Performance Evaluation for Software Migration

Issam Al-Azzoni
INRIA, France
Issam.Al-Azzoni@imag.fr

Lei Zhang and Douglas G. Down
Department of Computing and Software
McMaster University
Hamilton, Ontario
zhangl64@mcmaster.ca,
downd@mcmaster.ca

ABSTRACT

Advances in technology and economical pressure have forced many organizations to consider the migration of their legacy systems to newer platforms. Legacy systems typically provide mission critical services vital for an organization's business needs. These systems are usually very large and highly complex with little or no documentation. Furthermore, fewer people can understand and maintain these systems. While several techniques exist to verify the functionality of the migrated system, the literature is still lacking methods to effectively assess the performance impact of software migration. In this paper, we propose a new method designed specifically to address performance evaluation in software migration projects. The new method uses simple models and incorporates techniques for model validation and resource demand mapping for performance evaluation and capacity planning.

Categories and Subject Descriptors

D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement—*version control*; D.2.8 [**Software Engineering**]: Metrics—*performance measures*

General Terms

Measurement, Performance

Keywords

Software migration, performance models, model validation, mean value analysis

1. INTRODUCTION

Legacy systems continue to play a significant role in managing today's information systems. Many large organizations are using legacy systems to provide mission critical services. Most of the world's business data is still processed by legacy mainframe systems.

Many legacy systems are becoming out-of-date. After many years of evolution, these systems are also becoming hard to understand and maintain. The high cost of maintenance is compounded by the fact that there is a critical shortage of skilled developers who understand and thus can maintain these legacy systems. However, since these systems are vital for an organization's business needs, they can not simply be discontinued. In light of these factors, many organizations have opted for legacy system migration. Here, selected parts of the legacy system are migrated to modern platforms. In the migration of a legacy system, one or more of the following components are converted into a newer, more modern technology: the hardware platforms, the operating system, languages, data and databases. For example, data can be migrated from a flat file system to a database management system or from one database to another. Another example is the migration of a procedural system to an object-oriented platform. Several tools and strategies to assist the migration of legacy systems are presented in De Lucia *et al.* [10], Sahraoui [17], Teppe [18], and Zou [21].

In the migration of a legacy system, the application logic is preserved. Thus, the target system, *i.e.*, the new system, must provide the same functionalities and services as the legacy system, *i.e.*, the old system. While there are several promising approaches for verifying the functional equivalence of the legacy and the target systems, the literature is still lacking effective methods for verifying the preservation of performance properties. Our work addresses this problem by proposing a new method for verifying that migration to a new system does not disrupt the performance properties of the system and that the performance of the target system is better or at least similar to that of the legacy system.

Legacy systems are usually very large (sometimes with more than one million lines of code). Furthermore, these systems are highly complex with little or no documentation. These factors make performance evaluation for such systems a complex task. Stringent deadlines are also typical in migration projects. Furthermore, we note that performance evaluation for legacy systems must consider two systems: the legacy and the target systems. Traditional capacity planning and performance evaluation methods do not address these particularities for legacy system migration.

To the best of our knowledge, CAPPLES[1] is the only method developed specifically for the performance evaluation of target systems during the migration of legacy systems (da Silva *et al.* [7, 8, 9]). Although the methodology

---

[1]CAPPLES stands for CApacity Planning and Performance analysis method for the migration of LEgacy Systems

of CAPPLES has been shown to be effective in addressing several issues of performance evaluation during system migration, we believe that CAPPLES may suffer from several limitations (these are discussed in Section 2).

To address the limitations of CAPPLES, we introduce a new method named PELE, which stands for Performance Evaluation method for LEgacy systems. In PELE, a system is modeled as a queueing network which is solved using Mean Value Analysis (MVA) (Reiser and Lavenberg [16]). In this paper, we show that PELE is an effective method for performance evaluation during system migration. We apply PELE to evaluate performance in the migration of a standard web application.

The organization of the paper is as follows. Section 2 gives a brief summary of CAPPLES and discusses potential limitations. Section 3 presents PELE. In Section 4, we discuss the application of PELE in the migration of a web application. Section 5 gives an overview of the related literature.

