Migrating Legacy Software towards New Technology

Lei Wu[1], Houari Sahraoui[1], Petko Valtchev[1]

[1]Dept. of Computer Science and Op. Res., University of Montreal
CP.6128 Succ. Centre-Ville, Canada, H3C 3J7
{wulei, sahraouh, valtchev}@iro.umontreal.ca

Abstract. Focusing on the legacy issues toward new technologies, we present our legacy migration methodology that composed of automation techniques, migration process, and collaborative migration support system to facilitate legacy software migration. It has been partially applied in our experimental migration project, and shows promising results.

1 Introduction

With the continuing development of modern computing technology, long-life legacy software systems often face significant modifications to meet with new requirements. One of the most important changes is the shift of new computing technology. It is raised by both external collaboration with other modern systems and internal quality enhancement needs. This trend requires the migration of legacy software towards new technology, such as migrating procedure system towards OO technology; migrating stand-along system towards web computing (e.g. Web service, XML), etc. This type of real-life reengineering practices are normally high-cost and high-risk. In this paper, we discuss the legacy issues towards new technologies, and present our migration methodology that composed of automation techniques, migration process and collaborative migration support system, which is built on workflow technology, thus to facilitate legacy software migration projects.

2 Technology-targeted Automation Technique

Many new computing technologies emerged in the past decade, and more are on the way to come in the near future [3]. According to particular circumstance that legacy software may face, different modern technology may be selected as the migration destination. For example, migrating legacy system towards web/mobile computing is quite different from migrating procedural software into object-oriented technology. For each targeting new technology, different migration techniques may be applied, and some parts of migration tasks are suitable for automation. For example, Houari et al. proposed a concept formation-based automation technique to elicit object model from legacy procedural code [1]. We focus our research work on migration techniques in two directions. One is legacy architecture recovery; the other one is legacy re-

modeling towards destination technology paradigm. These two aspects are essential in many migration projects. The result can be directly applied to facilitate legacy migration. We have developed a dynamic visualization technique to recover legacy design and architecture, thus to fulfill the first mission. Meanwhile, we are conducting technology-targeted re-modeling techniques for different modern computing paradigms. These automation techniques will be applied for different migration destination.

3 Progressive Migration Process

Meanwhile, a controllable migration process is also vital to the success. Legacy systems are often large, multiyear systems, and also critical to their owners. Migrating such systems in a single deployment process will create a high risk of failure and the process will be time consuming, which means high cost. Furthermore, the targeting new technology normally does not consistent with legacy code. Therefore, new paradigm modeling of legacy software to shift towards new technology is an inevitable course for most migration projects. To avoid the disadvantages of one-shot migration, we propose an incremental legacy system migration approach. The migration process is divided into many steps. Each of them focuses on migrating a certain part of the target system. By developing an integrated migration process control environment, our approach also aims at providing a flexible management means for large migration projects. Thus, legacy system will be divided (according to a decomposition strategy) into logically separated parts. During each migration iteration process, a set of such parts will be migrated, leading to an increase of the portion of renovated code, and to a respective decrease of legacy code that are still running. Eventually, the system will be completely migrated to new technology based system. Since the modernized code cannot communicate with the remaining of the legacy system, a bridge code will be created that helps the two parts cooperate together to work as an integrated hybrid system. Hence, our approach constitutes an in-operational migration process that allows the target system to run with no downtime. Our approach has three major benefits. **First**, such an approach substantially reduces the risks and the cost of a migration project. In fact, since no single deployment strategy is adopted, we do not need to allot all the resources to the migration project at once. In a relatively short period, a part of the legacy system will be renovated and available for an (partial) evaluation of the migration impact. Meanwhile, the hybrid system will keep working, which will largely reduce losses from downtime. **Secondly**, at the starting point, preference will be given to the most tractable parts of the legacy system, while the remaining parts will be considered with respect to their dependency (service requests) on the previously migrated parts. Thus, a reworking of the code that has already been migrated will be avoided. **Finally**, our approach provides a tangible management mechanism by dividing the whole project into separate tasks (to be assigned to independent teams) that will be more flexible to assign and manage. Moreover, a workflow management platform will ease the control of a large migration project.

Migration Sequence Priority

After successfully decompose legacy system, we'll face the question that which MU (Migration Unit) should be migrated first? A certain kinds of criteria should be adopted to rate the migration priority:

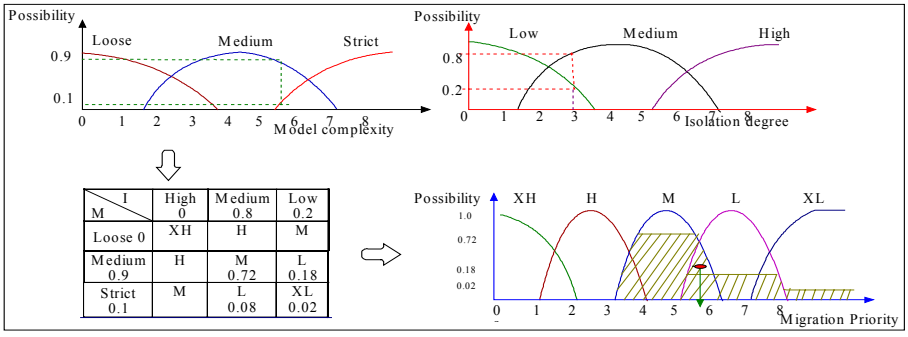
- *Module complexity*: metrics are applied to determine the complexity degree;
- *Isolation degree*: relationships with other MUs (migration units); are automatically calculated in form of reference number (weights). In another word, the less relationships with the rest of the system, the higher isolation degree it will have;
- *Module importance*: by the help of our dynamic visualization analysis tools, those MUs that perform higher important tasks of the system will be given a higher number of importance;
- *Understandability*: a subjective analysis and other source code metrics are combined together to indicate the understandability degree of a particular MU. The comparative difficult order will reflect the priority sequence;
- *Feasibility of applying automation techniques*: a number of randomly selected sample code segments will be used to test the feasibility of using automatic techniques to elicit new paradigm model from legacy code. The easier the MU has, the higher feasibility degree it will have, thus reflects its priority order.
- *Environment independency*: if a given MU relies little on any specific application environment (such as OS, hardware, language, etc.), it will have a higher degree of environment independency.

