Bit-to-board analysis for IT decision making

Kwiatkowski, L.M.

2012

document version
Publisher's PDF, also known as Version of record

1. Thesis summary

1.1 Research topic

The research presented in this thesis has its roots in the EQUITY project. The objective of the project is to study relationships between yields and information technology. The underlying assumption is that decision making in regard to IT must be done in a calculated manner on the basis of information derived from facts. In this thesis we investigated the application of source code analysis techniques in the context of IT management. The undertaken studies revolved mainly around the following two managerial concepts: risk and cost. Source code implementing the information systems comprising organizations' IT-portfolios was used as the primary data provider. The exploration areas were driven by the real-life IT management quandaries that are faced by IT dependent organizations. All investigations were based on the real-world case studies. In the presented analyzes we relied on data extracted from source code, high level project reports, systems documentation, and experts feedback.

Information retrieval from source code sets the underlying theme of this thesis. Particularly, we focused on the kind of information which is useful in supporting the IT decision making process at all three levels: strategic, tactical, and operational. Although for the major part we concentrated on addressing issues encountered at the strategic and tactical levels we also discussed the process of information recovery. We elaborately explained all the low-level steps required in the process. By doing so we made it explicit what operational decisions are involved.

Without a doubt, for large organizations Cobol applications constitute the common component of the IT-portfolios. The investigations described in this thesis concentrate on the Cobol applications which run in the mainframe environments. Source code of the information systems was the primary object of analysis. Particularly, the analyzes carried out encompassed the following parts of source code: Cobol compiler listings, Cobol source statements, embedded SQL statements, and also source code comments.

1.2 Summary

The thesis presents two extensive investigations involving the employment of source code analysis in the IT management context. Each investigation was based on a real-world case study. We now summarize the two case studies.

Recovering management information from source code It is a fact that IT management relies profoundly on relevant information which enables risk mitigation or cost control. However, as it turns out the needed information is frequently either missing or its gathering boils down to daunting tasks which do not always deliver the required outcome. To address this problem we proposed an approach to recovery of management information from the essence of IT; the software's source code. We showed how to employ source code analysis techniques so that the indispensable management information is recovered. In the presented approach we exploited the potential of data concealed in the source code implementing information systems. In particular, we focused on the source code statements, source comments, and also compiler listings. We demonstrated how to depart from the raw sources, process them, organize, and eventually utilize so that the bit level data gets leveraged to the portfolio level and becomes useful for board-level executives.

Our approach was deployed in an industrial setting. The study was based on a real-life IT-portfolio which propels business functions of an organization operating in the financial sector. The IT-portfolio comprises Cobol applications run on a mainframe with the total number of lines of code amounting to over 18.2 million physical lines of code. We dealt with a mixture of Cobol code written manually and generated with CASE tools, such as TELON, COOL:GEN, CANAM, and others. The essential input for this study was formed by compiler listings of the recent production version of the portfolio systems.

We showed how to obtain various managerial insights into IT. Particularly, we restricted ourselves to obtaining those insights which address some more common managerial quandaries. All illustrations were made on the basis of the IT-portfolio used in the study. Apart from deriving the essential management information such as portfolio partitioning into systems, systems' size approximation, size distribution over the entire portfolio and various business sub-portfolios, we also obtained an indication of the rate at which the portfolio expands. For instance, as it turned out the amount of source code in the studied portfolio expands annually by as much as 8.7%. We showed how we estimated the portfolio market value, assessed the cost of keeping the portfolio up-and-running, or the staff assignment scope. Using the recovered information we analyzed a number of what-if scenarios for the portfolio to assess future managerial indicators. We exposed various technology related challenges for the management. For instance, as it turned out maintenance of almost 60% of the modules is dependent on expensive CASE tools. Almost 40% of the modules rely on the no longer supported IBM OS/VS COBOL compiler. Approximately 15% of the modules suffer from excessive code complexity. We also analyzed various migration scenarios for the code generators and compilers. We found that such migrations will have a relatively high impact on the top critical business applications. Furthermore, the complexity of the migrations turned out to be non-trivial. For instance, we found that the Cobol implementation of the top critical systems involves as many as 4 different code generators. All the presented insights were discussed in the context of the organization which operates on the studied IT-portfolio.

