Invited Paper

Current Status of Software Engineering in Japan

YUTAKA OHNO*

This paper describes trends in software engineering research during the last two or three years in Japan, based on a review of the last twenty years of international research on software engineering. It describes several leading research and development projects in the software engineering domain in Japan during the years 1987–1990. Based on papers and reports in issues of the Journal of the Information Processing Society of Japan (IPSJ) and other periodicals, this paper surveys some of the most interesting topics of research. Finally, it presents the author's outlook for software engineering research in the near future in Japan.

1. Introduction

Software Engineering has been the subject of intensive research for more than 20 years. During this period, requirements of software have gradually changed in accordance with the progress in computer and communication technologies and the trend toward an information-oriented society. However, the problems of software development, software quality (including reliability), and software management have not yet been solved, and the shortage of software engineers is also a serious social problem. Furthermore, it is said that research on software engineering has made less progress and seems to have been stagnant over the last few years. After considering the background, this paper overviews the direction of software engineering research and development during the last few years in Japan.

2. Twenty Years of Software Engineering Worldwide

During the 1970's, the advances made in computer technology caused a movement toward an information-oriented society, and to increased planning and development of large-scale computer systems in many organizations and industries. This period was referred to as one in which there was a trend toward an information-oriented society promoted by experts. The information industry, including software and network technologies, also developed rapidly during this time.

Many large-scale software systems were developed in this period, but the problem of software reliability and delays in delivery brought about a "software crisis." These circumstances led to the introduction of software engineering, whose main topics in the early 1970's were reliability and software productivity.

At the same time, programming methodology based on the idea of structured programming was advocated and became the basic concept of software engineering.

The first International Conference on Software Engineering was held in 1975, and subsequent conferences have contributed much to the development of software engineering.

Participants in the early conferences advocated the waterfall model of the software life cycle, and recognized the necessity of scientific methodologies for software development and project management. Many technologies for defining requirements were developed, and methods of improving quality and estimating development costs were studied.

In the 1980's the rapid progress of semiconductor technology led to the widespread use of personal computers and workstations with high performance and low cost. The trend toward an information-oriented society and the advance of network technologies spread to private and family life. Along with this rapid movement toward an information-oriented society, there was an increased demand for software, with higher requirements for function level and quality.

The trend in computer systems has been a gradual shift away from centralized processing systems to distributed processing systems consisting of workstations and personal computers linked by networks.

The same tendency is apparent in the development environments of software, and software development on workstation networks has gradually become the mainstream. During the 1980's much research was done on aspects of software engineering such as reuse of software parts, development support based on knowledge engineering, CASE tools based on data-flow diagrams, object-oriented development methodology, fourth-generation languages, multi-paradigm languages, automatic program generation, process programming, hypertext-like repositories, human interface, including window technologies, and integrated development environments. Many of these research topics have resulted in the development of tools that can be used in practice and incorporated into actual development en-

*Professor Emeritus, Kyoto University, Professor, Ritsumeikan University, President, Advanced Software Technology & Mechatronics of Kyoto.

vironments.

3. 1987~1990 Software Engineering Research Activities in Japan

Research activities related to software engineering in Japan in the late 1980's were jointly conducted by universities, research institutes, and industries. For example, a high-priority research project by major universities is being supported by scientific research funds provided by the Ministry of Education; a cooperative research project on software and distributed environments with twelve universities is being supported by IBM-Japan Corporation; the Software Designers' Associates (SDA) Project was supported by a consortium for cooperation between universities and industries, in which thirteen companies participated; and the Σ-system development project was supported by the Ministry of International Trade and Industry. There have also been vigorous activities aimed at the standardization of software engineering, in line with international activities in this field.

Descriptions of the above projects have been presented in technical reports and journals of the Special Interest Group on Software Engineering (SIGSE) in the Information Processing Society of Japan, and partial descriptions have been given at the International Conference on Software Engineering and related international conferences and workshops. In this section, we will briefly introduce the main research and development projects.

• "Research project on constructing software with high-level functions and quality (project leader, E. Wada, University of Tokyo)", a high-priority research project supported by science research funds from the Ministry of Education.

