Knowledge-Driven Software Project Management Based on IDEAL Cycle

Tan Jialin
Senior Engineer, China Aerospace Science and Technology Corporation, Beijing, China
E-mail: 49613663@qq.com

Abstract. In recent years, aerospace products’ development has more needs for localization, integration and intellectualization, so how to improved efficiency and reliability of software development through optimizing the management modes has become the major challenge. This paper proposes a design idea of transforming the task-driven management model to a knowledge-driven management model based on the software process improvement model (IDEAL cycle). By combining the improvement process of IDEAL cycle with the knowledge-driven management activities, it achieves a knowledge-driven software project management model based on IDEAL cycle. The practice results show that, through the implementation of the program, the software team are promoted to achieve the acquisition, storage, sharing, exchange, application and innovation of knowledge while implementing software process improvement, the process capability and core competitiveness of the team are fully enhanced. The requirements of software process improvement and the knowledge-driven design are fully taken into account in program design, and its generalized workflow engine and platform function have great promotion value and applicable to other knowledge-based enterprises.

1. Introduction
The quality of aerospace products is strongly influenced by the the software products, and the quality of software products becomes more and more important in the aerospace system. The control and management of the software R&D process is an important way of ensuring the quality of software products.

Under the conditions of inadequate human resources, young teams, and extensive professional services, the traditional task-driven software development models have been gradually exposed to the following problems:

The disconnection between plan and reality, poor follow-up and supervision: At the initial stage of software development, the software project leader will formulate a software development plan, but there are frequent deviations between the actual progress and plan under the conditions such as the change of system requirements, loss of team members, and the Unpredictable technical difficulties, etc;

Requirements management problems: The problems of software requirements management are mainly reflected in frequent changes and complex sources of the requirements. Some requirements are delivered through formal requirement specifications, and some sporadic requirements are delivered through e-mail, telephone, and other means. Due to the certain arbitrariness in the delivery
of requirements and the developers’ commitments, the missing or misunderstanding of the requirements occur occasionally;

Lack of historical measurement data: Whether the project is successful or not mainly depends on the individual ability of the software developer. The project estimation and plan rely entirely on personal experience, subjectively made without historical knowledge and rules for reference, which results in inaccurate planning;

Unstable software quality: Due to the lack of organizational guidance in the software development process, and the dependency of software project development on personal experience, there are obvious differences among various software project groups; in addition, due to the lack of knowledge accumulation and summary, the quality varies in the development process of different software even if they are developed by the same group of people.

2. Knowledge-Driven Software Project Management Based on IDEAL Cycle

2.1. General Ideas

Facing the opportunity of market transformation, in order to improve software process management capabilities, combined with the system construction of GJB5000A, the process capacity continuous improvement model based on IDEAL cycle is put forward. At the same time, due to the large number of young people in the software R&D team with a strong innovation rather than the thinking set, it proposes the management concept of transforming the software development from task-driven to knowledge-driven, and builds the product development and management model for knowledge-driven software based on IDEAL cycle to fully improve software process capability and management efficiency.

2.2. Software Process Improvement Based on IDEAL Cycle

When SEI launched the CMM, an IDEAL method model for software process improvement was proposed. It consists of the I (Initiating) phase, the D (Diagnosing) phase, the E (Establishing) phase, the A (Acting) phase and L (Leveraging) phase. These five phases are made up of a series of steps that are necessary to implement the process improvement, demonstrating the phases, activities, and resources required to achieve the software process improvement.

![Figure 1. Software process improvement based on IDEAL cycle](image)

In order to further improve the software quality management level, the IDEAL model was introduced into the development and management process of aerospace software products, and the software process system construction and improvement was carried based on the IDEAL model. During the I (Initiating) phase, the D (Diagnosing) phase, the E (Establishing) phase, the A (Acting) phase and L (Leveraging) phase, process capability gap analysis was completed, process improvement goals and plans are identified, a series of process improvement activities were carried...
out, including process improvement pilots and promotions, and process improvements summary and enhancement. At the same time, through the continuous circulation of the five phases in the process of actual organization system construction, the organizational process capability was promoted towards an upward spiral trend.