2. THE CAPPLES METHOD

CAPPLES is the first method developed specifically to address the particularities of performance evaluation during the migration of a legacy system [8]. In a migration project, the CAPPLES method aims to predict the performance of the target system before it becomes fully operational. The assumption made is that the target system is already developed but is not in production yet. The method is based on simulation models.

The method outlines several novel strategies. One example is a strategy for the characterization of a synthetic workload for the target system. Another novel idea of CAPPLES is the concept of a service. Since a given transaction in the legacy system may not directly have a corresponding transaction in the target system, services are used to establish a mapping for the purpose of identifying the corresponding transactions between the legacy and the target systems. Services are abstractions of their subtasks (*i.e.*, transactions) where actions and user interactions occur [9].

CAPPLES is composed of nine steps which are summarized below. For a detailed discussion, see [7, 8, 9].

1. **Specification of the measuring time window.** The measuring time window corresponds to the on-line time window which is defined as a period that is dominated by the execution of on-line transactions (Menascé *et al.* [14]). Usually, the on-line time window corresponds to an organization's working hours.

2. **Measurement of the legacy system and specification of the simulation time window.** Once the measuring time window is defined, the legacy system must be monitored. CAPPLES only requires the frequency of each transaction during the measuring time window. These measurements help the identification of the legacy system peak hour (or fraction of the peak hour). This peak hour is the simulation time window that will be used in Steps 7 and 9.

3. **Identification of the relevant services in the legacy system.** The purpose of this step is to map the set of transactions obtained during the measuring time window into their related services. These services are subsequently used to characterize the synthetic workload for the simulation of the target system. In this step, there is no need to include all transactions. Instead, only the most executed on-line transactions are considered.

4. **Mapping of the relevant services from the legacy system to the target one.** In CAPPLES, the resource demands of each service are provided by the target system. Therefore, it is required to map the on-line transactions from the legacy system to the target system. Since it is difficult to recognize the correct equivalence between the transactions in both systems, the mapping is carried out indirectly through on-line services. This is possible since these services are high level abstractions of the real system's transactions.

5. **Generation of a synthetic workload for the target system.** The resource demands should be measured in the target system to complete the synthetic workload characterization.

6. **Modeling the target system.** A simulation model is built based on the synthetic workload constructed in the earlier steps. The model must include all services identified during the specification of the simulation time window.

7. **Calibration and validation of the target system model.** A simulation model is considered validated if the mean response time of the modeled services corresponds to the measured mean response time of the same services in the target system. Note that this step requires measuring the response times of the on-line services on the target system.

8. **Workload prediction.** Workload forecast techniques [14] can be used to condition the workload in order to reflect the moment that the target system will be operational.

9. **Target system performance prediction.** The predicted workload is submitted to the validated simulation model generating the required information concerning the behaviour of the target system.

We believe that CAPPLES potentially suffers from three main limitations:

1. In CAPPLES, services are mapped from the legacy system to the target one. The resource demands are subsequently measured in the target system to complete the synthetic workload characterization. Thus, the target system needs to be developed before the performance evaluation step in order to collect the resource demand measurements. A better approach would be to measure the resource demands for services on the legacy system and map the measured resource demands onto the target system. This enables the performance evaluation of the target system before it is fully developed.

2. CAPPLES is designed primarily to predict the performance of the target system. In particular, it does not model the legacy system. Modeling the legacy system can be helpful in the validation process. Furthermore, by doing so, it becomes possible to predict performance before the target system becomes operational. Also,

one can use the model of the legacy system as a basis for performance comparison between the legacy and the target systems. In CAPPLES, there is no performance comparison between the legacy and the target systems.

3. CAPPLES uses simulation models which are often too expensive to develop, validate, and run. Furthermore, developing a simulation model can be time-consuming. This can be problematic given the tight deadlines typically present in a migration project. Also, developing a simulation model requires detailed information about the behaviour of the transactions as well as distributions which may be difficult to obtain.

3. THE PELE METHOD

In PELE, a system is modeled as a queueing network. Input parameters of the model are obtained by analyzing the performance measurements collected from direct observation of the system. Typical input parameters include resource demands and workload intensity (arrival rates). The resource demands, for example, can be obtained by measuring the utilization of each resource (for example, the CPU, disks, etc.), while the workload intensity is obtained by analyzing the transaction log. The parameterized model is solved using an MVA-based solution technique to obtain the mean response time. The model can then be validated by comparing the mean response time calculated by the model to the average response time measured on the actual system. Also, the model can be used to predict the performance of the system under different scenarios and to compare the performance of different systems, *e.g.*, the legacy and the target systems.