Applications for funding of this project had been made to the Ministry of Education for several years before it was accepted in 1989 and begun in 1990. Researchers from 13 universities will be engaged in research on the basic theory of software constructs, modeling methodology, quality evaluation, construction processes and distributed development environments for a period of three years. This project has just started and has not yet resulted in any products.

• "Cooperative Research Project on Software and Distributed Environments (project leader, Y. Ohno, Kyoto University)" supported by IBM Japan.

This was a three-year project, which was conducted by researchers from twelve universities and finished in 1990. This project, in which IBM's RTPC was used as a common workstation, involved several kinds of research activity, such as research on software construction using various software paradigms, development of a distributed operating system, development of a multimedia workstation, and Japanese language processing. Some of the software items developed in this research is now available as tools, and the results of the project was published by Springer-Verlag Publishing Co. in August, 1991[1].

• "Software Designer's Associate (SDA) Project (project leader, K. Torii, Osaka University)"

This project started as a three-year project in April 1987. At first it was organized by a consortium with the financial support of eight industries. In 1990, the number of industries increased to thirteen and a second project began with the aim of developing actual products. The main research topics are software design support and the establishment of a standard architecture for distributed & integrated development environments. The results of these studies are presented at the 1988 and 1989 meetings of the ICSE[2].

• "Σ-system Development Project in the Information Technology Promotion Agency (IPA) (project leader, T. Tsujioka and chairman of the Development Committee, Y. Ohno, Kyoto University)" supported by the Ministry of International Trade and Industry.

This project was aimed at developing a nation-wide distributed environment as a common platform for software development in industries, such as software houses by allowing them to be connected to a network of workstations having the Σ-OS based on Unix, and was also intended to develop various kinds of basic tools for programmers. In the middle of 1990, a five-year development project was completed and a new company (the third sector) was established in order to operate this system in practice and to supply services for software development activities conducted by the member companies.

The project defined and initiated research on "Data architecture," which is a set of common interface formats between tools to allow integration of the tools and program synthesis. This may be one of the most important practical results in the project[10]. Moreover, Σ-OS aimed at integrating System-V, and the Berkley Version of Unix gave an impulse to a movement toward unification and standardization of Unix. The Σ-system company has started international activities by opening relations with X-OPEN, OSF, and UI.

• "Project on Standardization in Software Engineering"

Standardization in information technology in Japan is being promoted mainly by Japanese International Standards Committee (JISC), the Standards Committee of the IPSJ, and the Japanese Standards Association (JSA) in response to the activities of international standardization organizations. Projects related to standardization in software engineering have, of course, been performed in Japan. The principal areas for standar-

dization are interfaces or interconnections between computers, between OSs and application programs, between different application programs, and between man and machine. The main projects and topics include Open System Interconnection (OSI), a Portable Operating System Interface for a Computer Environment (POSIX), Interfaces for Application Portability (IAP), interconnection and evaluation of CASE tools, system documentation, software evaluation, reference models, man-machine interfaces including windows, and icons.

4. Main Subjects of Research in Japan

In this section, we will describe the main subjects of research on software engineering during the last three years in Japan from the auther's own view.

• CASE (Computer-Aided Software Engineering) tools

No software engineering techniques are possible without the support of a computer environment. A discussion of CASE would include most of the subjects in software engineering, and for this reason few papers are concerned with CASe itself. In the United States, some technologies based upon Structured Analysis/Structured Design (SA/SD) have been commercialized as useful tools, which have come onto the market and gradually achieved widespread acceptance as CASE tools. These tools usually have special functions, such as graphic editors and functions for manipulating diagrams and pictures, which are sufficient to increase productivity. Especially in Japan, some functions have been provided to make Japanese characters easy to handle in practice. However, these functions are still not sufficient to satisfy high-level requirements, such as transformation from specification to design, and functions for checking and validation are still lacking. Realization of these high-level functions is one of the most difficult issues in software engineering, and research on this issue is still insufficient.

