Diss. ETH No. 22335

Agile Software Development with Object Databases

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

Tilmann Zäschke

Dipl.-Ing., Technical University of Darmstadt, Germany

born March 13, 1975

citizen of Germany

accepted on the recommendation of

Prof. Dr. Moira C. Norrie, examiner
Prof. Dr. Gustavo Alonso, co-examiner
O.Univ.Prof. Dipl.-Ing. Mag. Dr.techn. Gerti Kappel, co-examiner

2014

Abstract

Agile development practices have become widely adopted in the past decade. They are increasingly used in industry and have been subject to a wide range of research. The promises of agile development practices range from customer satisfaction and adaptivity to changing requirements to developer motivation and reliable software through iterative development and customer involvement.

However, what has received little attention so far is the effect of agile development practices on the development of information systems where the static nature of databases conflicts with the agility of the development approach. One feature of agile projects is the iterative development approach by which users may start using the software as soon as minimal functionality is available. In the case of information systems, this results in the creation of databases with possibly valuable data. Together with the agile principle of frequent release of software updates, this leads to a need for frequent database evolution of customer databases which results in increased development effort. At the same time, the tight coupling of the incremental application development with application usage gives rise to new opportunities for exploiting the agile principles in favour of an improved development process. In this thesis, we explore various effects of agile development practices on the development of information systems and investigate new ways in which information system development can benefit from agile practices.

We start with a *case study of a large software development project* of the European Space Agency (ESA) in which the author was involved. The project developed an information system for operating the ESA Herschel Space Observatory and processing observation data. The information system was developed using an agile approach involving development teams located in several different countries and research institutions. The development approach of the project was such that users used the system from early on in the development period while creating significant amounts of data and databases that needed to be continuously evolved in order to be compatible with the evolving data model and to be available later when the software became operational. This led to roughly bi-monthly database evolution being necessary for hundreds of databases throughout six years of development. Based on this case study, we highlight several issues and concepts related to the agile development of information systems and explore solutions and ways to benefit from agile development practices.

For example, we discuss the suitability of object databases for agile development due to their approach of avoiding separate logical and conceptual data models with their associated mappings, thereby reducing the effort of evolving the data model. This tight coupling of application data model and database also has the advantage that it allows the tracking of navigation between objects and classes. This adds a dimension to profiling data that permits *profiling on the level of the conceptual model* and which is more difficult to obtain from relational databases. We visualise this profiling data

on the level of the conceptual model and use profiling analysis to recommend and, if desired, execute model refactorings to improve the model. This profiling analysis and refactoring is mainly aimed at improving database performance, but it can also be used to identify access patterns and thus yield semantic information about database usage. Database profiling on the conceptual level can therefore indicate inconsistencies in the information system if they are caused by semantic discrepancies between data model, queries and the user's idea of how the application should be used.

Another concept that is suitable for the agile development of information systems is that of model-driven development. It has been argued that the visual approach to model editing is appreciated by model designers and that the concept of code generation from a data model facilitates application evolution by reducing the amount of manual code refactoring required. However, next to the aspect of application code generation, the concept of *generating database evolution code* from data models has received far less attention. In this thesis, we present an approach to generating and optimising database evolution code with respect to performance, maintainability and ordering for the insertion of data evolution functions.

Building on model-driven development, we also present the concept of *adaptive code generation*. This concept discusses the idea that the modelling language used by the model designer may contain semantically rich constructs, such as multiple inheritance or multiple instantiation which are not available in the object model of the targeted programming language. While such constructs are very useful, the mapping layers required to implement such constructs in a programming language tend to affect application performance and database space requirements. Moreover the need for such rich constructs may increase during the development phase as the data model grows from a simple model to a complex model towards the end of the project. To support growing models and handle the trade-off between performance and features, we propose a solution that provides different mapping layers which support the different constructs in a modular way. This choice gives the model designer fine-grained control and is complemented by a model analyser that recommends a mapping layer depending on the characteristics of the data model.

