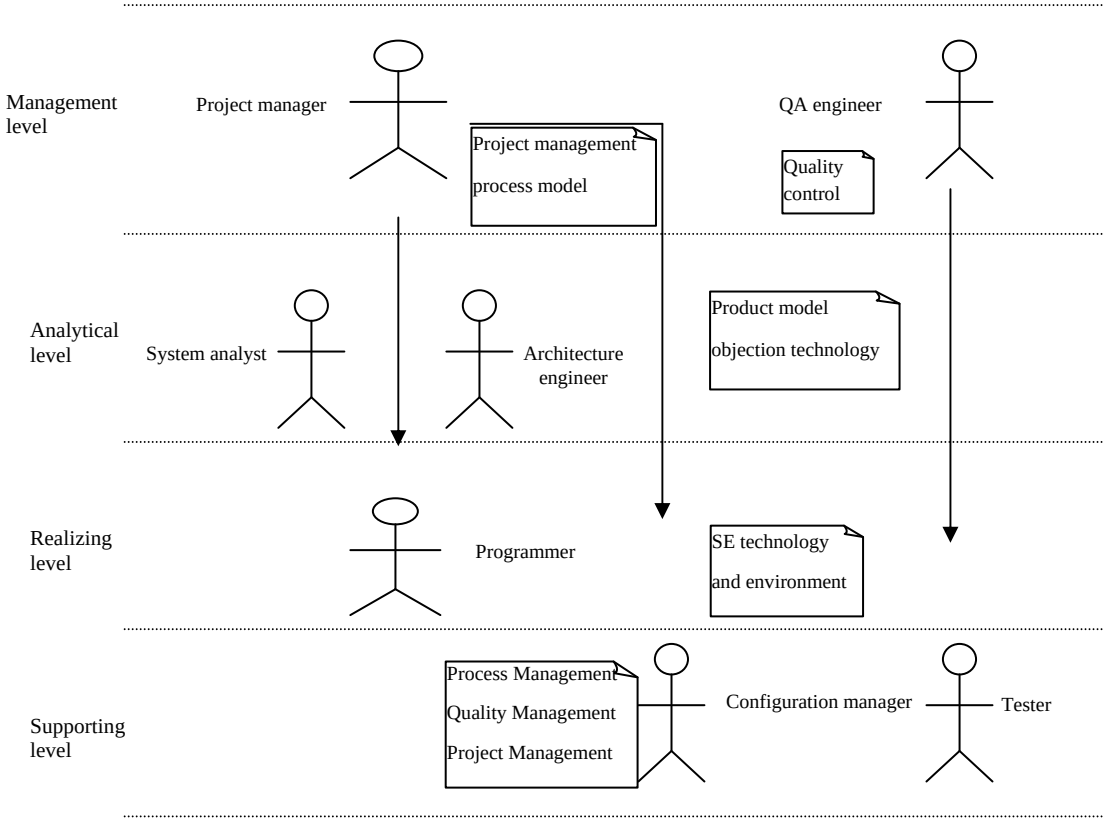

Figure 17. Software project organization of XXX corp

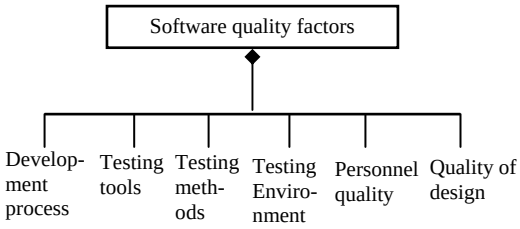

Figure 18. Software quality factors of XXX corp

9.2.3.2 Development Team

Development team can be further divided into a number of different conventional technology groups, such as network technology, Java technology, VB technology and PB technology. When the specific project is to be carried out, the project manager arranges temporary project teams according to the technical requirements. The organization of the project team is very important, so it should be decided by specifically analyzing the scale and technical characteristics of the project. The project organization is shown in Figure 17 (using the object-oriented technology, emphasizing on software system architecture, integrating the process model, product model and the object model, and describing that SPI needs project management knowledge, quality management expertise and software engineering knowledge and technology). If necessary, a project director can be set up to be responsible for total project control, cross-organizational coordination and policy management in order to achieve the project management and process management themes of the software process performance model.