• Object-Oriented Methodology

There are many research activities dealing with the reuse of software parts. At the stage of treating concepts further upstream in the development process than structured analysis (SA), it is difficult to realize reusable software parts, since a very large knowledge-base is needed for the domain in order to divide a certain concept into parts. Many research activities therefore cannot go beyond the mechanical treatment of the formalized framework, which may be a flow graph, a state transition model, an entity relationship model, and so on. When SA/SD is used for design, there are also problems of deciding about data flow, control flow, and the set of states in order to make a correct or good design. The recently introduced Object-Oriented Analysis/Object-Oriented Design (OOA/OOD) involves the same problem.

OOA/OOD contains well-known elements for modeling, such as encapsulation, class-instances, message communication, and inheritance. However, it is difficult to design an object structure correctly, since it is composed of an inheritance structure and a composite structure, and a knowledge-base is usually needed for the objects' domain. In both SA/SD and OOA/OOD, it is necessary to investigate the relationship between the model structure constructed by using the above-mentioned elements and the structure of knowledge in the object domain. Several basic research projects to support object-oriented methodology are being conducted on object-oriented computation models and languages, concurrency control, and constraint mechanisms.

• Repositories

The essential features of a CASE environment include not only a methodology but also a repository that can support development activities. This may be called an encyclopedia, an Information Resource Dictionary System (IRDS), or a Software Engineering Data Base (SEDB); its purpose is to manage the attributes of intermediate products in the development process and the relations among them. In connection with this, there is an approach of using a repository of commercialized tools, such as HYPERTEXT. What kind of model should be used for databases and files is one of the most important research issues, and the Deductive and Object-Oriented Data base (DOOD) is considered one of the most promising candidates.

• User Interface Management Systems (UIMSs)

Use of Workstations with bit map displays is increasing rapidly, and the speed of CPUs has been much increased by the effects of RISC. Because of the rapid increase in the capacity of main memories and the common use of window systems, people are paying much more attention to research on and development of UIMS, these days. Some of them are actually used in practice, such as the tool-kit on X-windows and window managers, and some are being developed and evaluated, such as the extended Model-View-Controller (MVC) of Smalltalk. There are also some research projects devoted to constructing UI support systems based on meta-UI, dialogue, and knowledge modeling. These research activities are desirable and significant, because the design itself of a UI, including the visual design, can be left to expert designers if computer experts are able to use the research results to realize a supporting system for constructing UIs.

• Some applications of object-oriented methodology

Since the object-oriented approach is considered suitable for constructing UIs, there are many research activities related to the development of UIs using obj-

ect-oriented methodology. In addition, there are many research projects on the application of object-oriented concepts to knowledge representation problems and expert systems, real-time system software for exchange and communication, and description of software processes.

• Software Processes

Like CASE, software processes have attracted much attention. In 1990, the sixth International Workshop on software processes was held in Japan for the fourth time. Methodologies for modeling software processes and mechanisms for realizing them have not yet gone beyond the analogy of Shell Script and Command Interpreter, but they have gradually gained recognition over the past few years, from both the practical and formal points of view. The case studies presented at the above mentioned international workshop helped to give a concrete image of the methodologies. At the last domestic workshop, there was a proposal for a Kyoto DB founded on an object base with a product-relationship and a cooperative work model, modeling based upon attribute grammar, and formal description using OBJ[7] and LOTOS.

• Groupware

Groupware is deeply related to software processes. As well as software engineering, research on general Computer-Supported Cooperative Work (CSCW) has been rapidly advancing in the last few years, and it seems natural to apply this technique to the process of software development and maintenance, including cooperative activities. In research on network protocols, people usually design and define methods for communication between machines, and in research on UIs, machines are enabled to communicate with humans. On the other hand, in research on groupware, machines are enabled to support communication between humans.

There are two approaches in groupware research.

(1) Free style approach

Communication is left up to the people involved, and the machine supplies a platform and tools for this purpose. The electronic meeting system is a typical example, and requires a technique for multimedia communication, a database, and a user interface.

