Building Large-Scale and Corporate-Wide Case-Based Systems: Integration of Organizational and Machine Executable Algorithms

Hiroaki Kitano* Akihiro Shibata+ Hideo Shimazu+
Juichirou Kajihara* Atsumi Sato*

Software Engineering Laboratory*
NEC Corporation
2-11-5 Shibaura, Minato,
Tokyo, 108 Japan

C&C Information Technology Research Laboratories+
NEC Corporation
4-1-1 Miyazaki, Miyamae,
Kawasaki, 216 Japan

Abstract

This paper reports a case study on a large-scale and corporate-wide case-based system. Unlike most papers for the AAAI conference, which exclusively focus on algorithms and models executed on computer systems, this paper heavily involves organizational activities and structures as a part of algorithms in the system. It is our claim that successful corporate-wide deployment of the case-base system must involve organizational efforts as a part of an algorithmic loop in the system in a broad sense. We have established a corporate-wide case acquisition algorithm, which is performed by persons, and developed the SQUAD Software Quality Control Advisor system, which facilitates sharing and spreading of experiences corporate-wide. The major findings were that the key for the success is not necessary in complex and sophisticated AI theories, in fact, we use very simple algorithms, but the integration of mechanisms and algorithms executed by machines and persons involved.

1 Introduction

This paper reports a case study on a large-scale and corporate-wide case-based system. Unlike most papers for the AAAI conference which, exclusively focus on algorithms and models executed on computer systems, this paper heavily involves methodologies and organizational constraints, which arise in the course of knowledge-based system development and deployment.

The central claims which we wish to deliver are: (1) the successful corporate-wide deployment of the AI system must involve organizational efforts, as a part of the algorithmic loop in the system in a broad-sense, and (2) the system should be flexible and robust enough to cope with incremental and changing nature of the corporate structure and activities. In essence, this idea dictates the significance of the integration of algorithms for the organizational activities and the machine executable program.

The underlying thrust involved in the proposed approach is the idea that the case-based system should be viewed as a part of a corporate-wide, or department-wide, strategic information system, which enhances *sharing of experience*. Traditionally, CBR systems have been developed as a kind of expert system, which provide solution to the problem [Riesbeck and Schank, 1989]. Of course, this is still a viable approach. However, the organizational and economic impacts would be far greater, when the case-based system was integrated as a part of a corporate-wide information system. It would be used as a new media, which facilitates spreading of knowledge and experiences.

The development of the large-scale and corporate-wide case-based system should involve a carefully planned case-acquisition process. Many research efforts have focused on the CBR systems or theories, and none of them have addressed the problem of how to acquire cases. In addition, many CBR systems have been built on domains which have already been carefully investigated — domain experts already know what features are involved, which features are important, and what values fill each feature. It is our observation that, for many possible CBR application domains, even experts do not have a well-organized idea of what factors are involved. Because of the fact that many real-world application domains are not well knowledge engineered and that the corporate activities changes over time, the CBR system should be able to cope with dynamically changing case structures, cases with missing and inaccurate information, and other constraints which arise from the real-world deployment.

As an example of the corporate-wide case-based system development efforts, this paper reports the SQUAD Software Quality Control Advisor system and the corporate-wide case acquisition process for building the case-base for the SQUAD system. The task domain is *Software Quality Control (SWQC)*[Mizuno, 1990]. We are conducting a corporate-wide knowledge acquisition process, which involves an estimated annual labor investment of over 200 person-years, which provides over 3,000 cases per year. Concurrent to the case acquisition efforts, the SQUAD system has been deployed, which meets various constraints which arise from real-world development and deployment.

This is perhaps the first attempt to employ a case-based system as a part of corporate-wide information system with corporate-wide knowledge acquisition process. Due to the fact that the system was actually deployed and the case acquisition process was introduced corporate-wide, we are not able to conduct controlled experiments to verify some of the hypotheses involved. Although we recognize the importance of scientific and controlled data collection, such attempts would costs bil-

lions of dollars, as the proposed process involves around 150,000 employees. Instead, this paper will describe reasons for decision we made in this project. Scientific investigation on the efficacy of each approach would be made possible when we taking cognizance of the numbers of such attempts in various domains, with various organizational constraints. It is our wish that this report will serve as the basis for discussions on how large-scale and corporate-wide case-based systems should be developed and deployed.

