INFORMATICA®
The Data Integration Company™

Agile Methodology for Data Warehouse and Data Integration Projects

Karthik Kannan, Informatica Professional Services

WHITE PAPER

This edition published August 2011

Table of Contents

ABSTRACT

This case study explores the applicability of agile software development methods in the context of data warehouse and data Integration projects. The conclusion is that agile methods are indeed suitable if several modifications are made. Data warehouse projects differ from other software development projects in that a data warehouse is never really a completed project. Every phase of a data warehouse project has a start and an end, but the data warehouse will never go to a stable end state and is therefore an ongoing process. Furthermore, the concept of software refactoring, which is common in agile software development, requires special treatment in a data warehouse because new iterations of the data model invalidate the historical data based on a prior data model. As a result, data conversion and data validation become necessary activities in data warehouse iterations.

KEYWORDS

Data warehouse project management, agile methodology, agile data integration, ETL project management, and data integration project management.

Introduction to data warehouse project management

There is a common belief among most project managers that data warehouse projects are difficult to manage. Data warehouse project management differs from most other software project management in that a data warehouse is never really a completed project. Every phase of a data warehouse project has a start date and an end date, but the data warehouse will never go to an end state. Roles and responsibilities assigned in a traditional way seem to result in too much rework, and the traditional waterfall methodology does not seem to work for controlling the project.

Data warehouse projects are ever changing and dynamic. These characteristics make project management for a data warehouse challenging and unique; they are also a key reason why agile methods are appropriate.

Purpose of this document

To provide a detailed description of the agile methodology and how it helps data warehouse and data integration projects. A prerequisite to reading this document is a basic understanding of project management and data warehousing.

Agile software development

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. The term was coined in 2001 when the Agile Manifesto was formulated. Different types of agile management methodologies can be employed such as Extreme Programming, Feature Driven Development, and Scrum.

Any software method is considered agile as long as it adheres to the four principles of the Agile Manifesto, which values:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

For this particular case study, we have used Scrum, an iterative incremental framework. We used 30- to 45-day iterations and emphasized close working relationships between the business and the project team.

Agile methods break an entire project into small increments that vary from a week to six weeks. These short timeframes are termed as iterations or sprints that have minimal planning and do not directly involve any long- term planning.

Each iteration is completed by a team through a full software development cycle, including planning, requirements analysis, design, coding, unit testing, and acceptance testing, when a working product is demonstrated to stakeholders. An iteration may not have enough functionality to make a full release of the product or project, but the goal is to have a release at the end of each iteration. Multiple iterations may be necessary to release a product, to add a feature, or to complete an entire project.

In Scrum, this concept is referred to as delivering something that is “potentially shippable,” meaning it is a finished product that if need be could be delivered to the customer as is even if all the features/functionalities are not built into the particular version. In concept, it is solid enough by itself to either be shipped or to have additional code built on top of it without breaking what has already been built.

Agile team structure

Team composition in an agile project is usually cross-functional and self-organizing without consideration for existing corporate hierarchy or the corporate roles of team members. Team members normally take responsibility for tasks that deliver the functionality that iteration requires.

The team is usually made of 5 to 9 members that generally work in a single open office because the project requires a lot of cross-functional discussion within the team

Principles of agile method

- Customer satisfaction by rapid, continuous delivery of useful software
- Working software is delivered frequently (in weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Close, daily cooperation between business people and developers
- Face-to-face conversation is the best form of communication (co-location)
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design
- Simplicity
- Self-organizing teams
- Regular adaptation to changing circumstances

Characters involved in the development

Scrum defines a number of roles. All roles fall into two distinct groups called pigs and chickens. The core group is referred as pigs and the ancillary group as chickens, based on the nature of their involvement in the development process. These groups get their names from an old joke about the role of a pig and a chicken in a bacon and egg breakfast; the chicken is involved but the pig is committed.

Pig roles

Pigs are committed to building software regularly and frequently, they are the ones producing the product.

Product owners

Product owners represent the customer. They review and prioritize the items for development. They are customer focused or customer-centric and are the key decision maker about whether the release is acceptable for production release.

Scrum master

Scrum is facilitated by a Scrum master, whose primary job is to remove impediments to the ability of the team to deliver the sprint goal. The Scrum Master is not the leader of the team (the team is self-organizing) but acts as a buffer between the team and any distracting influences. He or she acts as a facilitator.

