Agile Development in Data Warehousing

Nayem Rahman, Intel Corporation, USA
Dale Rutz, Intel Corporation, USA
Shameem Akhter, Western Oregon University, USA

ABSTRACT

Traditional data warehouse projects follow a waterfall development model in which the project goes through distinct phases such as requirements gathering, design, development, testing, deployment, and stabilization. However, both business requirements and technology are complex in nature and the waterfall model can take six to nine months to fully implement a solution; by then business as well as technology has often changed considerably. The result is disappointed stakeholders and frustrated development teams. Agile development implements projects in an iterative fashion. Also known as the sixty percent solution, the agile approach seeks to deliver more than half of the user requirements in the initial release, with refinements coming in a series of subsequent releases which are scheduled at regular intervals. An agile data warehousing approach greatly increases the likelihood of successful implementation on time and within budget. This article discusses agile development methodologies in data warehousing and business intelligence, implications of the agile methodology, managing changes in data warehouses given frequent change in business intelligence (BI) requirements, and demonstrates the impact of agility on the business.

Keywords: Agile Development, Agile Methodology, Business Intelligence, Data Warehousing, Waterfall Model

1. INTRODUCTION

Traditional data warehouse projects often follow a typical waterfall development model in which rigorous serial efforts are made to collect complete requirements, design comprehensive architectures and data models, develop and populate repositories, thoroughly test all functionality under a variety of scenarios, and, ultimately, deliver the analytical reports and artifacts that users require to run the business.

In real world enterprise data warehouses (EDW) we have seen that the duration of these projects are often nine months to more than a year, with few lasting less than six months. These complex affairs involve various stakeholders often with conflicting priorities. The water-fall model often disappoints the business intelligence community. We attribute many of these unsuccessful outcomes to frequently changing business requirements coupled with the blazing speed of technology innovation. Business users are often unaware of advances in technology, and technologists are frequently out of touch with...
the rapid changes occurring in the business environment. As a result, it is often not until the testing or even the deployment phase that significant learnings are achieved resulting in extensive redesign, redevelopment, and retesting. Thus slippage can be anticipated with the waterfall model. A solution designed 9 months or a year ago is almost guaranteed to be obsolete on delivery.

Software projects are labor intensive, time consuming and expensive, hence, there is significant pressure to deliver on-time. Issues with resource assignment are encountered throughout the traditional development life cycle, as not all team members are equally engaged during different periods. While the analysts are busy describing business processes and requirements in natural language, the developers play a consultative role only. Later, the developers will be working night and day to meet the deadline while the analysts play a consultative role. Usually, at some point in project’s life cycle, it falls behind schedule raising concerns of stakeholders. To resource manpower in such lengthy projects under waterfall approach managers have to make extraordinary efforts to deal with schedule pressures and late software projects (Williams et al., 2004). Often they respond by adding more man power to stay on schedule. This method of addressing schedule slippage is often counterproductive. Through his law, “adding manpower to a late software project makes it later,” Brooks (1995) asserts that the assimilation, training, and intercommunication costs of adding new team members outweigh the associated team productivity gain in the short term.

Agile development methodologies can help to overcome the manpower issues, schedule pressures, and late delivery of projects. These methodologies help the development team to work together, being more equally engaged throughout the process. Waterfall development life cycles consume large numbers of resources in data warehousing projects and often take longer than expected time to complete. Organizations have been working to remove redundancy and minimize project duration by eliminating idle time of project members during the project life cycle. It is critical to find ways to successfully implement software development projects on time, as nearly all of these initiatives contribute directly to an organization’s bottom line.

An agile development approach is proving essential for effective project implementation for data warehousing (Brobst et al., 2008). Life cycle analysis (LCA) is useful to identify redundancy among systems analysts, data modelers, developers and testers’ idle time as well as time used by each stage of a project. Agile development methodologies can help to identify inefficiency in each part of a data warehousing task or each phase of a data warehousing project. Identifying various ways to reduce inefficiency, redundancy, wasted time is best serviced with the development of a system of metrics. These metrics can help to identify which resources are best used for what tasks and at what point in the projects life cycle. Additionally they can highlight idle periods and waste.

The major driving force for shifting to an agile development approach is the need to achieve competitiveness from data warehousing applications which is put at risk by late project implementation and cost overruns. A mix of measures including iterative releases, a switch from waterfall approach to agile development principles and frequent production releases (Khramov, 2006) are needed. Agile development is the study and practice of implementing software projects efficiently and within a short cycle time via iterative releases. It usually involves using more efficient and experienced developers, using automated tools, such as the one proposed by Rahman et al. (2010), that consume fewer resources and use less project time. Another key component is that the same resources remain engaged with the project for successive iterations. This is critical as relationships as well as partnerships grow stronger with each release. Business people become savvier about the technology at the same time the technologists are becoming more aware of the business processes they are trying to enhance. In data warehousing there is growing pressure...