2 SWQC as a Corporate-Wide Knowledge Acquisition Process

2.1 Software Quality Control

Quality control is an essential element in modern industrial society. High quality products provide competitive edge, and assure a higher reliability. Software is by no means an exception. Recent establishment of the ISO-9000-3 standard [ISO, 1990] clearly urges the software industry to attain a high-level of software quality control. One of the major approaches to improving the product quality is the QC (Quality Control) activity, which assumes voluntary participation of all employees forming small groups in each factory and department. Particularly, QC activities have been successfully introduced in many Japanese firms. QC activities have been carried out in virtually every stage in manufacturing processes. Very comprehensive and voluminous reports on improvement in quality and productivity have been filed. In 1981, we established a company-wide organizational structure to enforce the *Software Quality Control (SWQC)*. Traditionally, QC activities have been viewed as an important process involving worker participation as well as a substantial means of continuous improvement in production systems. Each reported case provides analysis of problems under the status quo, possible causes, counter-measures taken, and effects of the counter-measures. Each case is reported as a two-page length paper and a form in which to fill major issues in the case.

2.2 Establishing the Knowledge Engineering Process

We made a radical departure from the traditional view of the QC activity. We consider the QC activity as a company-wide knowledge acquisition process. Thus, we established a knowledge engineering loop for the SWQC domain. Figure 1 illustrates organizations involved and the process flow. Each SWQC activity group in the entire NEC group submit their case reports. The review committee reviews each case. The review result classifies reported cases into 1% of *Best Cases*, 10% of *Selected Cases*, and 90% of *Pool Cases*. Best cases and selected cases are chosen, based on its quality of analysis, significance of the effects, and universality of the problem. They are considered to be the norm for the case report. Naturally, these cases are given top-priority in case-base building, and serve as the basis of the domain model and the case format definition.

The SWQC activity was started out in 1981 with no substantial case format. The rough case format and the case report category were introduced as the SWQC grew into the corporate-wide activity. Recently, we have begun to provide specific feature sets to all participating groups, so that they would help making standard case report format and improve analysis quality. This cycle is a step-wise cycle. It starts out from a vague standard and provides only top-level features. As knowledge engineering progresses, more detailed feature sets are provided. This is because we start to understand the domain model and what features should be identified. Due to the turn around cycle, up-grading the case format would require one year at minimum. This work is in the end of the fourth cycle in this loop. It is only after the third cycle that the minimum features could be identified, so that it was possible to build a crude case-base. The feature set will be further sophisticated, as this cycle progresses. Thus, the absolute requirement for the case-based system on this domain is that the incremental modification to the case-base, particularly the indexing features, can be accomplished with minimum cost, if not cost free.

In order to facilitate the stable flow of high quality cases which covers nearly all software domains, several organizational measures have been introduced:

Filtering: When the knowledge acquisition was scaled up to a corporate-wide level, some case filtering scheme would be necessary to select high quality cases, from among all cases with varying analysis quality. This is essential in establishing the norm for the high quality case.

Incentive System: This is a management issue for the corporate-wide knowledge acquisition. A top-down control has been established regarding the activity framework, awarding, symposium, conventions, and other incentive systems. Also, a bottom-up control of operational procedures and system critique scheme has been introduced, in order to maintain involvement of the participants.

Feedback: All the best cases and the selected cases have been compiled in a book form, and distributed to all involved sections.

Distributed Control: Each department hosts internal competitions, so that only qualified cases are reported to the company-wide review committee. This avoids overloading the review committee.

As a result of these measures, the number of acquired cases has increased significantly, and has reached the saturation point. Currently, over 3,000 cases of quality and productivity improvement measures have been reported every year. Over 20,000 cases have already been accumulated, as of December 1991 (Figure 2).

The main point of this part of the project is that the algorithm for case acquisition is essentially executed by people, not by machine, thus requires organizational supports and sustaining efforts for years.