Project team

The team has the responsibility to deliver the product. A team is typically made up of five to nine people with cross-functional skills who do the actual work (design, develop, test, etc.).

Chicken roles

Chicken roles are not part of the actual Scrum process and are not directly involved in construction of the product, but must be taken into account.

Stakeholders (customers, vendors)

Stakeholders are the people who enable the project and for whom the project will produce the agreed-upon benefits. They are directly involved in development only during the sprint reviews.

Managers

The managers set up the environment for the product development organizations and provide resources for the agile team members.

Meetings in an agile lifecycle

In an agile lifecycle, a project team can have different sets of meetings involving different participants; each meeting will have a different characteristic involved, as enumerated below.

Release planning meeting

A release planning meeting is used to create a release plan, which lays out the overall project goals, objectives and backlog of stories. The release plan is then used to create iteration plans for each sprint.

The purpose of the Release planning meeting is to have everyone in the team understand and commit to delivering the agreed release.

A release generally fixes only the target date or target scope, but not both since the time and effort to complete all the work is defined only at a high level. The project team and the scrum master are present at a release planning meeting as well as the product owner who determines the priority of items on the release backlog list.

Daily scrum

Each day during the sprint, a project status meeting occurs. This is called the “daily scrum” or the “daily standup”. This meeting has specific guidelines:

- The meeting starts exactly on time
- The meeting is limited to 15 minutes

During the meeting, each team member answers three questions

- What was done since yesterday?
- What will be done today?
- What obstacles are slowing down progress?

The Scrum Master is responsible for facilitating the resolution of the obstacles.

Daily scrums enable all the team members to speak daily, stay informed, and quickly help each other solve issues/problems rather than allowing them to linger.

Sprint planning meeting

At the beginning of the sprint cycle, a sprint planning meeting is held to:

- Select what work is to be done
- Prepare the sprint backlog, that details the time it will take to do that work with the entire team
- Identify and communicate how much of the work is likely to be done during the current sprint

Iteration plans created during release plan meeting are used in the sprint planning meeting. This meeting is held for a maximum of 8 hours.

Sprint review meeting

At the end of each sprint, a sprint review meeting is held. The sprint review meeting is time-boxed to 4 hours. The goals of this meeting are to:

- Review the work that was completed and not completed
- Present the completed work to the stakeholders (showing data validation/ data loads in test area)

Sprint retrospective meeting

This meeting is held after the sprint review meeting. In this meeting, all team members reflect on the past sprint. This meeting is time-boxed to 3 hours. To make continuous process improvement two main questions are asked in the sprint retrospective:

- What went well during the sprint?
- What could be improved in the next sprint?

Choosing suitable projects for agile

Todd Little developed an effective conceptual framework to select suitable projects for agile. It has two dimensions:

- **Uncertainty:** usually reflects customer churn or turmoil, venturing into a nascent market, fast technology change, or obsolescence of business designs
- **Complexity:** typically due to overly ambitious scale and scope targets for a project, insufficient understanding of the state of the art in a domain, numerous dependencies, or overly complex organizational structure

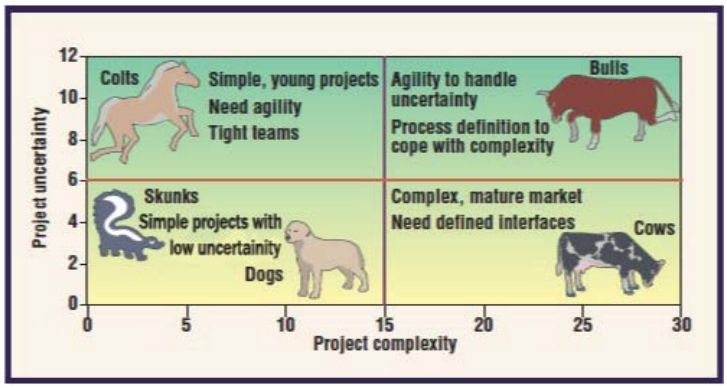

Figure 1.

