Core System Transformation and Big Data Re-Architecting

Rainer Gimnich
IBM Software Group
Big Data Industry Team, Europe
Wilhelm-Fay-Str. 30-34, D-65936 Frankfurt
gimnich@de.ibm.com

Abstract

This paper presents two major transformation initiatives in many industries today, and it tries to examine them from a reengineering perspective.

Core System Transformation deals with analyzing the existing core software system of an enterprise in order to migrate it into a renewed and more flexible system.

Big Data addresses the Enterprise Architecture by introducing new, data-centric capabilities and integrating them into the existing IT landscape.

Both approaches are rated relevant by most organizations. However, the potential benefits of reengineering techniques are not (yet) assessed and leveraged in most cases.

This paper aims at initiating a discussion of whether and how reengineering experts and technology can (and should) support these endeavors.

1. Core System Transformation

The term 'core system' is used here to denote the central or most-referenced application system in a company. This is often the one (or few) system(s) that would cause the biggest damage if down.

The core system can be a core-banking system or a core-insurance system, depending on the industry. Or it could be the PLM (product lifecycle management) system in manufacturing or the citizen information system in the government industry.

As an example, let us consider Core Banking Transformation (CBT) here [1]. Core banking typically covers the management of accounts, loans, mortgages and payments. In CBT practice, there are three major approaches:

A. Renovate / Migrate: Perform a language (and technology) transformation: e.g. analyze the existing COBOL or 4GL core banking landscape and migrate it into a Java environment.

B. Replace: Select and use an available software product/package for the core functionality and reengineer the "package externals" (interfaces, support modules, etc.) to achieve a working system.

C. Architecturally Transform: Define the target core banking system architecture based on existing and adapted models (process/service/data models). Select suitable additional assets (e.g. reference architecture, products). Develop the missing parts or migrate them from the existing core banking system.

Approaches B and C typically involve the use of service orientation in the architecture (SOA).

The transformation in Approach C typically includes functional extensions to the core banking system (vs. the existing system). So this cannot be a pure reengineering or reverse engineering exercise.

In summary, all three approaches have an inherent "reengineering kernel" in that their sponsors intend to reuse investments in existing applications with significant own development efforts.

And the transformation needs can largely be mapped to workflows of software migration as described in [2].

In practice, there are on-going projects for all three Approaches. A more detailed examination shows, however, that reengineering methods and tools are used mainly in Approach A – and there they are often restricted to source code analysis.

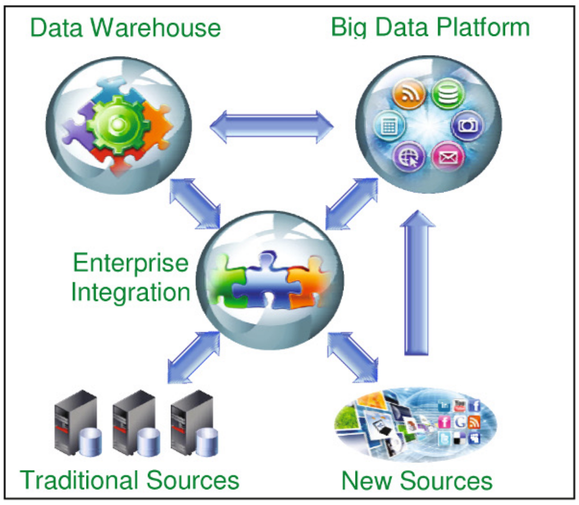
The reasons for this sparse use of reengineering techniques in such major transformation projects are not obvious at the beginning. The need to be discussed and potentially changed, because the neglected use of suitable technology can be the cause of degraded quality, project delays and significant cost overruns.

2. Big Data Re-Architecting

Big Data is becoming part of the Enterprise (Information) Architecture of many companies [3]. It provides additional functionality to generate business value from

- significantly more ("big") data or
- different (e.g. unstructured) data or
- fast-flowing (e.g. streaming) data.

Big Data solutions usually interface with existing Enterprise Architecture components and trigger architecture transformation:



In most cases today, Big Data components (e.g. a streaming engine, a Hadoop [4] based analytics engine, a text analyzer, an entity resolution engine, etc.) are added to the architecture. They usually have multiple interfaces among each other, and they require interfaces with the existing components, e.g.

- Data ingestion may need to include loads of existing data into a Big Data platform (not only into a data warehouse or staging area), in addition to the load of external/Internet-scale data.
- Big Data results are often structured and may need to be loaded into the data warehouse.
- A graphical user interface may need to be available in the Big Data exploration component, to provide a consistent look & feel to business users.

In order to provide these new interfaces, analysis of the legacy components regarding their functionality and interoperability will be useful. However, reengineering tools are rarely used for such analyses, let alone for the architecture and code transformations required.

One reason for this may be that analyses and changes address more "system components" (e.g. data warehouse, ETL components, data provisioning) than application components. The changes will often be different settings in an ingestion component or configuring a widget toward an existing GUI.
Yet the existing architecture is generally not completely documented, so the localization of/within components to implement the changes may be time-consuming.
In practice, one approach is to use The Open Group Architecture Framework (TOGAF™ [5]) to describe the re-architecting. As re-architecting is considered a major change, the TOGAF cycle must be performed starting from a new architecture vision, along with business architecture, application and data architecture, technology architecture, etc. The **migration planning** phase in TOGAF provides useful hints for the enterprise architecture evolution, including the use of reference architectures.
While the method basis is rather mature and tailorable, the tool support for real-life architecture transformation is still in its early stages.

3. Summary and Discussion

We have examined two major transformation initiatives (Core System Transformation and Big Data Re-Architecting), along with a number of real-life project examples that show the relevance of these initiatives.
We have also analyzed the use of reengineering methods and technology in the project examples. In addition to the positive reengineering experience, we address the areas of improvement in the reengineering portions of the projects:

- Communication: between business and IT; IT architect and project manager; project manager and IT specialist/developer; …
- Availability: of experts, methods, technologies, …
- Education: in reengineering, reverse engineering, re-architecting and the business topics of the transformation
- Interest: in working cross-discipline

These points are meant to initiate an open discussion on the evolution software reengineering.

References

[1] IBM: Core Banking Modernization. White Paper, October 2011.
[2] R. Gimnich, A. Winter: Workflows of Software Migration. WSR 2005. Softwaretechnik-Trends, vol. 25, no. 2, 2005.
[3] Paul Zikopoulos et al.: Harnessing the Power of Big Data. McGraw-Hill, 2013.
[4] Apache: Hadoop – an open source framework. http://hadoop.apache.org
[5] The Open Group: TOGAF™ Version 9. http://www.opengroup.org/togaf/