9.2.3.3 Quality Assurance Group

The specific objectives of quality vary from different projects. It should be balanced between the investment in quality assurance and quality losses. Furthermore, it is important to keep the concept of quality and cost and customer satisfaction in mind so as to conduct the quality assurance in proper direction and eliminate the problems in the early phase. It is necessary to distinguish between products and customized projects in the quality requirements, and in the customized projects, the quality assurance can be agreed to be achieved by the cooperation with clients. Quality assurance activities include the quality planning, test plan, test analysis reporting, testing activities, proposal verification and process assessment and so on.

Many factors affect the quality, mainly in terms of the quality of development process, testing tools, testing methods, the quality of testing personnel, testing environment and quality of design (Figure 18). Through an

organic integration of processes, technology and personnel to achieve the process management theme of the software process performance model.

9.2.4 Keys to Software Project Management

9.2.4.1 Make Full Use of Available Resources

Development team is only one of the development executive team, the development standards and protocols, the power of the quality test, the accumulation reusable resources of enterprise (components, technology, methods, document templates, analogous case, business experts and the experience of members, etc.), are off-the-shelf resources that the development team can use, the project managers must make better use of these resources, especially good at excavating hidden resources, so as to shorten the development cycle.

9.2.4.2 Strengthen the Planning

Some people think that "program can't keep up with changes", so they hold a negative attitude to plans. But as long as we emphasis on planning and strengthen the management of a variety of changes, we can get two major benefits:

1) To reduce uncertainties. When people prepare plans, it is usually difficult to plan some problems. In fact, the uncertainties are on the exposed, we only need to assess the potential consequences of these uncertainties, and then formulate corresponding contingency measures. When people form a new version of each plans, the earlier uncertainties will be less. The evolution versions of plans are in fact clear step by step processes to the target.

2) Help to improve efficiency. Plans enhance a sense of urgency to reduce the waiting time for each other. Project team members are in a project, the plans make members all identify with the target, the process of formulating plans are actually the process of members' understanding, also is to clarify their own roles, responsibilities and the process of tasks. It is conducive to get members' psychological together, do what they should do, thereby enhancing the efficiency of the entire team.

9.2.4.3 Control Project Changes

We must control the project's direction of change toward favorable conditions, prevent adverse changes and eliminate a vicious circle. According to the control stage, control changes can be divided into three types:

1) Prior to control. The crucial issues must be ascertained in order to avoid opening the Pandora's Box. For example, the validity and the requirement-bound of the project contract documents must be confirmed, and the relevant documents should be signed in time in order to prevent a lot of projects change from time to time later because of people or time.

2) Control when it is in progress. For a variety of things appearing in reasonable changes, to format a legitimate change records after the deliberations of the relevant aspects, and have the same effect as the original legal documents. These records will serve as the basis for other related changes and later verification.

3) After control. In the circumstances that issues have emerged, we should properly handle them to prevent further deterioration of the problem, from big to small, small to little. At the conclusion of the project, projects changes arising in the course must be analyzed to guide future project planning and project control. Let members learn from the changes and improve the team's ability to cope with the changes.

9.2.4.4 Strengthen Requirement Management

Large applications need business experts, technical experts and other key personnel, to spend a great deal of energy to find it out, so it absolutely can't be ignored. The failure of most projects is not to find clear requirement. The purpose of requirement analysis is to ascertain needs, recognize the requirement and define boundaries, as well as to reduce rework due to requirement uncertainty. The following are requirement access steps:

1) Select staff. It is important for the entire quality assurance to choose the appropriate staff. Research and analysis staff needs to be better with rich experience and professional knowledge. Because the customer's requirement is often not clear, it is necessary to deal with requirement as a separate item as much as possible.

2) Research. Due to business customers often do not know how to tie in with the software development research, research methodology is very critical. We must grasp the whole, understand the details and then take the line of top-down. From the horizontal view, staff must research in accordance with the business processes from downstream to upstream backward (output and then import). It would be better to access needs from variety ways, such as form, face-to-face communication etc.

3) Writing. Write the requirement specifications.