As Figure 1 illustrates, these two dimensions define four quadrants as follows:

- Low on uncertainty; low on complexity (Dog and Skunk)
- Low on uncertainty; high on complexity (Cow)
- High on uncertainty; high on complexity (Bull)
- High on uncertainty; low on complexity (Colt)

Once a project has been characterized in one such quadrant, its suitability to be carried out using agile methods can be assessed. The key point is that agile methods are concerned with addressing the uncertainty of outcome. Agile does not address complexity per se. It might indirectly help with complexity if it leads you toward deeper thinking about adaptive software development.

A project of high uncertainty will be suitable for agile methods. The value adds of agile can be conclusively demonstrated in such a project through the quick feedback cycle and the ongoing adjustments to the product backlog. In contrast, a project of low uncertainty might only benefit marginally from the application of agile methods.

Why agile methods are better than traditional project management for a data warehouse project

As mentioned above, agile methods are well suited for projects that are uncertain. Here are several reasons why data warehouse projects are appropriate candidates for agile methods.

- As it relates to a data warehouse or data integration, the more the customer drills down into understanding the data in an enterprise context, the more new rules will be identified
- Results from analyzing data lead to new requirements that vary from minor changes (change in reports) to major changes (change in technology)
- Business users expect quick delivery when they are still refining their requirements.
- During the development of a data warehouse, any number of small independent projects can be executed to load the data warehouse from different sources handling multiple interdependent DW projects under one DW or business intelligence program. This is a big challenge.
- During the development cycle, there are dependencies among different systems at different levels. Data model design depends on accesses and business requirements. Development depends on the data model. Data load depends on development, and business users depend on the data load to refine their requirements.
- Development teams fight hard to meet expectations to deliver data to the business users, which leads to skipping data standardization and taking shortcuts for testing, plus incomplete documentation and compromised quality
- Process steps are often either skipped or expectations are not set properly
- The data model changes frequently due to changes in reporting and other business requirement changes
- Business users are not clear on the data until they see the data in the target system
- Changes are made in the source system by the business after starting the project
- Because most data warehouse projects deal with different databases and different source and target environments, access-related changes are a major concern when a DW project is started
- The business logic has frequent changes
- Most DW projects run for long periods and are subject to continual change
- Required resources can be added as the project grows

Agile data integration case study

This case study is based on the project that was executed at a leading power company (LPC) in the United States, and this project plan is a modified version of the actual plan for clear understanding.

LPC is a $22 billion energy corporation with operations in the United States and Brazil. It has energy production plants in six different states. LPC has a mature Integration Competency Center (ICC) that uses Informatica® technology throughout its different divisions. Frequent merger and acquisition activities presented an enterprise data integration (DI) challenge in absorbing new data assets and customer groups. Also senior management focused energy efficiency projects on the customer retail marketing sector to stay strong in the market, which drove additional challenges.

The following case study is based on a project that is related to the energy efficiency project done at LPC.

- Project manager is identified
- Project manager/Scrum Master decides to use agile methods; the product owner and the stakeholders support this approach
- Sprint size is decided as four to six weeks, but it can be increased or decreased based on the requirements and deliveries met
- Business identifies the different data that has to be pulled from the various sources
- Project's main requirement is to create various reports for higher management for their new products on energy efficiency. The reports are built using the following data:
  - Budget data is a financial document created by higher management in Excel spreadsheets for various products
  - Forecast data deals with what is likely to happen and is created by middle managers in Excel spreadsheets
  - Customer Relationship Management (CRM) data of this organization resides in a CRM system in the cloud
  - Survey data from various customers about products are stored in a cloud server
- Resources are identified; below are the people needed for the preceding requirements:
  - Stakeholders/users
  - Product owner—to represent sponsors and stakeholders
  - Scrum Master - project manager
  - Team that includes data architect, system architect, and an ETL architect/ETL resource
- Based on the requirements and the meetings with the stakeholders, the business decides to start the project with budget data
- Because this white paper deals more with the ETL part of the project requirements, reporting requirements are not discussed. Reporting requirements can be planned and plugged in by segregating according to the availability of data on the data warehouse
- In this case study, all the items given in the Tasks planned column are identified during the sprint planning meeting. Tasks completed are validated at the sprint end meeting by the project team and include detailed deliverables.

