
Manual Rewrite or Automatic Transformation

Risks and The benefits

Version 2.0

www.softwaremining.com

1 Manual Rewrite of Legacy code

1.1 Overview

Green field IT projects are not good at meeting budgets and deadlines. This is particularly applicable to larger IT projects – but is equally applicable to medium size projects also:

A £6 billion project to put the medical records of 50m Britons online by the end 2005 is way over budget and has already been postponed by several months. In March 2005, the FBI finally abandoned a \$170m internal IT project, two years after problems with it had first surfaced. The Standish Group, a research firm which produces an influential annual evaluation of IT projects, judged that in 2004 only 29% of such projects “succeeded”, down from 34% in 2002. Cost over-runs averaged 56% of original budgets, and projects on average took 84% more time than originally scheduled.

(source : http://www.economist.com/business/displayStory.cfm?story_id=4065653)

Other famous delayed and over-budgeted large projects:

- Microsoft Windows 95 – delivered 2 years late.
- MI5's computers will be over budget and under-powered
http://www.theregister.co.uk/2005/04/08/mi5_it_budget/
- CSA computer system '£50m over budge -
http://www.theregister.co.uk/2002/08/13/csa_computer_system_163_50m/
- Government scraps £80m computer system -
http://www.theregister.co.uk/2001/02/15/govt_scraps_163_80m_computer/
- EDS suffers US Navy broadside
http://www.theregister.co.uk/2004/10/26/eds_navy_sinks/

1.2 The problem

1.2.1 Management Risk

To manage modern large-scale technical work, the management must know where the project stands, how rapidly the work is being done, and the quality of the products being produced. With non IT development projects (manufacturing, construction, etc), all of this information was more-or-less visible to the manager, while with modern software projects it often is not.

1.2.2 Vague Milestones

Most projects have major milestones such as completion of specifications, design, coding and testing. However, on real software projects, few of these high-level tasks have crisp completion dates. The requirements work generally continues throughout design and even into implementation and test; coding usually starts well before design completion and continues through most of testing.

1.2.3 Requirement Creeps

Requirements also tend to creep and the scope of project is often subject to changes.

If the management attempt to tightly control and disallow the changing requirement – there is a possibility of the finished product will not meet the overall clients requirements - for example see the article “GPs have no faith in £6bn NHS IT programme” http://www.theregister.co.uk/2005/02/08/npfit_gp_lose_confidence/.)

On the other hand – provision for “changes to requirement” often results in project slippage.

1.2.4 Project Slips

All types of projects are affected by slips. In Non-IT projects the manager could usually see these one- and two-day slips and could do something about them. With modern, complex, software systems, the daily schedule slips are largely invisible, and, the managers generally do not see the problem until it is so big that it is obvious. Then, however, it is usually too late to do much about it.

1.2.5 Large Development Teams

Project problems such as slippages are harder to detect with larger teams. Therefore larger teams are generally attributed with substantially increasing project risks. Software Projects with teams of 100 people are renowned for the associated risk.

1.2.6 Long Familiarization Time

In non-IT (or non-technical) projects, once the problems or slippages has been identified, increasing the size of resource force is a viable option to regain balance. In software projects new resources will require a long familiarization time before becoming effective.

2 Additional Problems With Manual Rewrites

2.1 Data-Migration Issues

The implication of a new requirement and design phase – means that migration strategy of the legacy data is typically one of the last project activities. The question of how to fit in the old data into the new structures invariably adds to the problems at the end of the rewrite phase.

2.2 Requirement Specification

The manual re-write projects are not exempt from vagueness in requirement. They have typically insufficient documentation, with the business logic deep in the legacy source code, and the end users vaguely aware of the functionality.

2.3 Change in Business Processes

Often the complete manual rewrite (or replacement by off-the-self products) results in changes in the Business-Processes. These changes often require a major change in the way people work –often resulting productivity losses.

Changes to business process are best managed in small steps, which “manual rewrite” projects are often in-capable of addressing.

A recent enquiry to SoftwareMining stated:

“During a change of IT system, our competitors lost some 25% of their market share due to the underlying change in their Business-Process. They are now finding it difficult to re-capture this market share “.

2.4 Re-Training

Even in the absence of changes to business-Processes, manual rewrite projects often do not aim to reproduce the same paths (screens or processes) to achieve the same business objectives. Therefore *extensive* training for the end-users becomes a necessity. The larger the user-base, the bigger the problem, as each user learns at a different pace.

3 Automatic Transformation

3.1 Overview

Automatic Transformation of legacy applications can only provide a good Return-On-Investment when they full-fill the following requirements:

- The transformation is to new technologies such Java and Relational Data bases. Otherwise – the issues raised by legacy nature of the application will only be delayed for several years.
- The transformation process results in legible and maintainable code – easing further modernizations and enhancements.

3.2 SoftwareMining Automatic Transformation

The primary aim of SoftwareMining's transformation services is the generation of legible and maintainable. Translation is from COBOL to modern languages such as Java and Relational Databases.

The SoftwareMining solutions provide have the following advantages over the manual rewrite projects:

- Managing Risks: the translation projects better controlled criteria: how much original source has been translated, how many tables converted, how many modules tested, etc.
- Introduces concrete and well-defined phases for analysis, transformation, data-migration, testing, production and improvements. Each phase can be fully completed before starting work on the next phase. Each phase only addresses a small number of manageable issues.
- SoftwareMining Business-Rule Extraction modules allow re-documentation of the code.
- Requires no change in Business-Process,
- After the completion of first stage deliverables – the screens will have similar look and feel to the original screens. No retraining required. (Future enhancements will result in gradual changes to screens – providing a easier path for end-user to adapt to each set of enhancements).
- Provide a Data-Migration and schemas strategy compatible with the legacy system. Each new table will comprise of the same columns – with same names.

3.3 Future Enhancements to Systems

- Extracted Business Rules can be reworked into new architectures – such as ILOG - JRules Rules-Engine.
- Architectural enhancements such as SOA or MVC pattern can be retro-fitted.
- Cosmetic enhancements to screens can be achieved from the 'framework' layer. E.g. screen colours, logos, menus can be added thru the framework.