4) Confirm. Recognizing the requirement, the two sides sign and generate an effective basis.

5) Change management. When management needs change, people must fill the requirement change list. It must be confirmed and then formatted the requirement tracking specifications.

9.2.4.5 Communication Management

Communication targets: project internal members, supervisors, customers, suppliers and supervisory company and so on. The regular medium of internal communications often is report forms and conference.

Communication forms: e-mail, fax, telephone and face-to-face and so on.

Five kinds of meeting: the project assessment conference, the demand review conference, the design review conference, and the product release conference.

To achieve the desired effect of communication, use e-mail or telephone and other forms in order to improve

the efficiency of communications as much as possible. For common resolutions and the content that must be addressed as an important basis of the communication, it is necessary to use fax and other written forms in order to improve the quality of communication. Written form communication is also a good way to reduce the communication misunderstanding, while oral communication is the most unreliable, which is mainly used to discuss, and its results ought to be recorded. The whole project written formal documents must be unified managed and archived collectively, having unified entrance and exit. Projects communications also need the right level and confidentiality rules.

9.2.4.6 Document Strategy

In accordance with the classification of software engineering methods, the development document can be divided into feasibility studies and planning category, requirements analysis category, outline design category, detailed design category, construction category, assembly and testing category, validation testing category, operation and maintenance category and so on. If there are too many documents, most people just cope with them in haste and pay no attention to the quality of documentation. Because a wide range of documents will increase the project cost, therefore, how to improve the efficiency to write documents is very important. The ways to enhance the management of documents are in the following:

Normative. We should establish a unified document language specification, edit, audit, grant, release, change rules. Build a complete documents list so that project staff can easily locate the right one.

Learn easily. We should establish document written guidelines and allow developers to share the document template, which can enhance the document's learn easily.

Readability. We should use charts, from the coarse-to-fine, surface to inner. Improve the document granularity. Facilitate supervision, accreditation and quality assurance personnel to participate. Write the document in accordance with the document and the reader (the development, use, training, implementation, sales, service and testing services), occupational characteristics and professional characteristics to reduce misunderstanding and the hardship to be understood.

Portability. We should pay attention to the document structure to facilitate the transplantation to other projects.

Tool support. We should use documentation tools (such as: UML tools, etc.) to automatically generate documentation.

Continuous improvement. We should provide the staff who writes documents with the continuous improving methods of document written.

9.2.4.7 Knowledge Management

The goal of knowledge management: Knowledge is the most valuable wealth in software companies. The stock, flow and network of knowledge reflect the accumulation and circulation of knowledge. Knowledge management is to collect the existing knowledge and skills in enterprise, and then to send them where it is needed and help enterprises maximize the benefits. The goal is to transfer the most appropriate knowledge in the most appropriate time to those most in need and help them make decision-making.

Table 6. The content of knowledge management

Classify	Item Details
Interior knowledge	1. The technical standards and management standards 2. Case-base 3. Questions-base 4. Knowledge-base 5.The public library module 6.The technical exchange activities 7.The registration of intellectual property rights (copyright registration)
Exterior knowledge	1. Establish the feedback information base to provide the product development requirements for new technologies 2. The supplier information database: For example, training institutions, the servers, match software, etc. 3. Competitor information base 4. The expert knowledge base (explain how to obtain the means of knowledge if it doesn't have electronic document) 5. Outsourced staff expertise 6. The latest technology terms 7. The relevant knowledge Web address
Staff knowledge creation	1. Staff expertise library, that can be easily convened to discuss technical problems 2. Staff recommends library 3. Troubleshooting