Priority Fuzzy Synthesis

The above criteria may not be coincident with all together. We apply fuzzy logic to indicate the final sequence priority order. It is an expert system that uses the collection of fuzzy membership functions and rules to deduce conclusion. The rules in a fuzzy expert system are usually similar to the following:

$$\textit{if } x \textit{ is low and } y \textit{ is high then } z = \textit{medium} \qquad (1)$$

Where x and y are input variables, z is an output variable, low/high/medium are membership functions defined on x, y and z. The part of the rule between "if" and "then" is the rule's premise which describes to what degree the rule is applicable. The part of the rule following "then" is the rule's conclusion that assigns a membership function to each output variables. We'll get a synthetic priority result by applying all the rules.



I / M	High 0	Medium 0.8	Low 0.2
Loose 0	XH	H	M
Medium 0.9	H	M 0.72	L 0.18
Strict 0.1	M	L 0.08	XL 0.02

Fig. 1. Fuzzy synthesis of migration priority by 2 criteria

4 Collaborative Migration Support System

In a large legacy migration project, a sophisticate migration support system is also keenly needed. It will be used as a common collaborative migration platform to control the whole migration process and human resources. We aim at providing a comprehensive migration support system (our project name: CARIBOU) to systematically manage the migration process and team collaboration. Meanwhile, it works as an integrated environment to accommodate various tools and techniques applied in detail migration tasks. The ultimate objective is to maximize the efficiency and to increase the possibility of success for any migration project. It works as a general platform that automates and monitors the migration progress. It also provides migration support tools, automatically collects/analyzes a large range of process/project metrics and accommodates quantitative migration control. By this way, it will greatly minimize the possibility of project failure. Furthermore, it will provide an idea collaboration framework to large migration projects, especially for those have geographical and time zone barriers.

We use dynamic workflow technology as one of the cornerstones in our migration support system. Workflow technology aims at providing computer aid supports to the modeling, execution and supervision [2]. The *Workflow Definition Tool* supports the capture of the definition of the migration process. The *Workflow Engine* performs the management of the execution of workflow processes in an operational environment. Finally, the *Workflow Monitoring and Control Tool* monitors the status of migration workflow processes and dynamically configures the runtime controller in a progressive migration project. Our progressive migration support platform will provide important functions to facilitate collaborative migration teamwork. The following are some of them:

- *Automatic activity modeling and modifying:* migration activities are dynamically modeled and monitored. They will be identified by their tasks and procedural constraints. When condition changes, it will support the dynamical modification and adjustment.
- *Migration process control:* the monolithic legacy source code will be divided into MUs (*Migration Units); W*fMS will manage the process for each task.
- *Task control of individual team members*: in our migration support system, WfMS will read workflow definition, execute and keep the track. The tasks assigned to individual programmer in a team will be well organized by WfMS, including time constraint control, sub-task partition, hybrid system test, bridge code version control, etc.
- *Request/responses control:* the question requests between individual maintainers will be processed through the interaction with the workflow engine. In our migration support platform, the interaction of team worker's requests and responses will be monitored through our multi-agent workflow modules. Not-answered questions will be prompted to team leader. This will largely facilitate problem tracing.

- *Administration and monitoring of the whole project progression:* this will allow migration project manager to track the status for each migration. A comprehensive project/process metric set is defined at the beginning of migration. Several predication and scheduling models are built within CARIBOU environment. With the progression of migration project, historical data will be daily collected, analyzed and saved. The dynamic change and analysis results are generated to facilitate the control of teamwork. When any calculation result exceed the threshold, alarming information will be sent to related person; the critical point will be pinpointed, and causation analysis will be provided by automatic tracing methods, therefore to ease the way to find out the possible origin. Predicate model will also calculate the possible damage of the event, and show possible remedy routes.
- *Risk analysis*: CARIBOU will perform risk analysis to optimize the organization of workload dispatch throughout the migration workflow. By using predefined impact model, each activity unit of the migration workflow will be assigned a crisis critical degree separately. Impact model will predicate the possible damage of the whole project, and the possible remedy cost, time delay etc, Stochastic sequence will be adopted to simulate the random and sporadic nature.

5 Conclusions

In this paper, we have presented our migration methodology that focuses on solving migration issues towards new technologies. It is composed of automation techniques, migration process and collaborative migration support system. It has been partially applied in our experimental migration project, and shows promising results. For the reason of limited size, we omit the presentation of this experiment. Future work will be concentrated on the validation of this methodology.

References

1. Houari A. Sahraoui, Hakim Lounis, Walcelio Melo, Hafedh Mili.: A Concept Formation Based Approach to Object Identification in Procedural Code. Automated Software Engineering Journal (1999).
2. The Workflow Management Coalition Specification, Workflow Management coalition Work Group, Oct. (2001)
3. Muller, H, Jahnke, J, Smith, D., Storey, M, Tilley, S., Wong, K.: Reverse Engineering: A Roadmap. The Future of Software Engineering. New York, NY: ACM, (2000) 47-60.