PELE is composed of eight steps:

1. **Characterization of the legacy system's workload.** In a migration project, an organization supplies a transaction log characterizing the workload of the legacy system. The workload consists of all transactions processed by the legacy system during an observation interval. It must be representative of the actual system's workload. A discussion on how to construct a representative workload can be found in Weyuker and Vokolos [19].

2. **Measurement of the legacy system.** Two kinds of parameters are derived from the legacy system's workload characterization of Step 1: intensity parameters and resource demand parameters (see Menascé *et al.* [15]). The intensity parameters provide a measure on the load placed on the system. The resource demand parameters provide a measure on the total amount of service time required by a transaction at a given resource. The measurement data presented in the transaction log are transformed into input parameters for the model in Step 4.

3. **Identification of the relevant services in the legacy system.** Since a transaction in the legacy system may not have a corresponding one in the target system and to enable the performance comparison between both systems, it is necessary to identify the relevant services (*i.e.*, the conventional actions performed by the legacy and the target systems). This step is identical to Step 3 in CAPPLES.

4. **Constructing a model for the legacy system. The model is analyzed and validated.** At this step, the resource demands and the intensity parameters of the relevant services in the legacy system are ready. An appropriate queueing model (*e.g.*, open, closed, single class, or multiclass) is used to develop a performance model of the legacy system. The parameterized model is solved using an MVA-based solution technique to obtain the mean response time. The model is then validated by comparing the mean response time calculated by the model to the average response time measured on the legacy system. As a rule of thumb, response times within 20% are considered acceptable [15].

5. **Mapping of the resource demands to parameterize the model for the target system.** In this step, the model of the target system is parameterized. This is done by mapping the resource demands of the legacy system services onto the target system. For example, consider a system in which data is migrated from a flat system into a relational database. The use of the new database system will impact how the resources are utilized. Thus, the new resource demands need to be computed. If the target system is operational, then the resource demands can be measured in the target system. Otherwise, the resource demands should be estimated and this can be done by considering the legacy system. For instance, consider the database migration example. The use of the relational database can be shown using empirical data to speed up the query services, for example, by a factor of half. Note that if there are new resources, these resources need to be added to the model as well as the corresponding resource demands.

6. **Analyzing and validating the model of the target system.** The parameterized model for the target system is solved using an MVA-based solution technique to obtain the mean response time. The model is then validated by comparing the mean response time calculated by the model to the average response time measured on the target system (if it is operational).

7. **Comparing the performance of the legacy and the target systems.** The mean response time calculated using the model of the legacy system is compared with that of the target system model. Ideally, the mean response time for the target system should be smaller.

8. **Workload prediction and target system performance prediction.** This step is identical to Steps 8 and 9 of CAPPLES. The target system model is used to predict the performance of the target system under different scenarios.

Figures 1 and 2 outline the major steps in PELE for the two cases in system migration: operational and non-operational target systems. As the figures show, the target system model can be validated depending on whether it is operational or not. In both cases, the resource demand mapping step is carried out to parametrize the model for the target system.

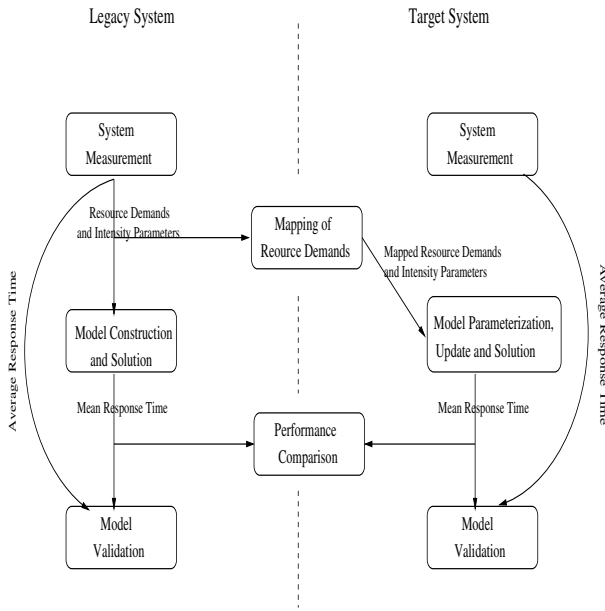The following outlines how PELE addresses the limitations of CAPPLES:

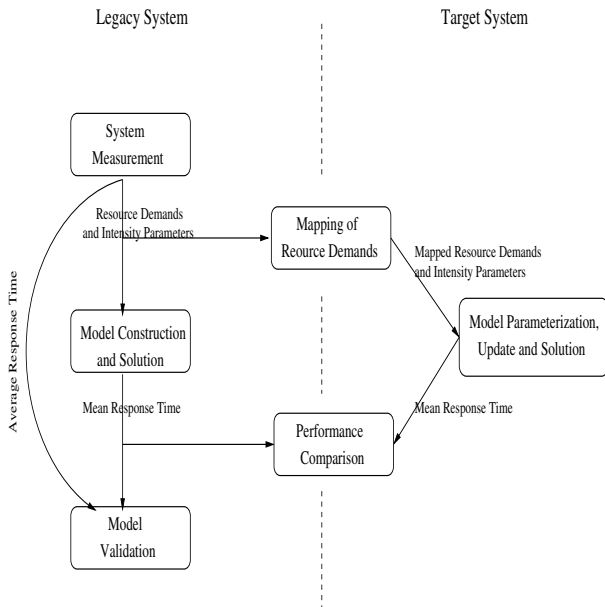**Figure 1: PELE steps for an operational target system**

Figure 2: PELE steps for non-operational target system

1. In PELE, resource demands for services on the legacy system are mapped onto the target system. This is done after modeling and validating the model of the legacy system. We note that using a queueing network model simplifies the mapping. In CAPPLES, the mapping is not straightforward since simulation is used and hence the distributions need to be mapped. Mapping averages (in the case of PELE) is easier to do than mapping distributions (in the case of CAPPLES).

2. In PELE, models for both the legacy and the target systems are constructed and validated. This enables performance comparison between both systems.

3. PELE uses MVA and does not require detailed simulation models. MVA is a simple recursion and several MVA-based solution packages are available (see Bolch *et al.* [6]). MVA does not require detailed information on the distributions. Also, by using MVA, generic tools can be developed for the performance evaluation of legacy systems.

4. CASE STUDY

In this section, we provide a case study in an experimental 2-tiered system. We use TPC Benchmark W (TPC-W) [2] as the workload generator to simulate customers browsing an e-commerce website. TPC-W is a transactional web benchmark. It has 14 Web interactions which simulate different behaviours of a Business-to-Consumer (B2C) website. In the experiments, we choose a Java Implementation of TPC-W from the PHARM group [1].

According to the TPC-W Specification in [2], the number of users or consumers is represented by Emulated Browsers (EB) in TPC-W, and this number is constant throughout an experiment. TPC-W also allows the configuration of the experimental environment through several input parameters. For example, Think Time (TT) is used to refer to the interval between two requests, and the number of customers and items in the database can also be set.

We built an experimental environment on a Dell desktop computer equipped with two Intel Pentium 4 3.00GHz processors, 512MB RAM, and one 80GB harddisk with speed of 7200RPM. For the Web Server, we use Tomcat 7.0.2 [3], and three different versions of MySQL as the Database Server: versions 4.1.22, 5.0.90, and 5.5.5 [4].

We model the system as a closed queueing network and solve it using a single class MVA model with two resources (CPU and Disk). The service demand of each resource is computed using the Service Demand Law [15]:

$$\text{service demand} = \frac{\text{utilization}}{\text{throughput}}.$$

In our experiments, we consider two scenarios of system migration among different versions of MySQL servers:

1. Migration from MySQL 4.1.22 to MySQL 5.0.90

2. Migration from MySQL 5.0.90 to MySQL 5.5.5

We would like to validate the MVA models first, and then our methodology. For the MVA model validation, we need to set the experimental environment: we modify TPC-W and configure the number of users (EBs) to 50, think time to 20 seconds, and generate the database to contain 28,800

Table 1: Utilization data collection

U for utilization

Migration Steps	U (%CPU)	U (%Disk)
MySQL 4.1.22	5.52	83.77
MySQL 5.0.90	5.28	75.91
MySQL 5.5.5	5.53	99.33

Table 2: Predicted and measured data collection

R for response time which is measured by seconds.