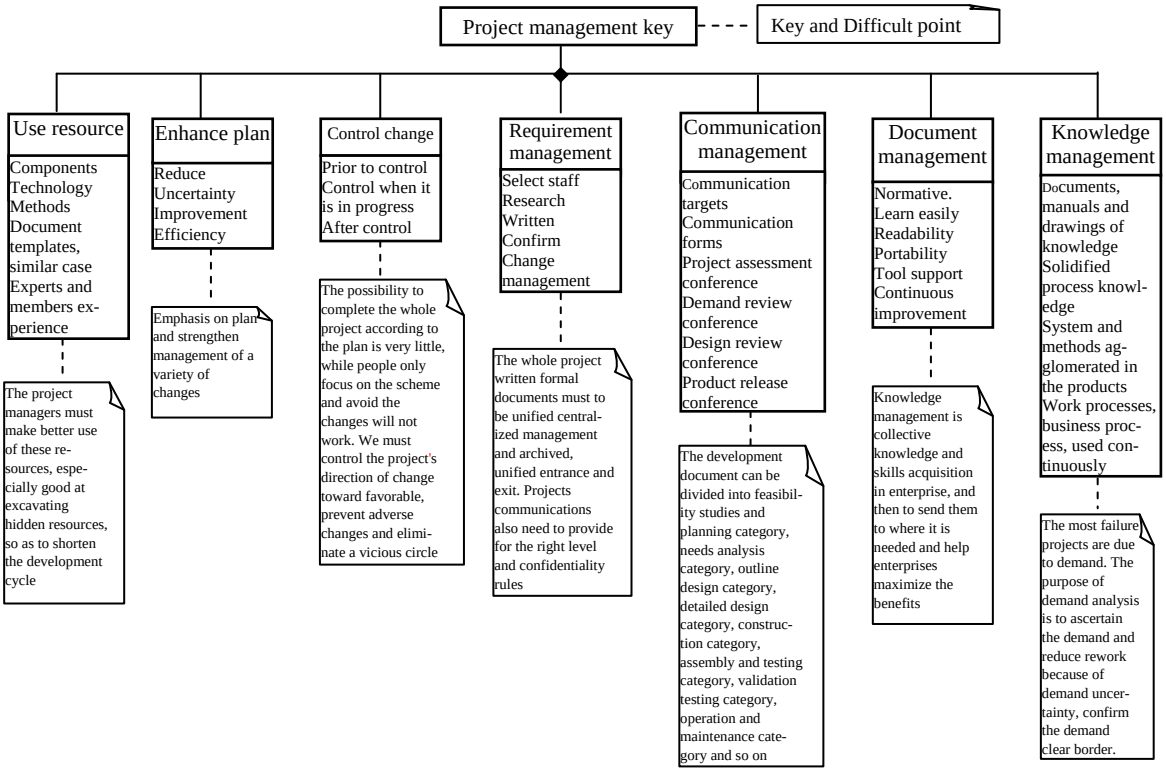

Figure 19. Class diagram of keys to software project management of XXX corp

The existing forms of knowledge: archives (documents, manuals and drawings of knowledge), the tacit knowledge in human brain, solidified process knowledge (system and methods agglomerated in the products, work processes, business processes, which are used continuously).

Knowledge management tasks: To build the knowledge store, improve the knowledge acquisition methods and the knowledge environment, manage knowledge assets. Set up knowledge management positions to collect and organize knowledge. Establish knowledge networks to provide the access for the project team at any time. Brainstorming, take full advantage of the tacit knowledge in organizations.

We should pay attention to sum up the project, analyze and extract useful knowledge archive. Table 6 is the classification and illustration of knowledge management that contribute to knowledge management. Figure 19 shows the key issues and points to implement software project management.

9.2.5 Knowledge Integration Support Structure of Quality Software Production

9.2.5.1 Japanese Knowledge Science

Ikurjiro Nonaka argued that knowledge is created and used in organization through knowledge transforming process, including four patterns: socialization, externalization, combination, and internalization. All four of these patterns exist in dynamic interaction, a kind of spiral of knowledge. Knowledge is truly created and used effectively and depending on established dynamic business system [22–23]. Nine questions are put forward to the following [23]: 1) What is the status of “truth” in the definition of knowledge? 2) Do tacit and explicit knowledge fall along a continuum? 3) Is the tacit/explicit knowledge distinction along the continuum valuable for organization science? 4) What is the conceptual basis of knowledge conversion? 5) Given the relationship between tacit knowledge and social practices, how can the concept of knowledge conversion be upheld? 6) What is the outcome of knowledge conversion? 7) What is the relationship between organizational knowledge creation and social practices in organizations? 8) When and why do social practices contribute to the conservation of existing tacit knowledge and existing routine rather than organizational knowledge creation and innovation? 9) How can leadership motivate and enable individuals to contribute to organizational knowledge creation by transcending social practices? College of Know- ledge Sciences was established in Japan Advanced Institute of Science and Technology (JAIST) to investigate and study how to establish and develop knowledge science in the world [24].