To evaluate our ideas, we developed the AgileIS framework, an integrated development environment for the agile model-driven development of information systems. The framework aims to resolving some of the issues observed during the case study and at finding novel ways of exploiting agile practices to further information system development. The features of the framework include model editors for multiple modelling languages, a model versioning system, application code generators with modular mapping layers for different DBMS, database evolution code generators for different DBMS and a tool for visualising profiling data including recommendations and execution of performance related refactorings. To demonstrate the validity of the approach, we integrated support for profiling features into the open source object database ZooDB. These features go well beyond the profiling support available in current object databases.

Zusammenfassung

Agile Software-Entwicklung hat sich im vergangenen Jahrzehnt zunehmend durchgesetzt, sowohl in der Industrie als auch als Gegenstand der Forschung. Dabei versprechen agile Entwicklungs-Praktiken zahlreiche Vorteile, einschliesslich grösserer Kundenzufriedenheit, Flexibilität gegenber sich änderndern Funktionsanforderungen, Motivation der Softwareentwickler, Software-Qualität durch iterative Entwicklung und laufende Einbeziehung des Auftraggebers.

Allerdings wurde bisher wenig untersucht wie sich agile Techniken auf die Entwicklung von Informationssystemen auswirkt, insbesondere wie sich die statische Natur von Datenbank-Schemata zu der Agilität der Software-Entwicklung passt. Eine besondere Eigenschaft von agilen Projekten ist, dass die Endnutzer, je nach Projekt, die Software benutzen können sobald minimale Funktionalität implementiert ist. Bei Informationssystem bedeutet das, dass die Nutzer schon frühzeitig potentiell wertvolle Daten in Datenbanken ablegen. In Verbindung mit den für agile Projekte kurzen Release-Zyklen resultiert dies in erhöhtem Aufwand für die Evolution der Datenbank. Gleichzeitig birgt die enge Kopplung von Entwicklung und Nutzung eines System neue Möglichkeiten für einen verbesserten Entwicklungprozess. In der vorliegenden Arbeit untersuchen wir wie die unterschiedlichen Effekte und neuen Möglichkeiten von agilen Praktiken im Zusammenhang mit der Entwicklung von Informationssystemen.

Als Ausgangspunkt diskutieren wir eine Fallstudie eines grossen Softwareprojektes der Europäischen Raumfahrt Agentur (ESA) in die der Autor involviert war. Ziel des Projektes war es, ein System zur technischen und wissenschaftlichen Betreibung des Herschel Space Observatory zu entwickeln. Das System wurde mit einem agilen Entwicklungsprozess und einem verteilten Team in verschiedenen Ländern und Forschungseinrichtungen entwickelt. In dem Projekt konnten die Endnutzer das System schon frühzeitig nutzen um parallel zur Softwareentwicklung die Hardware des Satelliten zu testen. Die anfallenden Daten mussten dabei umfnglich in späteren Projekt-Phasen verfügbar sein. Die kurzen Entwicklungszyklen machten es dabei notwendig, im Laufe von circa 6 Jahren rund alle zwei Monate hunderte Datenbanken zu evolvieren. Basierend auf dieser Fallstudie zeigen wir einige Probleme und Konzepte die bei agiler Entwicklung von Informations-Systemen auftreten können und diskutieren mgliche Lösungen und zusätzliche Mechanismen wie ein solches Projekt von agiler Entwicklung profitieren kann.

