Towards Quality Models in Software Migration

Gaurav Pandey, Jan Jelschen, Andreas Winter
Carl von Ossietzky Universität, Oldenburg, Germany
{pandey,jelschen,winter}@se.uni-oldenburg.de

Abstract

To preserve legacy systems in continuous software development and evolution, next to redevelopment, they can be migrated to new environments and technologies. Deciding on evolution and migration strategies early, requires predicting the quality of the migrated software systems depending on applied tools. There is a need for comparable measures, estimating the inner software quality of legacy and target systems.

Technically, software migration tools use a transformation-based toolchain using model-driven technologies. Therefore, quality measurement can be based on the underlying models representing input and output of applied migration tools.

This paper proposes a Software Migration Quality Model in order to provide support for quality-driven tailoring of utilized model-driven migration tools.

1 Motivation

Software Migration comes across as an important technique to evolve legacy systems into new environments and technologies without changing the system's functionality [5]. It continues the modernization, operation and development of software without dealing with the risk and cost of a complete redevelopment [9]. Each migration project requires an especially tailored toolchain [3], aiming at preferably automatically transferring legacy to target. Moreover, deciding between software migration and redevelopment as well as choosing the components of the migration toolchain, requires reliable predictions regarding quality of the migrated software. To achieve this, there is a need to measure and compare the quality of the legacy software, migrated software and the intermediate software stages.

Monitoring changes in software-quality during software development is supported by various incremental approaches: e.g. Teamscale [4] and SonarQube [10]. These approaches are restricted to a single implementation platform. Since language based software migrations, e.g. migrating from COBOL to Java, deal at least with two different development platforms, cross platform monitoring is needed. This challenges for providing metrics, which are applicable in both environments allowing comparison of quality issues across platforms.

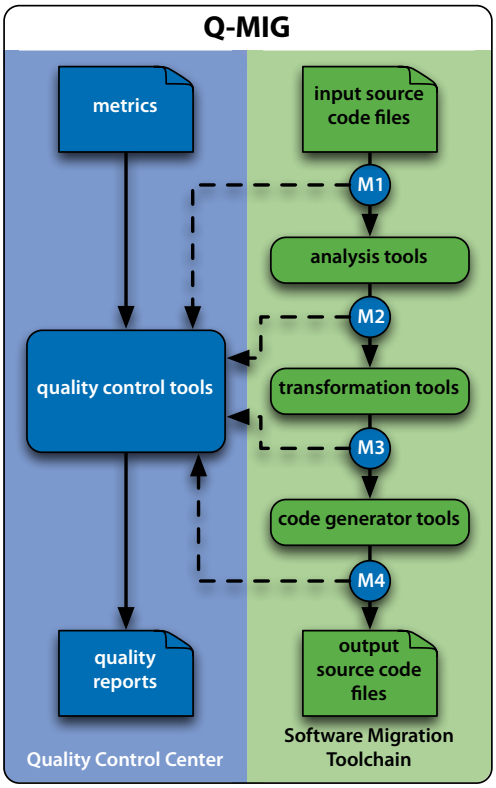

Figure 1: The Q-MIG integrated toolchain.

The Q-MIG-project[1] (Quality-driven software MIGration) aims at monitoring changes in software quality during migration [7]. The objectives to be achieved by quality monitoring are quality prognosis and selection of migration toolchain components. This enables supporting quality driven decisions on migration strategies and tooling.

2 Q-MIG

Q-MIG combines a software migration toolchain [2] with a quality control center (cf. Figure 1). It allows for software quality management during the migration process including quality assessment during project planning and tooling. The monitoring points *(M1-M4)* allow to measure, monitor and compare the quality in the software toolchain. Here, migration and

[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).*

quality tools are integrated. The calculation of quality at the intermediate monitoring points helps in taking decisions regarding the use of individual tools. The *Goal/Question/Metric* [1] approach is followed to derive the metrics, where the Goal is quality comparison and measurement and the Question is the set of relevant quality criteria for the project.

Cross-platform quality comparison is achieved by calculating the same metric at monitoring points. Here metrics like *Lines of Code*, *McCabe*, *Number of GOTOs*, *Comments Percentage* and *Clones Percentage* are used. Moreover, calculating metrics that represent the same quality but have different implementations at the monitoring points, also enables quality comparison. E.g. in a COBOL to Java migration, *Cohesion* (the degree of independence between the building blocks of a system) can be calculated considering classes as the building blocks for Java, whereas procedure-divisions can be considered as building blocks for COBOL.