9.2.5.2 Chinese Knowledge Engineering

Prof. Wang Zhongtuo defined the knowledge technology as the following: knowledge technology is the discipline about the methodologies, methods, procedures and tools of knowledge identification, acquisition, storage, processing, dissemination, and creation for the purpose of adding value to human economic and social activities. The object of knowledge technology is not only the explicit knowledge, but also tacit knowledge and their interrelations. The approaches are not only technique-oriented, but also human-behavior-oriented. The knowledge management involves primarily the knowledge processing cycle that includes the capture, analysis and communication of knowledge within organization. It also involves problems of searching and finding useful information. But the most important thing is the creation of new knowledge. He suggested that knowledge system engineering is the discipline of organization and management of knowledge systems. The architecture of knowledge systems, operation process of knowledge systems, engineering project development, systems intuition and knowledge fusion are under studying. Research center of knowledge science and technology was established in Dalian University of Technology (DUT) in 2000 [24].

9.2.6 The Domain of Science Model and the Domain of Knowledge Model

9.2.6.1 The Domain of Science Model

Figure 20 illustrated the domain of science model by Prof. Warfield in 1986.The two blocks consist of the corpus of the science. The two blocks consisting of the methodology and applications make up what is called arean. The arean is, of course, where the action is, which is often identified with "action research": a kind of research basing on the view that there is no substitute for becoming oneself in the problem as it occurs in a particular situation. The corpus refers to a body of knowledge but not to application-specific knowledge. The domain of science model is initially divided to reflect four components or blocks. There are: foundations, theory, methodology, and applications. The first three of these make up the science. The model represents a relationship of steering in that the foundations steer the theory, the theory steers methodology, and methodology steers applications (and through the foundations the other blocks of the science) [4].

