A Quality Control Center for Software Migration

Johannes Meier, Dilshodbek Kuryazov, Jan Jelschen, Andreas Winter
Carl von Ossietzky Universität, Oldenburg, Germany
{meier,kuryazov,jelschen,winter}@se.uni-oldenburg.de

Abstract

Software Migration, as transformation of legacy software into new software implemented in a different programming language, is motivated by selected quality goals like higher maintainability of the migrated software. To check which quality goals were reached, the inner quality of legacy and migrated software has to be determined and compared.

To investigate the inner quality of migrated software, this paper introduces a Software Migration Quality Control Center (QCC), which allows comparing the quality of legacy and migrated software systems. To this end, this paper discusses requirements for a QCC and their implementation in the Q-MIG project.

1 Motivation

Software migration is a means to translate existing legacy systems into another programming language without changing the functionality [1]. The motivation for such migrations is a system in a new programming language reaching quality goals like allowing better maintenance and improving functionality without redeveloping the complete software.

In order to achieve successful migrations by anticipating the impact of migration on the software quality, the quality of legacy and new software systems have to be observed and compared, so that the expected inner quality in terms of evolvability can be estimated.

Q-MIG[1] (Quality-driven software MIGration) aims at creating strategies and tools to investigate quality issues of software migrations from COBOL to Java [2]. To investigate the inner quality systematically, a toolchain for COBOL to Java migration created by *pro et con* [3] is combined with a Quality Control Center (QCC) with the following use cases:

1. *Compare quality of legacy and migrated software.* This use case compares the internal quality of both legacy and migrated software and helps *customer consultants* to check the final quality of the migrated software.
2. *Compare quality of migrated software using different migration tools.* This use case compares the quality of systems migrated using different tools with the goal of determining the quality of those tools. This helps *researchers* to rate and improve existing migration tools.
3. *Predict quality of migrated software before executing the migration.* This use case helps customer consultants to discuss the final quality of the migrated software before migration execution by only analyzing the legacy software. Doing this for different migration tools, allows selecting of the migration tool which produces the best quality.

The paper is structured as follows: Section 2 sketches the approach and identifies requirements for the QCC in Section 3. Applications are demonstrated in Section 4. Section 5 draws conclusions.

2 Requirements

To identify and define the inner quality of the migration results, the quality *characteristics maintainability* and *transferability* are selected from the ISO quality standard [4]. Several *metrics* relevant to the selected characteristics are chosen to directly, objectively and automatically measure the software quality [2]. The basic idea behind predicting the software quality is to estimate and use quality models, using automatically calculated metrics as independent values, and expert rated characteristics as dependent values. The measured metric values are used to predict the new values of the characteristics. The new quality characteristic values of the software can be used to satisfy the goals and use cases of Q-MIG in the QCC, for which some requirements are identified:

- *Quality Measurement (RQ1).* Metrics have to be calculated on COBOL and Java, and software experts have to rate characteristics for COBOL and Java systems.
- *Data Management (RQ2).* The QCC has to support data management tasks like importing data into the QCC, and creating detailed traceability links between legacy and migrated software.
- *Quality Analysis (RQ3).* The QCC enable quality comparisons of legacy COBOL and migrated Java systems for Use Cases 1 and 2.
- *Quality Prediction (RQ4).* The QCC has to support training quality models and predicting new characteristic values for new target systems reusing these models (Use Case 3).

3 Quality Control Center

In the upper part, Figure 1 depicts the used migration toolchain created by *pro et con*: *CobolFE* produces COBOL abstract syntax trees (ASTs) at measurement point $M2$, the *COBOL Transformator* transforms COBOL ASTs to Java ASTs ($M3$), and *JGen* produces Java source code at $M4$. The lower part dis-

[1]Q-MIG is a joint venture of *pro et con Innovative Informatikanwendungen GmbH, Chemnitz* and *Carl von Ossietzky University's Software Engineering Group.* It is funded by Central Innovation Program SME of the German Federal Ministry of Economics and Technology – BMWi (KF3182501KM3).