Project lifecycle of LPC reporting project

Sprint 1: 5 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Create a report for budget data	· DW data model for budget data · Design, coding, and testing of DW data model and all related documentations · Environment accesses required for the data loads of future sprints worked on Initial load – budget data	· Budget data load to the DW · Budget data verification on DW using sample reports
Sprint 2: 6 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Backlog work from previous sprint is identified as none · Create a report for CRM data	· Data model for CRM data · Design, coding, and testing of CRM ETLs and all related documentations · Budget data load in production for ongoing process scheduled · Initial data load – CRM data	· CRM data load to the DW · CRM DW data review using sample reports
Sprint 3: 6 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Create a report for survey data · Business identifies new requirements in budget data load after comparing it against the CRM data and after reviewing the results of budget data · New business requirements will be taken into a new sprint after the survey of data load as a backlog.	· Data model for survey data Design, coding, and testing of survey data load ETLs and all related documentations · CRM data load in production for ongoing process and budget data load ETLs scheduled · Initial data load – survey data	· Survey data load to DW · Survey data review using sample reports
Sprint 4: 4 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Backlog work from previous sprint – New source for budget data, new files for survey data · After reviewing previous results, business decides to include forecast data in the reports and to provide requirements during beginning of next sprint	· Changes to the budget data · New code change moved to production · Rerun of budget data	· Data verification on DW using sample reports
Sprint 5: 5 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Create a report for forecast data	· Data model for forecast data · Design, coding, and testing of DW data model and all related documentations · Initial data load – forecast data · All previous loads and forecast data load scheduled	· Data validation using sample reports
Sprint 6: 6 weeks		
Task(s) planned	**Task(s) completed**	**Demonstrable/Deployable product**
· Production warranty · Bug fixes	· Production warranty · Bug fixes	· Production warranty · Bug fixes

The project was successful because of the close working relationship between the development and the business teams. It helped enforce accountability.

If there was a task that a team member, either from the project team or from the business, was not keeping up with, the project team could quickly identify it and also identify the risks it posed to the team's ability to successfully complete the tasks in the sprint. The Project team could quickly raise a red flag with our sponsor to either have it taken care of so the sprint would be successful or have the accountable person explain in our stakeholder meeting why he or she failed to meet that commitment, in front of all the project stakeholders.

Thus, by attaining several near short-term goals, paved the way to reach the final long-term goal of delivering a fine product to the customers and stakeholders.

Bibliography

- Data Warehouse Project Management, Information Management Magazine, March 2006
- The Concise Executive guide to agile (IEEE CS Press ready notes) – Israel Gat
- Context adaptive Agility managing Complexity and Uncertainty, IEEE Software, vol 22 no. 2 (April 2005) – Todd Little
- Wiki links on Scrum developments and Extreme programming practices
- Advice on Conducting the Scrum of Scrums Meeting (May 2007) – Mike Cohn
- http://www.scrumalliance.org/articles/46-advice-on-conducting-the-scrum-of-scrums-meeting.
- http://www.sprintplanning.com/SprintPlanningRules.aspx
- http://www.extremeprogramming.org/rules/planninggame.html

About the author

Karthik Kannan is a senior consultant at Informatica Professional Services[SM]. He has extensive experience in the fields of data warehouse, data migration, and data quality. He has worked and led data warehouse project engagements using the Informatica Platform. He has successfully completed the Certified Software Quality Audit exam and earned a bachelor's degree in engineering in computer science at Bharathiyar University, Coimbatore, India.

kkannan@informatica.com

Learn More

Learn more about the Informatica Platform. Visit us at www.informatica.com or call +1 650-385-5000 (1-800-653-3871 in the U.S.).

About Informatica

Informatica Corporation (NASDAQ: INFA) is the world's number one independent provider of data integration software. Organizations around the world gain a competitive advantage in today's global information economy with timely, relevant and trustworthy data for their top business imperatives. More than 4,200 enterprises worldwide rely on Informatica to access, integrate and trust their information assets held in the traditional enterprise, off premise and in the Cloud.

INFORMATICA®
The Data Integration Company™

Worldwide Headquarters, 100 Cardinal Way, Redwood City, CA 94063, USA
phone: 650.385.5000 fax: 650.385.5500 toll-free in the US: 1.800.653.3871 www.informatica.com

1749 (08/12/2011)