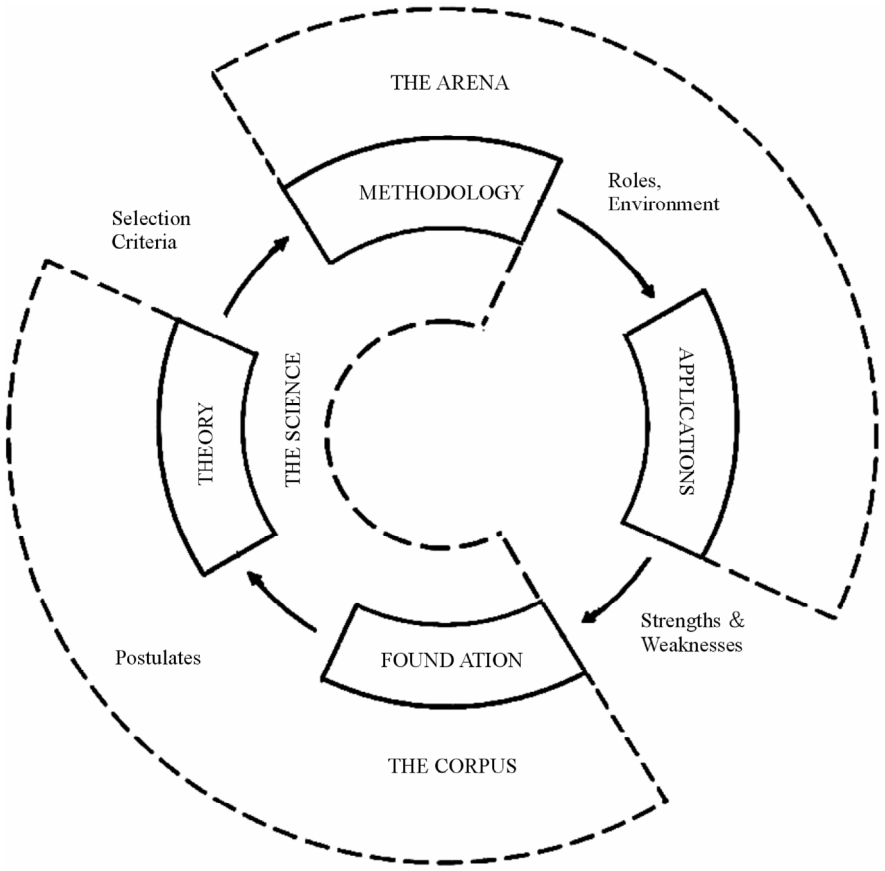

Figure 20. The domain of science model

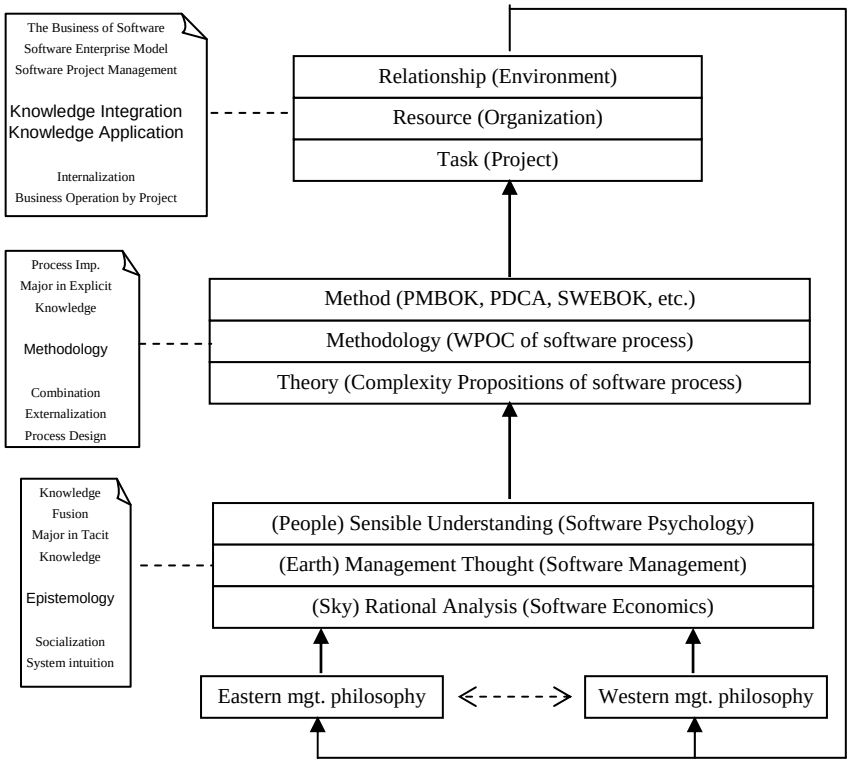

Figure 21. Knowledge integration support structure of quality software production

The foundations in the model have the specific function of providing the decision-making basis for the science. Whenever issues arise in the theory or the methodology, it must be possible to refer them to the foundations for resolution. If it is not possible to resolve an issue in this way, one must assume that the foundations are, in some way, lacking and must be upgraded. In order to fulfill its function, the foundations must incorporate the Universal Priors (the human being, language, reasoning through relationships, and archival representations) in a way that is directly relevant to the particular science being constructed. The foundation also must incorporate those particular concepts (the essences) that are required to distinguish the particular science from other sciences [4].

In order for the foundations to steer the theory, it is clear that the foundations must be prior to the theory, i.e., they must contain concepts and propositions that do not depend on still deeper ideas foreign to the science and they must not depend on the theory. The dependence must be from the foundations to theory. A certain parsimony is necessary; but matters must be excluded that are not so established. To fulfill the function of being a fount of resolution for issues arising in the Science, the Foundations must not too great in number. To the extent possible, ideas should move out of the foundations toward the theory and out of the theory toward the methodology, where greater volumes of information can be tolerated [4].

If the various sciences were clearly organized in terms of the three blocks of the Domain of Science Model, people who have to work across the sciences would find the task of drawing upon them much easier. And this would permit the integration of parts of the recognized scientific disciplines into newer integrative sciences [4].