3 The SQUAD System

3.1 System Requirements

As is always the case, the real-world deployment imposes various constraints, some of which require a major compromise in straightforward implementation of the laboratory-level models. The SWQC domain is not an exception. Some of the major requirements are:

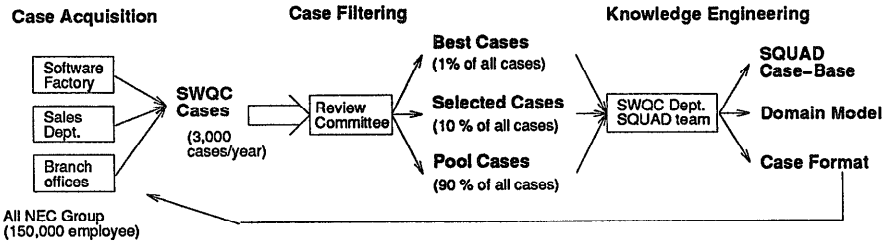

Figure 1: SWQC Knowledge Engineering Loop

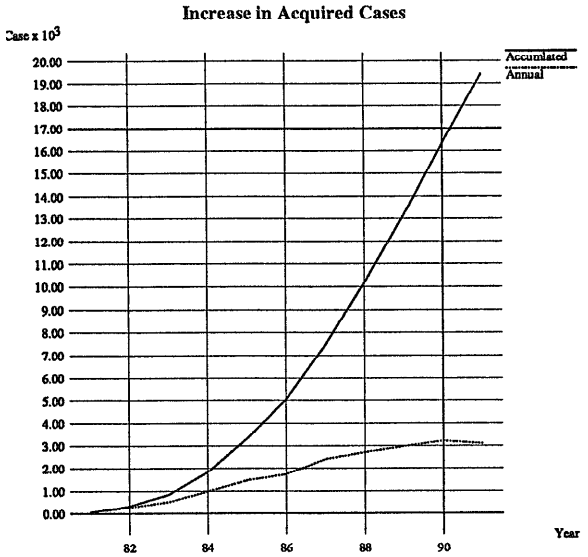

Figure 2: Number of Acquired Cases

Response time: The system should provide fastest response time, to be a part of a corporate information system. When the central server approach is taken in the deployment, the potential size of the user group is over 150,000. Although the use of massively parallel machines would guarantee quick retrieval (see [Kolodner and Thau, 1988] and [Cook, 1991]), budget constraints precluded this option.

Low Development Cost: The development cost should be as low as possible. The bottom line is that the case-base can be built and maintained by one or two full-time employee.

Robustness: The system should allow missing information, inaccurate description, and other erroneous data entries. It is not possible, both from economic and organizational reasons, to force every QC activity group to report cases with a uniform level of accuracy and specificity. It is also not possible for the case-base builder to frequently inquire about details of the reported case.

Flexibility: The system should allow a maximum level of flexibility to the user, in specifying cues for case retrieval.

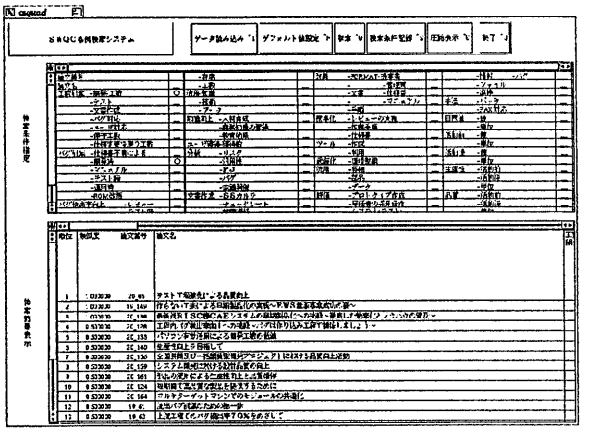

Figure 3: SQUAD System

Incremental modification: The system should allow an incremental modification of index features for the case-base. As actual products and systems change, the domain itself changes. It is not economically and logistically feasible to carry out exhaustive knowledge engineering, to identify a well-formed index structure over 20,000 cases with 3,000 additions every year.