The I phase

Under the background of the software profession with certain software engineering capability, the software process improvement will establish the normalization and repeatability of the project process. By taking the CMMI level 2 certification as the goal in the first stage, relevant professional consultation and training are organized, thus further eliminating the obstacles in the promotion of process improvement activities.

The D phase

The construction and implementation of the GJB5000A system itself is a course of process improvement. It is necessary to make consistent and standard adjustments to existing foundations instead of “overturning them and then rebuilding them”. Therefore, in order to improve the existing software development processes in a targeted way, it first analyzed and diagnosed the status of software development and then evaluated and summarized the problems including requirements management, plan tracking, configuration management, and risk management, and further made some targeted improvements.

The E phase

The system establishment focused on organically integrating the 5000A system with the existing software development system, and formulating a system document consistent with the standards and the software development status of the organization, that is, solving the localization problem of the system.

The A phase

On the basis of completing the system documentation, the pilot project was selected to carry out the trial operation. During the trial operation, the project team and the EPG members constantly studied the coordination between the system and the actual model development process as well as the operability of the process details, and strove to achieve the coordination of standards, systems, and actual development process requirements.

The L phase

Through the practical application of the system in the model software development process, continuous improvement of the development process system was achieved from multiple perspectives of system, project, process, and method, and the solidification of the process was completed.

2.3. Knowledge-Driven Design Solutions

The implementation of software process improvement provides a platform for the accumulation of organizational knowledge assets, and the accumulation and application of knowledge also provide “impetus” for the improvement of organizational software process capabilities. In order to effectively accumulate software expertise and process data, a software process management system is used as a platform to complete the establishment and maintenance of a knowledge asset library, including professional templates, reusable components, fault cases, measurement data, best practices, methods and specifications, operation instructions, work environment standards, etc. At the same time, based on the knowledge asset library, a knowledge management system is established, which involves knowledge acquisition, knowledge encoding, knowledge storage, sharing and exchange, knowledge innovation, and knowledge application. And thus standard program for knowledge-driven design is built.

- Knowledge acquisition
- Knowledge encoding
- Knowledge storage
- Knowledge sharing and exchange
- Knowledge innovation
- Knowledge application
Figure 2. Knowledge-Driven Design Solutions

Knowledge Acquisition

Knowledge management begins with the knowledge acquisition, and knowledge acquisition is the premise of knowledge management. Knowledge acquisition channels include the following ways:

1) Acquiring knowledge through employee training or technical communication;
2) Acquiring knowledge through measurement and analysis activities in software process improvement;
3) Acquiring knowledge through drawing inferences from other cases and accumulating experiences during the software project development process;
4) Acquiring knowledge through text data, such as various types of books, professional journals, etc.;
5) Acquiring knowledge through the use of the Internet, including the acquisition of new technologies.

Knowledge Encoding

The knowledge obtained from knowledge acquisition is disordered and requires processing of the acquired knowledge, that is, encoding the knowledge to transform the disordered knowledge into ordered knowledge. The encoding of knowledge is a series of processing of identifying, classifying, extracting, organizing, and standardizing the knowledge after the acquisition, to make the knowledge easy to open, share and exchange, and be delivered by means of information.

Knowledge Storage and Sharing

The accumulation and storage of knowledge is the foundation of knowledge management. In order to increase the accumulation of knowledge and facilitate the sharing of results, the software process data, fault cases and other professional related knowledge are managed, respectively based on the software process management platform, fault alerting platform and portal platform. Thereby, the software development process is built as a process of knowledge accumulation and sharing.