Analyzing the quality of migrated software with respect to tools used, helps in determining the combination of components in a migration toolchain. As the metrics in this case need to be applicable only for the migrated system, Java specific metrics are utilized like: *Number of Classes*, *Average Number of Subclasses*, *Attribute Hiding Factor*, *Method Hiding Factor*, *Average Number of Methods per Class* and *Inheritance Tree Depth*.

The software migration toolchain in Figure 1 can technically be viewed as a combination of model-driven tools. As model-driven environments can handle code and model in the same fashion, we define the internal representation of the two as a *codel*. At the monitoring points these codels are available for picking their quality prior and after each migration step.

3 Software Migration Quality Model

As the migration toolchain is already model-driven, model-based approaches to measure the quality can be applied. Measurement of the quality of codels can be based on querying [8]. So, quality measurement and monitoring in software migration can utilize the already existing model-driven querying tools to calculate and to compare the quality of succeeding codels.

Comparing the succeeding codels requires to align metrics, codels and the applied migration tools, which can be viewed as model transformations. To this end, the quality model for software migrations (*Q-MIG-Model*) has been derived. It takes into account *components* providing the required transformation services to transform one codel to another in a migration. These migration steps can be viewed as services realized by *components* according to the service-based tool integration approach SENSEI [6].

The *Q-MIG-Model* also aligns the originating and resulting *codels* to migration projects specific *Quality Models* which summarize all *metrics* defining the project specific quality issues. For each *codel* all relevant metrics-values are stored. These *values* will be monitored during migration and knowledge on changing their values during migration will help to predict the quality of migration results.

Metrics are calculated by applying *queries* to *codels* resulting in the appropriate *values*. Since the *codels* conform to certain language definitions (either grammars or meta models) defining the codel's abstract syntax, the *queries* also have to conform to the language definitions.

4 Summary

This paper presented the first steps in the Q-MIG project in providing a quality-driven support to software migration. The strongly model-driven foundation of Q-MIG was given by referring to Q-MIG's Software Migration Quality Model.

Next steps in Q-MIG deal with specifying relevant metrics in COBOL-to-Java migration projects and applying these values to all codels in a given migration tool chain to provide an initial migration monitoring. Then the quality measurements can be further utilized to derive quality prognosis.

References

[1] V. R. Basili, G. Caldiera, H. D. Rombach. The Goal Question Metric Approach. In *Encyclopedia of Software Engineering*. Wiley, 1994.

[2] C. Becker, U. Kaiser. Test der semantischen Äquivalenz von Translatoren am Beispiel von CoJaC. *Softwaretechnik-Trends*, 32(2), 2012.

[3] J. Borchers. Erfahrungen mit dem Einsatz einer Reengineering Factory in einem großen Umstellungsprojekt. *HMD*, 34(194): 77–94, Mar. 1997.

[4] CQSE GmbH. Teamscale, 2014. http://www.cqse.eu/en/products/teamscale/overview/.

[5] A. Fuhr, A. Winter, U. Erdmenger, T. Horn, U. Kaiser, V. Riediger, W. Teppe. Model-Driven Software Migration — Process Model, Tool Support and Application. In A. D. Ionita, M. Litoiu, G. Lewis, editors, *Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments*. IGI Global, Hershey, PA, USA, 2012.

[6] J. Jelschen. SENSEI: Software Evolution Service Integration. In *CSMR-WCRE Software Evolution Week*, Antwerp, Belgium, 469–472, 2014. IEEE.

[7] J. Jelschen, G. Pandey, A. Winter. Towards Quality-Driven Software Migration. In *Proceedings of the 1st Collaborative Workshop on Evolution and Maintenance of Long-Living Systems, Kiel*, 8–9, 2014.

[8] B. Kullbach, A. Winter. Querying as an Enabling Technology in Software Reengineering. In *3rd European Conference on Software Maintenance and Reengineering*, 42–50. IEEE Computer Society, 1999.

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

[10] SonarSource. SonarQube, 2014. http://www.sonarqube.org.