Using Scrum in Master Data Management Development Projects

Goran Milanov, Angelina Njeguš

ABSTRACT

Master Data Management (MDM) development projects are specific both in their functionalities and organizational requirements. The existing literature about practical experience in managing MDM development projects using Scrum is limited. This paper aims to analyse problems and challenges as well as to present guidelines with defined rules when applying Scrum framework within MDM development project. In practice these projects are huge and costly since they involve a big part of enterprise resources, so implementing MDM using Scrum can speed up its realization with successful outcomes. In this paper, a literature review of Scrum framework, MDM concept and different MDM implementation techniques are carried out. The empirical guidelines with proposed rules for managing MDM projects using Scrum are represented.

Keywords: Master Data Management, Scrum, Data Governance

1. INTRODUCTION

The lack of a single consistent enterprise view of core business entities impacts on overall business therefore many organizations have decided to go for master data management development project. Master Data Management (MDM) represent a single view of master data that include the relationships and dependencies of all the enterprise data in order to support business processes and decision making [4]. The goal of MDM is to create and maintain consistent, complete, contextual and accurate business data for all stakeholders in a controlled and „single-view“ capable manner [18].

There are a number of ways to architect a successful MDM solution. To determine the best architecture, it is necessary to understand the benefits and drawbacks of each type of solution. In this paper, different types of MDM implementations will be considered. Managing MDM projects involve different stakeholders, processes, resources and policies (through data governances) what makes these projects time-consuming and expensive. The main question that enterprise IT managers consider is what software development methodology to use, traditional or agile.

Recent empirical research identify a number of advantages of agile software development methods by comparing them to traditional methodologies. However, from the spiral to current agile methodologies there are still some obstacles that makes MDM project challenging and heavy-duty work [11].

Common agile software development methodologies are Open Unified Process, Scrum, Extreme Programming (XP) and Microsoft Solution Framework (MSF) for Agile. Among them the most popular are Scrum and XP. Agile methodologies such as Scrum give best result for small stand-alone systems with self-motivated developers and a participating users [10][17]. However, today agile methodologies are also used for large, complex systems like MDM, which require an enterprise perspective for activities such as enterprise data modeling, common metadata, data integration, and so on. Even though there is limited research on agile MDM. Some review articles [1][12] are written on MDM which try to give global agile MDM approach but there are almost no practical research articles about using Scrum when managing MDM projects. Understanding the concept of master data management and integrating them into Scrum project management framework is not an easy task. The purpose of this paper is to clarify activities that should be undertoken when managing MDM projects using Scrum and to understand related challenges in practice.

2. METHODOLOGY

MDM concept and its general implementation techniques are first studied by using existing literature as a key source. MDM when implemented correctly can provide significant value to the organization. Understanding and implementing MDM systems may be a priority for many organizations, but successful MDM implementations involve more than just the right technology. A literature review was also conducted to gain an accurate understanding about Scrum roles, process and artifacts. Scrum as loose set of guidelines, govern the development process of a system, from its design stages to its completion. Some common failures of the typical development process, such as unrealistic estimates of time, cost, and system quality, chaos due to changing requirements within and across enterprise systems, bad communication, integration, coordination and control, are improved through Scrum framework [19]. Based on the literature review, and authors knowledge and practical experience in architecting and managing different software development projects that include MDM functionalities using traditional or agile methodologies, the empirical guidelines with suggested rules for managing MDM projects using Scrum, were created. At first, the analysis of common problems, importance and implementation of MDM in practice are described. Through Scrum phases the key activities in MDM implementation are explained and seven rules are defined. Furthermore, MDM critical success factors and challenging issues related to managing MDM projects using Scrum are given.
3. LITERATURE REVIEW

In general, an enterprise has three kinds of business data: Transactional (supports the applications), Analytical (supports decision-making), and Master (universal business objects) [13]. Master Data is the critical business information supporting the transactional and analytical operations of the enterprise. In many companies, master data is kept in many overlapping systems and is often of unknown quality [4]. Improving master data quality and managing it more efficiently to optimize business processes is known as Master Data Management (MDM) [6]. Master data management initiatives have several goals, such as to create a single version of the truth that will reduce risk and eliminate debates about “whose data is right”, to create a standard, shared vocabulary about the key “nouns” of the business, to build a comprehensive view across silos, to simplify data management and standardize interoperability and so on [9]. From technical point of view MDM is a combination of applications and technologies that consolidates, and cleans corporate master data, and synchronizes it with all applications, business processes, and analytical tools [18]. The core question is how to create unified views of data from multiple systems and diverse sources.