Reducing operational costs through MIPS management MIPS incurred costs are substantial and the need for controlling them is indispensable given the high percentage they involve in the overall IT operational costs. Despite this fact the majority of IT executives do not manage MIPS usage. We departed from the fact that MIPS consumption

is directly dependent on the applications' code and leveraged the code level aspects up to the level of operational costs. In this chapter we focused on an approach to reducing costs of running applications. The major principle behind the approach is to achieve cost reduction through improvements to the SQL code embedded in the applications source code. We showed how to employ, in an industrial setting, analysis of a mainframe environment to locate the most promising source code for MIPS optimization. Our approach relies on the mainframe usage data, facts extracted from source code, and is supported by a real MIPS-reduction project which involved tuning of the SQL code.

In this study we investigated a production environment running Cobol applications totalling to 19.7 million of physical lines of code. Particularly, we analyzed the IMS-DB2 portion of the applications. According to our inspection approximately 25% of all the Cobol programs interacts with DB2. In the study we supported ourselves with the MSU figures relating to the top 100 most executed IMS transactions. Our analyzes showed that on a weekly basis between 72%–80% of all the major IMS transaction invocations involved database utilization. As it turned out the MSU consumption for those transactions was on average higher by more than a half compared to the consumption reported for the non-database related invocations.

An earlier small scale MIPS-reduction project triggered our full-scale portfolio analysis, since it saved approximately 3.8% of the monthly MIPS bills. Our approach enabled us to effectively constrain the search space for inefficient SQL code in a large set of source files. To achieve this we bridged the source code dimension with the financial dimension. We related the mainframe usage data to the source code implementing the IMS transactions. We proposed two code improvement scenarios and calculated the potential savings. We showed that by selecting as little as 14 Cobol-DB2 modules the average expected monthly MIPS related savings from code improvements are approximately 6.1%. By carrying out a more extensive code improvement project involving a potential 180 modules the savings can reach as much as 16.8% of the monthly bills.

The combination of SQL programing knowledge, findings from the MIPS-reduction project, and input from the experts gave us a set of syntactic rules which enable filtering out the potentially inefficient SQL constructs from the portfolio sources. We showed how to use those rules along with the code interdependencies information to narrow down the number of source files potentially worth optimization. With our approach we could identify as little as 0.78% of all modules as candidates for actual improvements. We presented our tooling in detail so that others can use our approach in their own context. As we showed the tooling is simple to implement. The approach we presented is suited for facilitation within a mainframe environment of a large organization. Furthermore, the two key characteristics of the approach are low-risk and low-cost.

1.3 Concluding remarks

In the face of the above results the employment of source code analysis in the context of IT-management presents itself as a vital tool. Such picture is drawn from the discussed case studies which were restricted to tackling the more common managerial problems. Nevertheless, the information which source code analysis yielded turned out to provide

input suitable to fueling the decision making processes. To conclude this thesis we now make a few remarks.

First, in the carried out investigations pragmatism was considered as one of the requirements behind the design of approaches to extraction of data from source code. To address the daunting task of analyzing sources we relied on the lexical approach. We showed that to be sufficient to effectively analyze code. Not only did the approach prove to be effective but also feasible for use in the industrial setting. This being mainly due to the simplicity concerning deployment and high scalability potential.

Next, the use of source code analysis showed to be a valid means to easily obtain information at the portfolio level. In both case studies we used the source code extracted data to recover information relevant for analysis of the high level IT management concepts. Furthermore, mapping the source code obtained information with the management data turned out not to be difficult. Take as an example the case study concerning cost reduction in which next to source code derived data we also employed mainframe usage data originating from external management reports. Reconciliation between the two data sources was not burdened with much effort.

Moreover, by reaching for source code we showed that it becomes possible to obtain information not available to IT-executives otherwise. So, aligning code analysis with IT-management delivers new dimensions. The dimensions provide additional input for planning, risks identification, costs assessment, etc.

Finally, the work presented in this thesis provides organizations with ready receipts. For the discussed studies we provided the approaches, tooling and examples. All assumptions were made explicit, so that everybody can adjust the assumptions to their own specific situation. For example, instead of public benchmarks organizations can use their own custom figures, provided they are available. The quintessence of using source code to fuel decision making remains unaltered by those adjustments.