Knowledge Innovation and Application

Knowledge innovation and application are the goals of knowledge management. By knowledge accumulation, it provides a basis for software process and design decision. On the other hand, when a software project encounters a new problem, the explicit, implicit or analytical knowledge mastered by the team can be applied to the practice to solve the problem and thus to provide support for the long-term professional development.

2.4. Knowledge-Driven Software Project Management Based on IDEAL Cycle

In order to improve the software management level, a knowledge-driven software management model based on the IDEAL cycle is proposed. By following the main line of improving the software process based on the IDEAL cycle, it strengthens the contribution to the organizational asset library throughout the entire life cycle of the software development, and encourages design innovation based on data accumulation, and promotes professional process capabilities and technical capabilities. The specific program model is as follows:
Figure 3. Knowledge-Driven Software Project Management Based on IDEAL Cycle

This model mainly consists of four parts, the software process improvement based on IDEAL cycle, knowledge processing and control, technical support and knowledge application feedback.

The software process improvement process based on IDEAL cycle

Most of the knowledge resources come from the software engineering process, so this part is foundation of the entire knowledge-driven design model. In order to fully explore knowledge resources, it completes the accumulation of process data through the automatic process platform. On the other hand, it strengthens the sharing of achievement, mobilizes employees’ enthusiasm for knowledge acquisition, and carries out the development of the fault alerting platform and software process management platform to provide long-term support for knowledge accumulation. In addition, regular quality learning and communication mechanisms are set up to encourage employees to promote an open exchange of ideas and information.

Knowledge processing and control

Knowledge processing and control is the core of the entire model, including knowledge acquisition, classification, storage, sharing and communication, and knowledge innovation. The knowledge processing and control is also the process of maintaining and applying the organizational knowledge base. The knowledge base is divided into three levels: software process data, software professional data, and typical fault cases. Employees can complete the knowledge storage and sharing through an electronic platform.

Technical support system

Technical support system, including tools supporting the operation of models, provides strong support for the implementation of knowledge-driven design program. According to the software process data, software professional data and typical fault cases in the knowledge base, the software process management platform, portal platform and fault alerting platform are used to complete the data management.

Software process management platform achieves the automatic acquisition of software process data, and provides data measurement and analysis platform, involving scale, schedule, workload, requirement stability, problems and other software process data.

Portal platform provides knowledge platform for software professional data accumulation, supports the storage and sharing of software professional standard specifications, quality data, design experience, reusable components and other knowledge.

Fault alerting platform achieves a database system to batch collect various fault cases and provide domain experts with a working window for refining design criteria, contributing collective wisdom to the software design process.

Knowledge Application and Feedback
Knowledge application and feedback is not only the driving force of the professional development, but also the foothold of knowledge-driven design. The application feedback of knowledge is mainly manifested in the following four aspects:

1) Decision supporting: It is mainly reflected in project estimation process based on organizational process data, improving the accuracy of estimation and ensuring the feasibility of project plans. On the other hand, it provides the basis for determining the deviation threshold in the process of project monitoring and provides support for the improvement of organizational process capability.

2) Reuse supporting: Through the accumulation of general components of software products, software product design is achieved, not only improving work efficiency while guaranteeing the quality of software products, but also providing basis for the establishment of a standard software product development process.

3) New product development: Through the accumulation and exchange of new technologies and cutting-edge technologies, it provides technical support for the development of new products.

4) Reliability design: Relying on the fault alerting platform, it effectively inherits the collective design experience, avoids repeated mistakes, and ensures the reliability of software design through the extraction of design criteria.

2.5. Guarantee Mechanism

In order to ensure the effective implementation of the knowledge-driven management program, the organization implements the guarantee mechanism from the three perspectives of human resources, working systems, and technology guarantee to ensure that the program is implemented and the effect is maximized.