3.1 Master Data Management Implementation Techniques

In literature can be found different types of MDM implementation, and usually they refer to a particular vendor technology [3]. However, depending on available budget, time schedule and resources, enterprise IT manager can choose one of the four general types of MDM implementation [20]:

a. Master Data Registry

Each enterprise solutions (ERP, SCM, BI, etc.) remains in control of its own data and they are entry points (Systems of Entry, SOE) and systems of record (SOR). Master data of the same class (for example, Customer), created in the different enterprise systems, are mapped through surrogate keys in the new data model called Registry, which is specially modelled data structure. Mappings between the surrogate keys can be in relationships one to one or one to many. This type of implementation requires constant supervision by owner of MDM because he has to map every new master data record that appear in some of the existing systems.

b. Data Aggregation

The system which is critical for enterprise business processes (usually an ERP system) is chosen as System of Record (SOR) and System of Entry (SOE) of master data. The disadvantages of this type is the fact that the ERP system usually does not include all master data classes that exist in other software systems. Therefore, there is a need to develop MDM application which will entry new master data into the ERP system. Despite those limits, this technique is still good way for fast organization and implementation of MDM system.

c. System of record only (Hub)

MDM application has complete control of the master data. New records from other systems are transferred into the MDM application for integration, which is the system of record. The flow of data from System of Entry is bidirectional. Disadvantages of this type is that still require a degree of data integration and cleansing as elements may come from the source systems and MDM application [15].

d. Complete enterprise MDM

The entire life cycle of the master data are managed within MDM solution. Although MDM is both SOR and SOE it may still require mapping data to other applications. The control of master data entry enables the MDM system to manage the data quality proactively. This type of implementation is the ideal case which is very difficult to implement in practice because certain compromises are always required due to the specificity of both the software systems and the organizational systems of the specific business systems.

3.2 Scrum Framework

Scrum is an agile, lightweight framework for managing and controlling software development in rapidly changing and distributed environments [2]. Scrum is especially useful for MDM project management and teamwork because it provides effective communication in the form of boundary objects (artifacts) and spanners (roles), it provides effective social integration by building up project team, and it provides needed control and coordination mechanisms [14]. Scrum has three major components:

a. Scrum Roles

Scrum has a very simple team structure that involves three boundary spanning roles: Product Owner, Scrum Master, and Team Members. Product owner works closely with the team to identify and prioritize system functionality in form of a Product backlog. Scrum master is responsible for tracking project status and coordination, defines the sprint duration, runs the daily meetings, helps team productivity and other activities in Scrum process. The Scrum team is usually a cross-functional, self-organizing team that consists of five to ten people who work full time on the project. The team members are responsible for building and testing system quality.

b. Scrum Artifacts

The primary artifact of a Scrum project is the system itself, although the main artifacts are: User stories, Product Backlog, Sprint Backlog, and Burndown chart. User story is the basic unit of requirements specifications, and typically consists of a title and a brief, usually narrative, description of the desired functionality. Features and user stories express functional requirements. The product backlog consists of features, bug fixes, nonfunctional requirements, and other that needs to be done in order to successfully deliver a system. The sprint backlog is the set of stories planned for implementation in an iteration (called Sprint) that usually lasts two or four weeks. There is only one backlog for the team. At the start of the sprint, the team breaks down each story into a set of tasks.
and when a task is finished it is registered on a burndown chart.

c. Scrum Process

Major activities in the Scrum process are: the sprint planning meeting, the kickoff, the sprint, the daily Scrum, and the sprint review meeting. The sprint starts with a sprint planning meeting where the targeted functionality is broken down and estimated. In the first part of Scrum meeting, the team defines the product backlog, which is basically a list of the project requirements, and than determines the sprint goal, which is the formal outcome from particular sprint. In the second part of the meeting, the focus is on creating the sprint backlog. On the kickoff meeting the team define the high-level backlog and the major project goals. Every day the Scrum team meets in a daily short stand-up Scrum meeting where they share what they did, plan to do or are doing, and if there are some problems or have new idea to share with others. The sprint review meeting is more informal and is held at the end of each sprint where developed functionalities are presented to the product owner.

4. ANALYSIS AND RESULTS OF MANAGING MDM PROJECT USING SCRUM

Realization of MDM projects differs from the most development projects because there is no clear problem visibility with master data from the business side and therefore, there is no clearly defined need for an MDM system. In most cases, there is awareness of the problems from IT department but there is not enough knowledge and experience. On the other hand, even if the knowledge exists, the IT team does not have enough strength to persuade top managers, to invest in such project, and influence on human resources to participate in the development and implementation of the MDM system. Since the problems with master data will not disappear by themselves, but will deepen, at the start, the IT should engage itself in finding a sponsor of the MDM project. In general, applying Scrum practice within MDM project has three groups of phases: planning and MDM architecture design, development, and closure phase [13]. Each one has a broad set of activities that should be done, and after each we suggest appropriate rules to be followed in order to successful MDM implementation.