3.2 System Architecture

The SQUAD system is a case-based software quality control advisor system. In its first development phase, the SQUAD system does not involve an adaptation phase. It only has a case retrieval phase. The main reasons are (1) retrieval of cases alone suffice for most advising tasks, and (2) the domain is so complicated and ill-formed, that any form of adaptation scheme would require significant development costs and the system behavior would be unstable, as ourself do not fully understand the nature of the domain. The screen of the SQUAD system is shown in Figure 3.

Coping with Case-Base Modification and Flexible Query The critical issues in the SQUAD system are (1) ability to cope with case-base modification after each knowledge engineering cycle, (2) robustness against missing information, and (3) ability to cope with varying query specification level.

The SQUAD system deals with these requirements

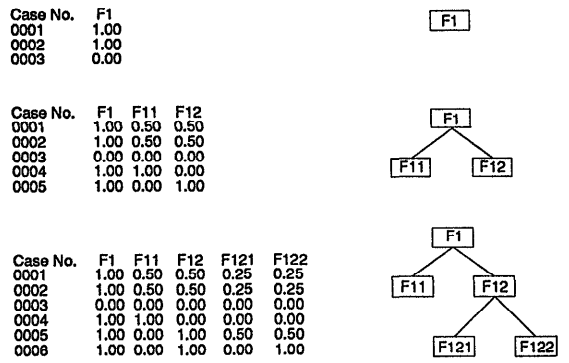

Figure 4: Case-base modification after each cycle

within a consistent and uniform model. Figure 4 illustrates how the case-base, represented in a flat table, can be modified after each knowledge engineering cycle. Initially, only feature F1 is defined in the case-base. Cases with this feature (Case 0001 and 0002) are assigned with 1.0, and a case without this feature (Case 0003) is assigned 0.0. In the second cycle, feature F1 was found to have subcategories: F11 and F12. Due to the lack of specific information for Cases 0001 and 0002, values for these cases on F11 and F12 are assigned as 0.5 for each new feature. In case three subcategories were found, the value would be 0.33. Basically, we assume equi-probability distribution regarding which subcategory may be correct. When information is available, either 1.00 or 0.00 will be assigned. Assuming that F12 was found to have a further subcategory in the third cycle, e.g. F121 and F122 in Figure 4, the same equi-probability rule is applied for cases, which do not have this information detail. This is a simple mechanism, but it was proven to be effective in coping with changing case structures. Thus, it enables the SQUAD system to cope with incremental modification of case-base in each cycle, and maintains reasonably accurate retrieval, even with missing information.

This mechanism also allows flexible and consistent case retrieval. Each case can be described at any specificity level, and the users can specify a query cue at any level. So far, CBR retrieval methods have been assuming that the query, or the cue case, is specified and represented with the same level of abstraction with cases in the case-base. For example, if a feature `Tools-Used` is specified at the level of specific name for the tool, the assumption is that the cue case is also specified at this level. Similarity would be computed, using the domain heuristics or statistical means. However, the assumption that the user can specify features at the same level of abstraction with the case-base is too strong to be implemented in the practical system. For example, users tend to specify kinds of tools (such as `Spec-Acquisition-CASE`, `Programming-CASE`, or `Version-Control-Tool`), rather than specific name for tools (such as `ProSpec`, `SEA/I`, or `Life-Line`). By the same token, cases collected do not necessarily contain information at the same abstraction level.

This is illustrated in Figure 5. The figure shows an example of user query selection and two cases. Typical

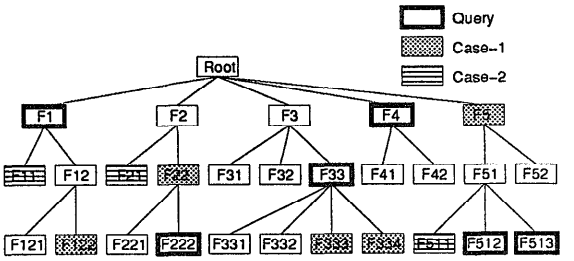

Figure 5: Examples of Feature Selections by User

query is specific for some of the features, but not for other features. However, for example, the query should return Case-1 with highest similarity results, because only difference in between Query and Case-1 is the specificity of features selected. Actually, all selected items in Query and Case-1 are subsumed in other selected features, in one way or the other.

