Chapter 2.10
Test-Driven Development of Data Warehouses

Sam Schutte
Unstoppable Software, Inc., USA

Thilini Ariyachandra
Xavier University, USA

Mark Frolick
Xavier University, USA

ABSTRACT

Test-driven development is a software development methodology that has recently gained a great deal of traction in the software development community. It focuses on creating software-based test cases that define the business requirements of an application before beginning the coding of the application itself. This paper proposes that test-driven development could be a useful methodology for data warehouse projects, in that it could help team members avoid some of the major pitfalls of data warehousing, and result in a higher-quality end product.

INTRODUCTION

Over the course of the last decade, the business of software development has gone through rapid changes due to the introduction of new light-weight methodologies. These methodologies - such as Extreme Programming, Agile Development, and SCRUM - emphasize a focus on "frequent inspection and adaptation" (*Agile Software Development,* 2009) of business requirements and technical architectural structure, and introduce new programming methods such as peer programming and stand-up meetings that help reduce re-work and improve quality.

One of the newer methods used by Agile development teams is test-driven development, which is "a software development technique that uses short development iterations based on pre-written test cases that define desired improvements or new functions" (*Test-Driven Development*, 2009). The overall result of the introduction of these new methodologies has been a measurable improvement in the quality, time-to-market, and

DOI: 10.4018/978-1-61350-456-7.ch2.10

productivity of software development teams (Desmond, 2009).

At present, as organizations compete in a dynamic, hyper competitive business environment characterized by a massive influx of data, business intelligence (BI), is seen as the ultimate solution that will help organizations leverage information to make informed, intelligent business decisions. According to Gartner, the worldwide BI platform revenue is forecast to grow at a compound annual growth rate of 8.1 percent through 2012, reaching $7.7 billion in 2012 (Knight, 2008).

While BI and data warehousing is now a mature market, historically, failure rates have been high (Kelly, 1997). A more recent assessment of data warehouse failure in 2007 suggests data warehouse failure rates can be as high as 50 percent (Embarcadero, 2008). In addition, ensuring high data quality within the data warehousing environment still continues to be a major issue. Industry trends in data quality indicate that companies are looking for "pragmatic approaches" to managing data quality (English, 2007). Organizations are searching for practical ways to handle errors and inconsistencies within source systems to create effective data staging practices. Test-driven development offers a unique opportunity to enhance current data warehouse implementation practices; especially in the area of improving data quality.

The purpose of this paper is to discuss the background and benefits of test-driven development and explore the application of this effective software development methodology to data warehouse implementation projects.

STATUS OF TDD IN THE BI AND DATA WAREHOUSING SPACE

While the data warehouse and business intelligence industry has adopted a few of the methods from agile development, such as "bite size analysis" (Arnett, 2002) and improved coding practices, the methods of test-driven development are only starting to gain use. For instance, test-driven development has been proposed as a way to verify the validity of business intelligence reports such as Crystal Reports (Landes, 2005). In the specific area of data warehousing however, test-driven development does not appear to have made an impact.

If the principles of test-driven development were applied to a data warehousing project, the resulting data warehouse would likely be of high quality and its functionality would not exceed the scope of the original request. Additionally, it would be scientifically provable through the use of the software-based test cases that the data within a data warehouse was correct – even in the face of disbelieving executives who may question the accuracy of reports generated from the data warehouse.

To succeed in a test-driven environment, a data warehouse team would have to follow several guidelines. First, all functionality and data in the system must be specified by end-users, and then test cases must be created which specifically address each piece of data or data relationship. Second, as the system is developed, these test cases must be run against the data warehouse. When a test passes, that particular feature (e.g., database table or field, or ETL process) within the data warehousing environment is considered complete. In this way, the pending tasks in the project implantation plan simply become any tests that are currently failing.

BACKGROUND: WHAT IS TEST-DRIVEN DEVELOPMENT?

Test-driven development (TDD) was first introduced as a part of Extreme Programming (XP), which is a software development methodology created by Kent Beck in the early 1990s. Extreme programming focuses on four core values: "Communication, Simplicity, Feedback, and Courage" (Wells, 1999). These core values lead to a series of