Migration Steps	R (MVA)	R (6000 seconds)	R (30000 seconds)
MySQL 4.1.22	1.87	1.92	2.39
MySQL 5.0.90	1.28	1.49	1.12
MySQL 5.5.5	4.95	4.16	4.62

customers and 1,000 items. In addition, we run TPC-W for 6000 seconds after 120 seconds of ramp-up.

We apply PELE as follows:

1. Run TPC-W with MySQL 4.1.22 to measure the average response time while also measuring CPU and Disk utilizations.

2. Use MVA to calculate the mean response time, and compare it with the measured average response time.

3. Repeat Steps 1 and 2 with MySQL 5.0.90, and then compare the performance of MySQL 5.0.90 with that of MySQL 4.1.22.

4. Repeat Steps 1 and 2 with MySQL 5.5.5, and draw a comparison between the performance of MySQL 5.5.5 and that of MySQL 5.0.90.

The CPU and Disk utilizations measured in our experiments are shown in Table 1. The mean and average measured response times are shown in Table 2. There are two observations:

1. The MVA-based mean response time R(MVA) and the measured average response time R(6000 seconds) are quite close, so that our model is validated.

2. TPC-W achieves best performance with MySQL 5.0.90. Thus, PELE makes a "positive decision" for the migration from MySQL 4.1.22 to MySQL 5.0.90 and a "negative decision" for the migration from MySQL 5.0.90 to MySQL 5.5.5.

To validate the methodology, we change the measuring time from 6000 seconds to 30000 seconds to observe the long term performance. We run TPC-W with MySQL 4.1.22, 5.0.90, and 5.5.5 respectively to measure the average response time over a long period of time. The average response time under each MySQL version, R(30000 seconds), is shown in Table 2. TPC-W under MySQL 5.0.90 still has the best performance which means that PELE decisions are validated.

5. LITERATURE REVIEW

BMM is another method for the performance evaluation of legacy systems (Jin *et al.* [13]). BMM combines benchmarking, production system monitoring, and performance modeling to predict the performance of a legacy system when subjected to dramatic increases in workload. In BMM, the legacy system is modeled using a performance model such as Layered Queueing Networks (LQNs) (Franks and Woodside [11]).

While BMM outlines a systematic approach to the performance evaluation of legacy systems that are subject to exorbitant load growth, it does not address the particularities of performance evaluation during the migration of a legacy system. For example, BMM is missing the mapping step of CAPPLES in which services in the legacy system are mapped to their corresponding services on the target system. Also, BMM requires constructing and analyzing a performance model which requires detailed information on the transaction behaviour and underlying distributions.

Several methods for performance evaluation and prediction have been integrated with the software development process (see Balsamo *et al.* [5] and Woodside *et al.* [20] for a survey). The majority of these methods require creating a performance model early in the development cycle. The performance models are used to provide quantitative results in order to adjust the architecture and design with the purpose of meeting performance requirements. These methods aim to support performance engineering early in the development process. However, when applied to a legacy system, it is of interest to explore how these methods can be used to address the particularities of performance evaluation during system migration.

In Jiang *et al.* [12], the authors present a method which automatically analyzes the execution logs of a load test for performance problems. Given the execution logs of two different runs, the method flags scenarios which follow a different response time distribution. The two logs are generated by performing load tests on different versions of the same system under similar workload. While the method can be adapted for the performance evaluation of legacy systems, it suffers from several limitations. First, the method requires detailed execution logs that are typically missing for legacy systems. Second, the method requires that the target system be operational in order to generate the execution logs. Finally, the method requires that the same transactions are executed in the legacy and the target systems. Since this may not be true in migration, it is of interest to study how the method is impacted when services are mapped from the legacy system to the target system.

6. DISCUSSION

Since PELE applies MVA, the decision to migrate is based solely on the average performance (response times). In particular, PELE may fail to detect those infrequent transactions which have problematic performance on the target system. Similarly, PELE may fail to detect slowly evolving performance degradation due to issues such as memory leaks. We note, however, that PELE may be the only option to do when evaluating performance for a legacy system. Furthermore, while PELE can be appended with tools to detect the special cases above, we believe that considering average performance should be the first step in performance eval-

uation. One can apply PELE first in evaluating a legacy system, then apply other techniques on the target system specifically designed to detect memory leaks or to find those infrequent transactions having problematic worst-case performance.