For example, the query specifies F1, F222, and other features. Case-1 and Case-2 match F1, because the feature F1 value for Case-1 and Case-2 is 1.0 (See Figure 4). On the other hand, F222 would have a low similarity score, because only partial information is provides for Case-1 and Case-2. Actually, the similarity score between F222 and F22 is 0.5. F222 and F21 have a similarity score 0.0, because Case-2 has F21 feature which makes the F22 value to 0.0. In this example, assuming equal weight distribution, the similarity score for Case-1 and Case-2, for the query, are 0.305 and 0.167, respectively.

Retrieving Cases from Sparse and Dynamically Changing Salient Features The case indexing is very sparse and salient features change in each retrieval session. With regards to the sparseness, on an average, only 3.4 out of 75 features (as of the third cycle) are specified for indexing cases. This is due to (1) broad coverage for the SWQC domain, and (2) the use of flat table case-base representation. Since the coverage domain is so broad, some features, relevant to cases in one of the subdomains, are not relevant to cases in other domains. Also, the use of flat table type representation, a constraint imposed from practical consideration[1], drastically increased the sparseness. If a conventional similarity metric, used in many CBR systems, is applied which assumes a dense indexing of features, retrieval would be inaccurate. At the same time, the goal of retrieval changes in each session. Thus, it is not possible for us to improve quality and speed of retrieval with a sophisticated indexing scheme as seen in [Hammond, 1986], [Hunter, 1988], and [Kolodner, 1984]. Let X, Y, and A be a feature vector for the query, a feature vector for the case in consideration, and a weight matrix. The conventional similarity metric is computed by:

[1]To be more specific, we need to develop case-base using the Lotus 1-2-3 type spreadsheet interface. The case-base entry should be made by simply specifying items which can be identified from the reported case.

$$Sim(X,Y) = \frac{X^t diag(\frac{A}{\|A\|})Y}{\| X \| \| Y \|}$$

However, this method involves matching many irrelevant features, thus significantly degrading retrieval accuracy. Instead of using this similarity equation, an equation is defined which ignores irrelevant features for each retrieval. The proposed alternative formula is:

$$\begin{aligned} Sim_X(X,Y) &= Sim(C_X(X), C_X(Y)) \\ &= \frac{C_X(X)^T diag(\frac{C_X(A)}{\|C_X(A)\|})}{\| C_X(X) \| \| C_X(Y) \|} \\ &= \frac{X^T diag(\frac{A}{\|A\|_X})Y}{\| X \|_X \| Y \|_X} \\ \| Y \|_X &= \sqrt{\sum_{i \ s.t. X_i \neq 0} y_i^2} \end{aligned}$$

This problem is illustrated using a simple example. Let us assume that a query X, a case-base which contains cases Y and Z, and a weight matrix A, as shown below:

$$A = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad X = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \quad Z = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

When appling a traditional similarity measure, similarities between X and Y (Sim(X,Y)), and X and Z (Sim(X,Z)) would be computed as:

$$\begin{aligned} Sim(X,Y) &= \frac{1}{\sqrt{2}\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{3}} & & \\ & \frac{1}{\sqrt{3}} & \\ & & \frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \\ &= \frac{1}{2\sqrt{3}} \\ Sim(X,Z) &= \frac{1}{\sqrt{2} \cdot 1} \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{3}} & & \\ & \frac{1}{\sqrt{3}} & \\ & & \frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\ &= \frac{1}{\sqrt{6}} \end{aligned}$$

Similarities for the same example, using the proposed metric would be:

$$\begin{aligned} Sim_X(X,Y) &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & & \\ & \frac{1}{\sqrt{2}} & \\ & & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \\ &= \frac{1}{2} \\ Sim_X(X,Z) &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & & \\ & \frac{1}{\sqrt{2}} & \\ & & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\ &= \frac{1}{2} \end{aligned}$$

In essence, this method ignores features which are considered to be irrelevant. Cain et. al. [Cain et. al., 1991] also proposed a similarity metric, which applied less weight to irrelevant features. However, their scheme heavily relies upon the domain knowledge used in Explanation-Based Learning (EBL). Their method cannot be applied to the SWQC domain, because such domain knowledge is extremely difficult to obtain, and relevant features dynamically change at each query. One other approach is to keep track of the past solution [Veloso and Carbonell, 1991]. However, it was not possible to take this approach because no such traces were available.