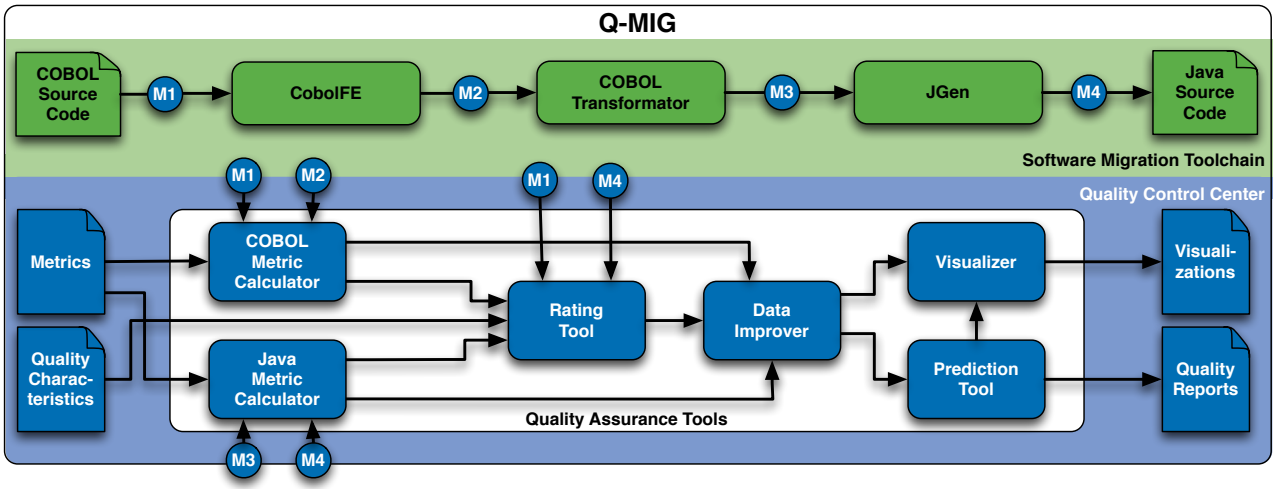

Figure 1: COBOL-Java-migration toolchain created by *pro et con* with tools of the QCC.

plays the main components of the QCC, which fulfills the requirements presented in Section 2. The *COBOL Metric Calculator* and the *Java Metric Calculator* calculate a given set of metrics on COBOL ($M1, M2$) and Java ($M3, M4$) source code and AST, respectively. Internally, the Java MetricCalculator uses GReQL graph queries [5] for metric implementation. The *Rating Tool* is created to support quality experts rating characteristics for COBOL and Java structures like classes and methods extracted by the metric calculators (RQ1).

The component *Data Improver* fulfills the data management tasks such as creating traceability links between the legacy COBOL and migrated Java software for later comparisons, and stores all data about the software systems and their characteristics and metric values in a central repository using TGraphs (RQ2).

The *Visualizer* creates different types of visualizations, such as HTML reports and line, bar, and scatter charts allowing to compare and analyze the quality of software systems before and after migration, as well as different migration tools (RQ3).

The *Prediction Tool* provides different machine learning algorithms including *artificial neural networks* and *multiple linear regression* for training and predicting the quality characteristics. It allows the estimation of prediction models by training samples of metrics and expert-rated characteristics. Later on, these prediction models are used to predict quality characteristics for future migrations. Quality models can be trained once and used multiple times (RQ4).

4 Application

The QCC is realized in Q-MIG and applied to three real-world examples provided by *pro et con*. After having measurement results of the metrics for COBOL (*COBOL Metric Calculator*) and Java (*Java Metric Calculator*) systems, and ratings of the software quality (*Rating Tool*), the *Visualizer* creates HTML reports, enabling both researchers and customer consultants to compare the quality of legacy and new software (Use Case 1). Additionally, further graphics like line and bar charts visualize detailed quality aspects.

The HTML reports display views for comparing different migration tools by showing Java systems migrated from the same COBOL system using different migration tools (Use Case 2). Some more visualizations are currently under development.

In order to predict the quality of prospective Java software which will be the result of future migrations, the *Prediction Tool* operates on existing quality data from previous executed migrations (Use Case 3). These prediction results help customer consultants to asses the viability of migrating the legacy COBOL software.

The QCC serves as a knowledge base to improve the quality of predictions, and will be enriched by the statistical data about further migrations which helps both researchers and customer consultants for effective quality prognosis.

5 Conclusion

This paper discussed a Quality Control Center (QCC) with several use cases for software migration quality analysis and prognosis. The QCC is realized in Q-MIG and applied to real-world applications. The QCC is valuable for researchers and customer consultants for analyzing and comparing quality of migrated software systems and making future decisions based on that knowledge. In the long term, the QCC gathers statistical data about the quality of different migrated software systems and makes prognoses on how the quality changes in prospective migrations.

References

[1] H. M. Sneed, E. Wolf, and H. Heilmann, *Softwaremigration in der Praxis: Übertragung alter Softwaresysteme in eine moderne Umgebung.* Heidelberg: Dpunkt.Verlag GmbH, 2010.

[2] G. Pandey, J. Jelschen, D. Kuryazov, and A. Winter, "Quality Measurement Scenarios in Software Migration," in *Softwaretechnik Trends*, vol. 34, no. 2. Gesellschaft für Informatik, 2014, pp. 54–55.

[3] U. Erdmenger and D. Uhlig, "Ein Translator für die COBOL-Java-Migration," *Softwaretechnik-Trends*, vol. 31, no. 2, 2011.

[4] ISO/IEC, "ISO/IEC 25010 - Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models," Tech. Rep., 2010.

[5] J. Ebert and D. Bildhauer, "Reverse engineering using graph queries," in *Graph Transformations and Model-Driven Engineering*, ser. Lecture Notes in Computer Science. Springer, 2010.