Unter anderem diskutieren wir zum Beispiel den Nutzen von Object-Datenbanken in agilen Projekten, besonders bezüglich deren Vermeidung von getrenntem logischen und konzeptionellen Modell, was die Evolution der Datenbanken deutlich vereinfacht. Die enge Kopplung von Datenbankmodell und Applikationsmodell vereinfacht es auch die Navigation in der Datenstruktur zu erfassen und auszuwerten. Wir visualisieren derartige Profiling-Daten, generieren Empfehlungen fr die Optimierungen des Modells

und führen diese gegefenenfalls auch aus. Das Profiling und die Optimierungen zielen hauptsächlich auf Performanz, aber sie enthalten durch die Nutzungsanalyse auch eine starke semantische Komponente. Dadurch können die Ergenbisse auch auf semantische Probleme des Systems hinweisen, etwa bezglich semantischer Diskrepanzen des Datenmodells, der Queries und der Art wie der Endnutzer das System nutzt um einen realen Anwendungsfall zu lösen.

In dieser Arbeit verfolgen wir auch die Idee der Modellgetriebenen Entwicklung. Es wird oft argumentiert, dass der visuelle Aspekt bei der Modellierung von Modelldesignern begrüsst wird und dass automatiserte Generierung von Code den Aufwand der manuellen Programmierung reduziert, insbesondere im Bezug auf Refaktorings des Modells. Dabei wird seltener diskutiert, dass Modell-Getriebene Entwicklung auch fr die Generierung von Code für Datanbankevolution nützlich sein kann. In dieser Arbeit präsentieren wir ein Konzept zur Generierung von solchem Code, insbesondere mit dessen Optimierung bezüglich Performanz, Maintainability und der Erweiterbarkeit hinsichtlich Funktionen zur Evolution der eigentlichen Daten.

Aufbauend auf dem Konzept der Modellgetriebenen Entwicklung diskutieren wir die Idee der adaptiven Codegenerierung. Die Idee dahinter ist, dass whrend der Modellierung einer Applikation eine 'reiche' Modellierungssprache mit Konstrukten wie multipler Vererbung oder multipler Instanzierung verwendet werden kann, wobei diese Konstrukte nicht notwendigerweise in der anvisierten Programmiersprache nativ verfügbar sind. Solche Konstrukte können zwar in allen üblichen Programmiersprachen abgebildet werden, aber der notwendige Abbildungscode hat oft einen negativen Einfluss auf Performanz und bentigten Speicherplatz. Unabhängig davon ist es oft zu Beginn eines Projektes nicht klar, ob entsprechende Konzepte nützlich sind, gleichzeitig kann die Wahrscheinlichkeit, dass solche Konzepte von Nutzen sind, mit zunehmender Komplexität während der Entwicklungszeit zunehmen. Um solche wachsenden Modell und steigenden Anforderungen zu unterstützen, und den negativen Einfluss des Abbildungscodes zu kontrollieren, schlagen wir ein Konzept vor welches solchen Abbildungscode modularisiert. Das Konzept gibt dem Modell-Designer detaillierte Kontrolle über die generierte Abbildung, wober er/sie von einem Analyse-Werkzeug unterstützt wird, welches basierend auf einer vorheringen Analyse des Modells ein Abbildungskonzept vorschlägt.

Um unsere Ideen zu evaluieren haben wir das AgileIS Framework entwickelt. AgileIS ist eine integrierte Entwicklungsumgebung für agile modellgetriebene Entwicklung von Informationssystemen. Diese Entwicklungsumgebung zielt darauf ab, einige der besprochenen Probleme zu lösen und ausserdem neue Wege zu finden wie agile Entwicklung für Information-Systeme nützlich sein kann. AgileIS enthält Modell-Editoren für verschiedene Modellierungssprachen, ein System zur Modell-Versionierung, Generatoren für Applikationscode fr verschiedene Datenbanksysteme, Codegeneratoren für Evolutions-Code fr Datenbanken und ein Werkzeug zur visualierung von Profilingdaten inklusive halbautomatischen Refaktorings basierend auf generierten Empfehlungen fr Modelloptimierungen. Ausserdem haben wir die quelloffene Objektdatenbank ZooDB angepasst und mit profiling Komponenten erweitert die in anderen Systemen nicht verfügbar sind.