Human Resources Guarantee

In order to enhance employees’ awareness of knowledge sharing, employees are relied to actively carry out knowledge acquisition, knowledge sharing, knowledge innovation and other activities. On the one hand, the department actively organizes various knowledge sharing activities to create a working atmosphere of sharing and mutual assistance. On the other hand, it emphasizes knowledge-based value orientation and encourages everyone to transform into a knowledge-based employee in order to develop a sustainable competitive advantage for the department.

System Guarantee

In order to achieve a more effective incentive mechanism, a knowledge performance mechanism is established, evaluate the employee’s knowledge achievements and make the value of each employee embodied in performance.

Technical Guarantee

The information platform is implementation carrier of the development and management program for the knowledge-driven software product based on IDEAL cycle, combined with the knowledge base construction program, completes the construction of the software process management platform by using a certain type of research guarantee conditions. On the other hand, based on the implementation of the tool requirements analysis work, the department deploys the informatization software team to complete the design of the portal platform and the fault alerting platform, providing guarantee for the smooth implementation of the management program.

3. Implementation Effect

After more than one year’s research and practice, the software knowledge base has achieved large amount of accumulation, and the software process capability has been effectively improved. The following is a brief description of the implementation results of the program by taking a “Test Bench Software” project as an example.

3.1. Project Planning and Monitoring
On the basis of project planning based on historical data, it tracks the scale, workload, progress, risks, and resources by using three monitoring methods, namely, double week monitoring, phase summary, and project milestone review. No severe disconnection between actual progress and the plan occurred during the project implementation, ensuring the software development process is under control.

![Figure 4. Project Plan Deviation](image)

### 3.2. Requirements Management

The project carried out the maintenance of the requirements traceability matrix at all stages of software development to ensure the integrity and consistency of the software requirements. During the development of the project, the first party proposed a requirement change through the revision of the task book. The project team carried out QA inspection and three-level review and verification for the changed software product on the basis of conducting domain of influence analysis of software change and completing software changes. And it was included into a controlled library to ensure the consistency of requirement changes. There was no missing or deviation in understanding throughout the entire life cycle of the software project.

### 3.3. Measurement and Analysis

According to the project planning, the project carried out measurement and analysis of several measurement items such as phase progress, milestone deviations, workload, requirement stability, and product defects. Measurement items and data acquisition timings are shown in the table below.

| No. | Measurement Target                     | Measurement Item       | Data acquisition timing |
|-----|----------------------------------------|------------------------|-------------------------|
| 1   | Delivery of products on schedule       | Phase Progress         | End of Phase            |
|     |                                        | Milestone Deviation    | Milestone point         |
| 2   | Improvement of project planning capabilities | Workload             | Every day               |
| 3   | Controllable Project scope             | Requirement stability  | End of Phase            |
| 4   | Enhancement of product quality         | Product Defects        | End of Phase            |
By implementing measurement and analysis in the software R&D process, it effectively monitors the implementation of the software quality system, timely solves the problems and adjusts the plans to ensure that the project process is effectively managed and controlled. On the other hand, it has accumulated a lot of historical data for the organizational asset library, providing a reference for the subsequent software development process.

3.4. Software Quality Control

All key process areas and work products in the software process activities of the software project were reviewed according to the development plan. Thirty six problems were discovered during the entire life cycle. The problem discovery trends are shown in the figure below:

![Problem Discovery Trends](image)

Figure 5. Problem Discovery Trends

Through analysis, in the phase of planning and requirement analysis and design, there was a repeated occurrence of requirement tracing problems. Afterwards, through modification and introduction of tools, problems of the same type were avoided. Except the obvious increase in the number of problems caused by software design flaws in the test phase, the quality trend throughout the entire software life cycle was stable.

4. Conclusion

The innovation achievements have been made as follows:

- The task-driven software development and management model has been transformed to a knowledge-driven management model, and the organizational process capabilities have been improved.
- By combining the process improvement model based on the IDEAL cycle with the knowledge-driven model, new ideas on software product development and management model have been explored, the process capability and the core competitiveness of the organization were fully improved.