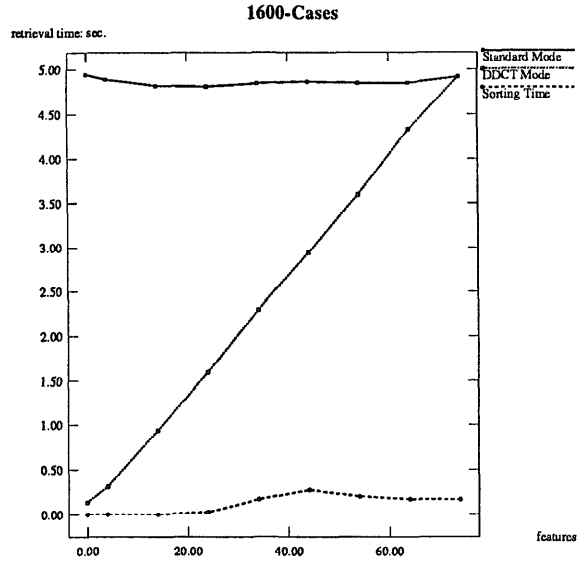

Figure 6: Retrieval Performance

The proposed method provides faster response time and higher retrieval accuracy. The response times have been measured for various numbers of cue features, and various sizes of case-bases. Figure 6 shows case retrieval time for the traditional method, which checks matching for all features, and the new method. In case of the traditional model, the retrieval time is almost constant, regardless of the number of cue features specified by the user. As our analysis discovered, the average number of cue features for each retrieval was around 3. Thus, the method significantly curtailed the computing cost — 1/10 of the traditional method. Also, the retrieval accuracy, with subjective judgement by users, records almost 20% improvement over the traditional method. This coincides with the results reported by [Cain et. al., 1991]. Figure 8 illustrate the difference in distance between cases in the case-base (only 30 cases are used in this figure to make it readable). The left hand side is the cluster created by the traditional similarity metric and the right hand side is the cluster created by the new method. Clearly, the new method provides fine-grain distance distinction between cases, which has been signified by the improved accuracy.