(2) Compulsory approach

The purpose of groupware is to establish a communication/action model, that is, to stimulate and optimize human activities in a certain field/area of cooperative work. In other words, it clarifies and confines the parts of the task that require human judgement. To realize this purpose, it is enough to enhance the E-mail system, and the resulting groupware can be used in software engineering directly.

In the first approach, there is also a communication model that includies the concepts of meeting administration protocol and operation right control.

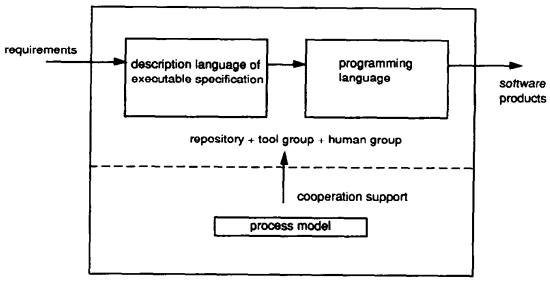

Fig. 1 New framework for software development (Ochimizu and Sunohara, modified by Ohno).

As UIMSs are designed by computer scientists and UIs are designed by UI designers, it seems logical to leave the design of concrete communication in cooperative work to sociologists and psychologis, within the framework of the communication model built by the computer expert.

In this section, we have briefly surveyed some of the major recent topics in software engineering, but of course there are also many research activities on traditional subjects in software engineering.

In communication software, there are research activities dealing with, for example, design support for communication software and support for communication protocols.

In the quality control/assurance domain, activities include research on quality development with respect to function development, that is, assignment of quality elements to software constructs at the stage of designing, description of processes for improving quality by numeric measure control, software reliability growth models using error importance and error correction processes, and generation of test sequences from specifications. In the specification domain research topics include design and verification methodologies based on formal specification, graphical visualization of software, structured editors for specification and manual documentation, derivation of specifications from natural language descriptions, and analysis of the behavior of concurrency control.

5. Concluding Remark

Finally, I have comments on future software engineering; more specifically, on technology transfer and on an appropriate framework for new developments in the future. Most problems in the software domain seem solvable if we can only manage to effectively organize technologies accumulated during twenty-years of software engineering and put integrated development environments into practice. At present, the "bridge" be-

tween research and development does not seem to be adequate, and therefore one of the most important issues is to fill this gap by supplying well-organized and practically usable development environments. A new framework for software engineering in the future is shown in Figure 1. This fugure may be widely recognized and need not be explained here in detail. Realizing practical development environments in full accordance with this framework should be our primary task.

Acknowledgement

The auther is grateful to Mr. Tsuneo Ajisaka, Associate Professor in the Department of Information Science at Kyoto University, who assisted him in preparing Section 4.

References

1. OHNO, Y. and MATSUDA, T. Distributed Environment—Software Paradigms and Workstations, Springer-Verlag Publishing Co. (August 1991).
2. KISHIDA, K., KATAYAMA, T. et al. SDA: A Novel Approach to Software Environment Deisgn and Construction, *Proceedings of the 10th International Conference on Software Engineering* (1988), 69~79.
3. *Trans. IPS Japan* (in Japanese), 1988~1989.
4. Research Reports of SIG Reports of Software Engineering, IPSJ (in Japanese), 1989-1990.
5. MATSUMOTO, Y. and OHNO, Y. Japanese Perspective in Software Engineering, Adison-Wesley Publishing Co., 1988.
6. Toshiba Review (in Japanese) (May 1990).
7. Fujitsu (in Japanese) (September 1990).
8. IPSJ, "Footprints in Thirty years of the IPSJ", (in Japanese), 1990.
9. FUTATSUGI, K., GOGUEN, J., JOUANNAUD, J. and MESEGUER, J. Principles of OBJ2, *Proceedings of the 12th Symposium on Principles of Programming Languages, ACM* (1985), 52 - 66.
10. NAKATA, N., NISHIJIMA, A., KONISHI, Y. and KUBO, T. "Sigma Tool Implementation and Integration," Proc. of Info Japan '90, IPSJ, Oct. 1990, pp. 449/456.

(Received April 1, 1991)