9.2.6.2 The Domain of Knowledge Model

In our understanding, philosophy beliefs, which steer what type of theory being established by knowledge workers, must be involved. The “methodology” originally refers route in which one traces another and it evolves the principles and program to do work. Methods must divorce from methodologies and becomes new layer. In addition, the foundation of a given discipline can be merged with its relative theory layer. The domain of knowledge science is established referring the domain of science model, including philosophy belief, theory, methodology, methods and applications. Of course, the applications are also counteractive on the philosophy belief. Because the philosophy does not belong to the science category, it is referred to as the domain of knowledge model [10].

9.2.7 Knowledge Integration Support Structure of Quality Software Production

Software process is a collection of complex activities that have strict time sequences, some are currently asynchro-

nous, and some are condition each other and some interact with each other. In fact, the activities in software process have complex network relationship. Software development are highly dynamic processes, and the dynamic change can be found in all phases in software processes, such as requirement specification, assignment tasks, debugging, development policy, tools and support environment etc. These changes are often unpredictable and their influences often couldn't be identified because software process is people-oriented and software product is intangible itself, and this results in that the software processes are hard to control and theirs work quality are not easy to assess. Software process has great complexity. What philosophy beliefs are necessary for knowledge economy? How to design methodology of quality software production management? How to align quality software production management with software enterprise business objects to compete in both domestic market and overseas, which are rapidly changing?

Basing on both the domain of knowledge model and support structure of interactive management, knowledge integration support structure of quality software production is illustrated in Figure 21.

Figure 21 illustrates that managing the complexity of quality software production can be abstract through essence (e.g. the complexity propositions of knowledge management of software process etc.), and the abstract problems can be embodied through interactive cognition process (process design, e.g. work program of complexity of knowledge management of software process etc.), and the embodied problems to be structured through learning process continuously (process implementation, e.g. PMBOK, PDCA and SWEBOK etc.), and the structural problems to realize the business of software through business operation by project (e.g. knowledge integration model of software enterprise etc.).

The keys to knowledge integration support structure of quality software production may be knowledge fusion among Eastern management philosophy, Western management philosophy, sensible understanding (People, Ren-li and Software Psychology), management thought (Earth, Wu-li and Software Management) and rational analysis (Sky, Shi-li and Software Economics). In other words, tacit knowledge of software is socialized. At the same time, the business of software is also reaction on its support elements. “Sky-Earth-People” is the “Three-Cai” of Change of Science [25], which is origin of Chinese traditional philosophy, and “Shi-li Wu-li Ren-li” is put forward by Prof. Gu jifa and Prof. Zhu zhichang. “Software Psychology – Software Management – Software Economics” is adopted from software terms.

Prof. Cheng Zhongying argued that management always depends on a structure and evolves to an adaptable process [25]. In our understanding, quality software production depends on knowledge integration support structure which is illustrated in Figure 21, and evolves to design process with continuously transforming, creating and integrating process between explicit knowledge and tacit knowledge, and it's also an organizational learning process to resolve cognitive complexity continuously from system intuitions, and knowledge fusion, and process’s design and process's implementation, and business operation by project to the business of software. Because software industry is the core of knowledge economic, the knowledge integration support structure of quality software production is also generic resolution for managing complexity of knowledge systems.

10. Conclusions

Based on our understanding, there is a long way to go for knowledge integration support structure of quality software production in both theory and practice according to the domain of science model (Warfield), knowledge science (Ikujiro Nonaka) and knowledge technology (Wang Zhongtuo), but it will certain that the research is useful for both software production and knowledge economy in the future.

11. Acknowledgment

Thanks for the helpful discussion with Prof. Warfield, Prof. Wang Zhongtuo, Prof. Yang Jianmei, Mr. Li Jiangzhuang, Mr. Hou Yawen, Mr. Zhou Qiyang, Mr. Zhou Zhijun and my studtets Liu Qingjing, Wan Dan etc.

REFERENCES

[1] W. S. Humphrey, “Managing the software process,” Reading. MA: Addison-Wesley, pp. 19–24, 1989.

[2] J. D. Herbsleb and D. R. Goldenson, “A systematic survey of CMM experience and results,” in Proceedings 18th Intennational Conference on Software Engineering, Berlin, Germany, pp. 323–330, March 1996.

