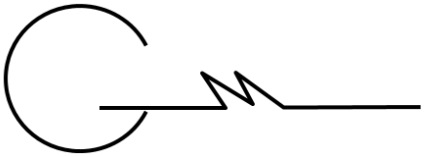
Code Quality



Mark Juras
Great Migrations LLC
June, 2007

Quality of the Migrated Code

A code quality framework helps in planning a migration effort and in tracking its progress. We define two dimensions of code quality: code correctness and standards conformance.

Code Correctness

Correctness is the first dimension of code quality. It is defined by how well-formed the code is relative to formal rules of computer language syntax and semantics and it can be measured objectively and systematically using compilers and traditional functional testing techniques. If the code builds without compiler errors and meets functional and performance requirements, it can be considered correct. Rules of correctness are typically externally imposed: the language platform vendor provides a compiler that tells you if your code is syntactically correct, and the QA team tests the application and tells you if your code is functionally correct.

A platform migration radically changes the rules of correctness: the C# compiler certainly will not accept VB6 code. In general, the new rules for correctness are non-negotiable: the migrated application must at a minimum build with the new compiler and it must still meet its functional requirements.

Table 1. Code Correctness Index

Level 1: Translation Complete	All statements/constructs of the source code translate into target code statements/constructs with no translation errors and the resulting code meets generally accepted standards of well-formed syntax on the target platform. The resulting target projects are well formed and load cleanly in the Visual Studio .NET platform for further development on that platform.
Level 2: Build Complete	All statements/constructs of the target code compile with no compiler errors. In addition, specialized project elements (e.g., Forms) are accessible through their specialized editors provided by the target development environment.
Level 3: Verification Complete	The target application yields correct results on the target platform. The functionality of the target application matches that of the source application. The application is ready for production and further upgrade on the .NET platform.

Standards Conformance

Standards conformance, or simply conformance, is the second dimension of code quality. It measures how well the code meets standards for coding, architecture, and configuration management. Note that standards reflect our best attempts to define coding patterns and practices that improve less-tangible measures for code quality such as maintainability, robustness, scalability, reusability, etc. Thus the manner in which a codebase meets standards correlates with how well it meets the intangible measures of code quality.

Conformance is subjective because there are many possible coding and architecture standards. However, given a specific set of standards, conformance can be measured by simply inspecting the code or with the assistance of properly configured code review tools. Rules of conformance are typically self-imposed: they are defined and enforced by the development team.

A platform migration also changes the rules of conformance: the new platform will offer new options for software construction and it may also take some options away. However, unlike the new rules for correctness, which are strict, the new rules for conformance may be flexible.

Your choices around new standards and conformance will strongly impact the total cost of migration (TCM) and total cost of ownership (TCO) of the migrated application going forward. The challenge is to control both TCM and TCO. Our methodology will help you meet this challenge by helping you balance automated translation with manual work to give you correct code that also meets new standards.

Measuring conformance involves inspecting code to see how well it meets the different aspects of your standards. In a real world migration, the team will select the coding and architecture standards that they feel are most important and then work towards making sure that these are followed in the new codebase.

Achieving Standards Conformance In Migrated Code	Complete conformance to standards and rarely achieved – even in hand written code. However, a platform migration creates a need and opportunity to revisit standards and conformance on a large scale. We work with our clients to define conformance objectives as must-have, should-have, and nice-to-have and then we develop a strategy that delivers all the must-haves, most of the should-haves, and some of the nice-to-haves. The strategy balances tool-based and manual development and also considers the time to market value of the various conformance rules. The should-haves and nice-to-haves that don't make the cut for the initial migration effort may be scheduled for implementation after the application has stabilized on the new platform.

Figure 3 shows that there are multiple ways to migrate an application and that decisions regarding architecture impact the overall effort.

In this illustration, Profile 1 shows that additional up front investment in smarter solutions to issues and target architecture pays off in terms of less effort later on. The smarter solutions usually relate to knowing when old coding patterns should be reworked rather than translated. For example, instead of translating a complex function that uses a series of Win32 calls, the migration team chooses .NET calls that do the same work in a simpler and more standard way. Another example is instead of directly using interop to access COM components from the application code, the team chooses optimized .NET wrapper components that simplify the migration.

Figure 3. Code Quality vs. Effort