7. ACKNOWLEDGMENTS

8. REFERENCES

[1] TPC-W Java Implementation, originated of PHARM at the University of Wisconsin - Madison. "http://mitglied.multimania.de/jankiefer/tpcw/", last accessed September 3, 2010.

[2] TPC Corporation, TPC-W official website, "http://www.tpc.org/tpcw/", last accessed September 3, 2010.

[3] Apache Software Foundation, Apache Tomcat official website. "http://tomcat.apache.org/", last accessed September 4, 2010.

[4] MySQL AB, Sun Microsystems, MySQL Community Edition. "http://www.mysql.com/downloads/mysql/", last accessed September 4, 2010.

[5] S. Balsamo, A. D. Marco, P. Inverardi, and M. Simeoni. Model-based performance prediction in software development: A survey. *IEEE Transactions on Software Engineering*, 30(5):295–310, 2004.

[6] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. John Wiley & Sons, second edition, 2006.

[7] P. P. da Silva, A. H. F. Laender, and P. B. Golgher. A simulation model for the performance evaluation when migrating legacy systems. In *Proceedings of the Conference on Software Maintenance and Reengineering*, pages 210–215, 2001.

[8] P. P. da Silva, A. H. F. Laender, R. S. Resende, and P. B. Golgher. CAPPLES - A capacity planning and performance analysis method for the migration of legacy systems. In *Proceedings of the Workshops on Evolution and Change in Data Management, Reverse Engineering in Information Systems, and the World Wide Web and Conceptual Modeling*, pages 198–212, 1999.

[9] P. P. da Silva, A. H. F. Laender, R. S. Resende, and P. B. Golgher. Characterizing a synthetic workload for performance evaluation during the migration of a legacy system. In *Proceedings of the Conference on Software Maintenance and Reengineering*, pages 173–181, 2000.

[10] A. De Lucia, R. Francese, G. Scanniello, and G. Tortora. Developing legacy system migration methods and tools for technology transfer. *Software: Practice and Experience*, 38(13):1333–1364, 2008.

[11] G. Franks and M. Woodside. Multiclass multiservers with deferred operations in layered queueing networks, with software system applications. In *Proceedings of the International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, pages 239–248, 2004.

[12] Z. M. Jiang, A. E. Hassan, G. Hamann, and P. Flora. Automated performance analysis of load tests. In *Proceedings of the International Conference on Software Maintenance*, pages 125–134, 2009.

[13] Y. Jin, A. Tang, J. Han, and Y. Liu. Performance evaluation and prediction for legacy information systems. In *Proceedings of the International Conference on Software Engineering*, pages 540–549, 2007.

[14] D. A. Menascé, V. A. F. Almeida, and L. W. Dowdy. *Capacity Planning and Performance Modeling: from Mainframes to Client-Server Systems*. Prentice-Hall, Upper Saddle River, NJ, USA, 1994.

[15] D. A. Menascé, V. A. F. Almeida, and L. W. Dowdy. *Performance by Design: Computer Capacity Planning By Example*. Prentice-Hall, Upper Saddle River, NJ, USA, 2004.

[16] M. Reiser and S. S. Lavenberg. Mean-value analysis of closed multichain queuing networks. *Journal of the ACM*, 27(2):313–322, 1980.

[17] H. Sahraoui. Coping with legacy system migration complexity. In *Proceedings of the International Conference on Engineering of Complex Computer Systems*, pages 600–609, 2005.

[18] W. Teppe. The ARNO project: Challenges and experiences in a large-scale industrial software migration project. In *Proceedings of the Conference on Software Maintenance and Reengineering*, pages 149–158, 2009.

[19] E. J. Weyuker and F. I. Vokolos. Experience with performance testing of software systems: Issues, an approach, and case study. *IEEE Transactions on Software Engineering*, 26(12):1147–1156, 2000.

[20] M. Woodside, G. Franks, and D. C. Petriu. The future of software performance engineering. In *Proceedings of the Future of Software Engineering Conference*, pages 171–187, 2007.

[21] Y. Zou. Quality driven software migration of procedural code to object-oriented design. In *Proceedings of the International Conference on Software Maintenance*, pages 709–713, 2005.