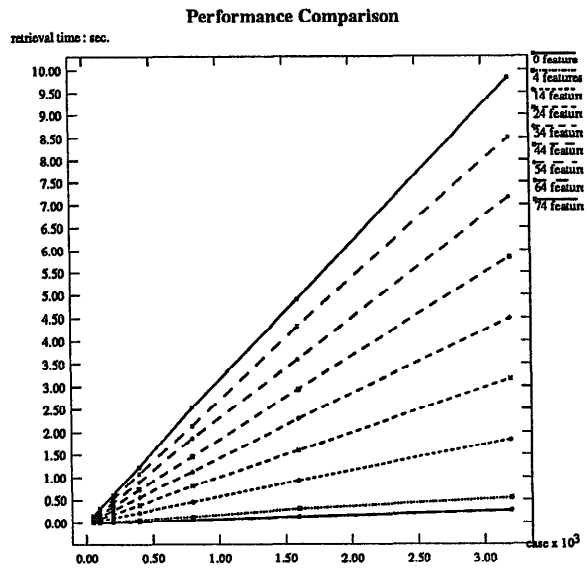

Figure 7: Scalability

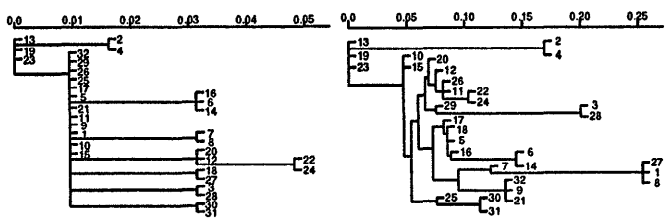

Figure 8: Cluster of Cases

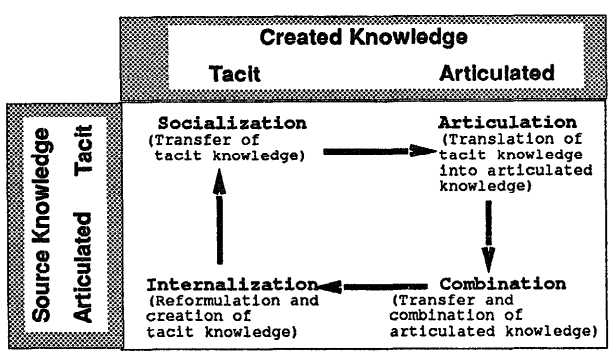

Figure 9: A Process of Organizational Knowledge Creation

4 Discussions

4.1 Organizational Knowledge Creation

Perhaps one of the most advanced organization theories in the area of business administration is a theory of organizational knowledge creations proposed by Nonaka [Nonaka, 1990][Nonaka, 1991]. He assumes the existence of tacit knowledge and articulated knowledge. He observes that organizational knowledge spreads through the process of (1) socialization, (2) articulation, (3) combination, and (4) internalization (Figure 9). He argues that the articulation and the internalization are a decisive step for improving the organizational knowledge.

The SQUAD system and SWQC activity can be a powerful scheme for organizational knowledge creation. First, the SWQC activity encourages a certain level of articulation of knowledge, since all participants are supposed to submit reports and forms, which describe their improvement activities. Although not all of such reports are articulation of tacit knowledge, there are some reports which articulated their tacit knowledge. Socialization process takes place, while they are working as a group. We have discovered that the knowledge acquisition through the group activity is much more efficient and effective than individual activity by experts. Although we do not discount the value of well trained experts, it is logistically not feasible to rely on experts on software quality control and productivity in the corporate-wide scale. In addition, we consider that each member in each department is a tacit expert in the field, as they are the most experienced persons in the specific task. Next, the knowledge engineering conducted by the SWQC department further augments the level of articulation, and combination took place, when the domain ontology and models were built. Finally, the SQUAD system allows efficient access to the SWQC cases, which enhances internalization of articulated knowledge in different divisions. It should be noted that our knowledge source is case-bases, not rule-bases. Thus, we are much closer to tacit knowledge than abstract rules.

Of course, whether such a knowledge transfer process actually took place has not been made clear. We only knew that the project have made substantial contribution for software quality and productivity, which is estimated to worth over a hundred million dollars per year. Since effects of such improvements accumulate as the process continues, the net effects are expected to be over multi-billion dollars. In addition, the proposed approach is consistent with the modern organizational theory. By identifying the proposed scheme in the perspective of the organizational theory, it is possible to logically deduce the direction which should be pursued.

4.2 The Next Move

In Nonaka's model, however, the knowledge was transmitted to different divisions only in the form of articulated knowledge. We wish to go one step further, past the Nonaka's model, by providing a multi-media user interface to possibly transmit a part of tacit knowledge. The multi-media SQUAD not only provides user-friendly access to the SWQC case, but also enables us to record discussions and talks by the person involved. Often, the essential part of the experience is hard to formalize. By using the multi-media interface, the users can obtain both formal and informal information.

One other aspect of improvement is the combination process. We wish to provide not only the domain ontology and models, but also domain heuristics. Acquisition of useful heuristics is a major subject in data-base mining. This is particularly important for case-based systems, which do not provide explicit rules themselves. Once we start to acquire rules, the process of organizational knowledge creation will be carried out with a full spectrum of knowledge — rules, cases, and informal knowledge.

4.3 CASE by Case

The idea of *sharing of experience* applies to other software development domains. One of the most promising approaches which we propose is the *CASE by Case* paradigm, in which Computer-Aided Software Engineering (CASE) tools support case retrieval and certain adaptation levels. In essence, this approach uses past cases of software design and programs during the development of the new software system. There have been several attempts to reuse programs and design results in the past. However, most of such projects have failed, due to inflexibility in the retrieval specification, difficulties in defining the design representation, etc. Our experience in SQUAD system demonstrates that a well-designed case retrieval mechanism, togather with organizational efforts, would allow an entire system to grow. Thus, an incomplete representation scheme could be the starting point (although complete representation is obviously preferred), due to the similarity search capability of the CBR paradigm. Introduction of the CBR idea offers a new opportunity to establish a practical and cost effective software engineering paradigm.