[3] R. S. Pressman, “Software engineering: A practitioner’s approach,” Vol. 5, McGraw-Hill Companies, Inc., pp. 19, 2001.

[4] J. N. Warfield, “A science of generic design: Managing complexity through systems design,” IOWA State University, pp. 142–146, 187–188, 1994.

[5] J. N. Warfield and A. R. Cardenas, “A handbook of interactive management,” AJAR Publishing Company, 1994.

[6] J. N. Warfield, “Twenty laws of complexity: Science applicable in organizations,” Systems Research and Behavioral Science, Vol. 16, No. 1, pp. 3–40, 1999.

[7] J. N. Warfield, “Understanding complexity: Thought and behavior,” AJAR Publishing Company, 2002.

[8] M. Scott and J. N. Warfield, “Enterprise integration of product development data: Systems science in action,”

Enterprise Information Systems, Vol. 1, No. 3, pp. 269–285, August 2007.

[9] Brooks and P. Frederick, "No silver bullet: Essence and accidents of software engineering," Computer, Vol. 20, pp. 10–19, April 1987.

[10] J. P. Wan and J. M. Yang, "Research on the work program of complexity of software process improvement—Methodology for implementation of SW-CMM," Science Press, Beijing, 2004. (in Chinese)

[11] J. P. Wan and J. M. Yang, "On the meanings of complexity, generic design science and work program of complexity," Journal of Systemic Dialectics, Vol. 10, No. 4, pp. 41–44, December 2002. (in Chinese)

[12] J. P. Wan and J. M. Yang, "Interaction management and its application," International Journal of Knowledge and Systems Science, Vol. 2, No. 1, pp.25–32, March 2005.

[13] J. P. Wan and J. M. Yang, "Knowledge management in software process improvement," Application Research of Computer, Vol. 19, No. 5, pp. 1–3, May 2002. (in Chinese)

[14] J. P. Wan and J. Z. Li, "Some considerations on knowledge management in software enterprise," Application Research of Computer, Vol. 20, No. 1, pp. 13–16, January 2003. (in Chinese)

[15] J. P. Wan and Y. L. Zhuo, "The e-business challenge," Application Research of Computer, Vol. 20, No. 9, pp. 9–10, September 2003. (in Chinese)

[16] J. Z. Li and J. P. Wan, "Considerations on project management in small and middle software organization," Application Research of Computer, Vol. 20, No. 9, pp. 14–17, September 2003. (in Chinese)

[17] J. P. Wan and J. M. Yang, "Research on the complexity of software process improvement," In Proceedings of 2003 International Conference on Management Science & Engineering, Moscow, USA, pp. 168–172, August 15-17, 2003.

[18] J. P. Wan, J. M. Yang, and H. Y. Han, "Support structure of knowledge management in software process improvement," In Information Systems: e-business Challenge, IFIP 17th World Computer Conference, Montreal, Canada, pp. 17–29, August 25-30, 2002.

[19] B. W. Boehm, "Seven basic principles of software engineering," Journal of Systems and Software, Vol. 3, pp. 3–24, March 1983.

[20] Z. Y. Zhou, "CMM in uncertain environments," Communication of the ACM, Vol. 46, No. 8, pp. 8–27, August 2004.

[21] J. N. Warfield, "Mentomology: The identification and classification of mindbugs," http://mars.gmu.edu:8080/dspace/bitstream/1920/3199/1/Warfield%20_20_20_A1b.pdf, 1995.

[22] Nonaka and Ikjurio, "Dynamic theory of organizational knowledge creation," Organization Science, Vol. 5, No. 1, pp. 14–36, 1994.

[23] Ikujiro Nonaka and Georg von Krogh, "Perspective—Tacit knowledge and knowledge conversion: Controversy and advancement in organizational knowledge creation theory," Organization Science, Vol. 20, No. 3, pp. 635–652, 2009.

[24] Z. T. Wang, Knowledge Systems Engineering, Beijing, Science Press, 2004. (in Chinese)

[25] Z. Y. Cheng and C. Theory, The Management Philosophy of China, Shanghai, XueLin Press, Vol. 315, 1999. (in Chinese)