5 Conclusion

This paper has described an approach for building a large-scale and corporate-wide case-based system, as exemplified in the SWQC and the SQUAD software quality control advisor system. The main thesis of this paper is that the model, or algorithms, for the organizational activities and the machine executable program should be integrated in order to establish a large-scale and corporate-wide AI system. Since such an attempt would be fairly substantial for any corporate system, the project should involve a carefully organized knowledge acquisition process and a CBR system which is flexible enough to cope with organizational and other constraints inherent in real-world deployment.

It took us almost a decade to establish a corporate-wide process of case acquisition, which provides a stable flow of up-to-date cases of software quality control and productivity improvement. The current statistics indicates that this process offers quality and productivity improvement over a hundred million dollars each year. The net effects could easily exceed multi-billion dollars. We believe that this is one of the most substantial knowledge engineering projects ever conducted.

The proposed model also presents substantial consistency with one of the most advanced organization theories, called the theory of organizational knowledge creation. This is, by no means coincidence, since we are striving toward a new paradigm of corporate information system, which enables *sharing of experience*, which is the central dogma in the proposed paradigm.

The most important findings drawn from our experience is that integration of the organizational process and the flexible system design to cope with the changing corporate activity and constraints, exhibits maximum utility in improving the entire production process, which has significant economic impacts. The algorithm of the SQUAD system is very simpl. However, the point we wish to make is the fact that this is the best approach for the incremental deployment of the corporate-wide system which matches changing corporate structure and task domains.

Acknowledgements

The authors would like to thank Dr. Mizuno, Dr. Fujino, Mr. Saya, and Mr. Kai for continuous support for this project. More than that, however, we would like to thank all NEC employee who have engaged in this project for a decade, and continue to participate in this endeavor in one way or the other.

References

[Cain et. al., 1991] Cain, T., Pazzani, M., and Silverstein, G., "Using Domain Knowledge to Influence Similarity Judgement," *Proc. of Case-Based Reasoning Workshop,* 1991.

[Cook, 1991] Cook, D., "The Base Selection Task in analogical Planning," *Proc. of IJCAI-91,* 1991.

[Hammond, 1986] Hammond, K., *Case-Based Planning: An Integrated Theory of Planning, Learning, and Memory,* Ph. D. Thesis, Yale University, 1986.

[Hunter, 1988] Hunter, L., *The Use and Discovery of Paradigm Cases,* Ph. D. Thesis, Yale University, 1988.

[ISO, 1990] ISO, *Guidelines for the application of ISO 9001 to the development, supply and maintenance of software,* DIS 9000-3, ISO, 1990.

[Kolodner, 1984] Kolodner, J., *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model,* Lawrence Erlbaum Assoc., 1984.

[Kolodner and Thau, 1988] Kolodner, J. and Thau, R., *Design and Implementation of a Case Memory,* GIT-ICS-88/34, Georgia Institute of Technology, 1988.

[Mizuno, 1990] Mizuno, Y. (Ed.), *Total Quality Control for Software,* Nikka-giren, 1990 (in Japanese).

[Nonaka, 1991] Nonaka, I., "The Knowledge-Creating Company," *Harvard Business Review,* Nov.-Dec., 1991.

[Nonaka, 1990] Nonaka, I., *A Theory of Organizational Knowledge Creation,* Nikkei, Tokyo, 1990. (in Japanese)

[Riesbeck and Schank, 1989] Riesbeck, C. and Schank, R., *Inside Case-Based Reasoning,* Lawrence Erlbaum Associates, 1989.

[Veloso and Carbonell, 1991] Veloso, M. and Carbonell, J., "Variable-Precision Case Retrieval in Analogical Problem Solving," *Proc. of Case-Based Reasoning Workshop,* 1991.