4.1 Scrum Driven MDM Development Process

Guidelines with specific rules for successful managing MDM projects using Scrum are:

a. MDM Project Sponsorship

As discussed in the problem analysis, it is very hard to convince the top management to approve changes in the existing enterprise systems in order to implement an MDM system. It is more difficult to obtain permissions to plan and execute organizational changes which the MDM solution implementation requires. MDM systems do not have some obvious, exceptional manifestation which can be stated as a reason and justification for MDM system implementation. It is very difficult to give real reasons based on which some manager will decide to be a sponsor of an MDM project and take over the responsibility to convince the other top managers. The first thing that a sponsor of an MDM project should do is to secure the support of the top management for development of the MDM project, therefore the first rule can be stated as below.

Rule 1: Find a MDM project sponsor among top management to ensure that all necessary changes are implemented and supported at any management level.

b. Scrum Team Formation

According with Scrum practice [8], IT manager should appoint a Product Owner and a Scrum Master. The Product Owner should be familiar with enterprise master data, understands MDM problems, know available MDM implementation techniques, technologies and tools. The role of the Product Owner is to define MDM requirements, to select solution, technology and tools, and together with project sponsor to define adequate MDM implementation technique. According to that, the Product Owner and Scrum Master select the Scrum team who should have previous experience, understand the business domain and know how to use chosen technologies [16].

c. The Factual Situation Of Master Data Document

The first document that should be created is about factual situation of enterprise master data. According to this document team suggest architectural solution, define functional requirement, choose implementation technique, assess costs and duration of the project etc. The document should contain the following information:

- The list of master data sources, i.e. all applications that produce master data and are SOE and SOR
- The list of databases, their locations and access parameters where master data are stored
- Data models (ER diagrams and data dictionaries) where master data are stored
- The list of master data attributes, data types, allowed values, constraints, default values, dependencies, hierarchies and who owns the definition and maintenance of the master data
- The list of application systems that only use master data (where are master data neither generated nor maintained)
- The list of master data transformations which are used in data warehouse or reporting systems
- The list of working positions where employees insert or update master data
- The list of the same master data which are entered and maintained in different application systems
- The estimated quality of master data etc.

Based on foregoing, the second rule could be stated as below.
Rule 2: The Product Owner and the Scrum team should create the document called „The Factual Situation of Master data“ upon which the type of implementation, architecture and functional requirements are defined as well as the scope, time schedule, price and other resources of the project.

d. Defining the MDM Architecture

MDM architecture corresponds to the chosen implementation technique. However, architecture can be changed during the project, between sprints, or if the functional requirements are changed. The answer to the question which MDM implementation technique to choose depends on the number and diversity of application systems and on the readiness of the master data owners to transfer their ownership. In practice, data aggregation implementation [5] is usually realized first, since the majority of business systems possess one central (usually ERP) system on which they base the greater part of their core business activities. Large organizations have more essential application systems that their core business activities are based on. In such cases, it would be ideal to transfer the control of all master data to the MDM system (Complete enterprise MDM implementation), although the hub implementation would be also a satisfactory solution.

There are also enterprise systems that use ERP, CRM or other solutions that are acquired on the market and not developed from its own resources. In that case, it will be troublesome to do any required changes since the customer does not possess the source code. In the majority of cases, the functionalities of such systems were designed to enable insert and maintenance of data, not to take over data from other systems. Import of master data from an MDM system by the logic of some ETL mechanism can also be problematic because in some ERP and CRM systems the business logic of the application system itself defines value of keys and sometimes it is impossible to embed that logic into an ETL system. Such cases can make the implementation of Complete Enterprise MDM system impossible and in that case a master data Registry implementation could be used.

e. MDM System Ownership

Master data ownership depends on which MDM implementation technique is used. If the Data aggregation implementation model is used, then the owner is the organizational department which is the owner of the ERP data. In case of the Hub implementation or the complete enterprise MDM model the best way is to form a new department which will have the ownership of the MDM system.

Rule 4: Defining the owner of the MDM system is an organizational necessity in order to enable implementation of the MDM system and also to know who the client is, to whom the Scrum team can present results at the end of every sprint.

f. Defining Functional And Nonfunctional Requirements

The product backlog is complete list of functional and nonfunctional requirements, defined in the form of user stories that have to be implemented during the development process. In most cases, the team and the product owner start by writing down everything they can remember easily. That is usually enough for the first sprint. The product backlog can be easily expanded and changed as the knowledge grows. The product owner is responsible for setting the priorities, i.e. for grouping the items according to the priorities. An example of the user stories in product backlog for an MDM project is given in the table 1.

Rule 5: The Product Owner and the team members define the functional and nonfunctional requirements and create the Product backlog.

<table>
<thead>
<tr>
<th>Category</th>
<th>User Story</th>
<th>Conditions</th>
<th>Story Pts</th>
<th>Priority</th>
<th>Sprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDS</td>
<td>Get all the necessary installations of Microsoft SQL Server 2008 R2 and Master Data Services (MDS)</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MDS</td>
<td>Install Microsoft Master Data Services (MDS) on Windows 2008 virtual machine</td>
<td>It is necessary that the System unit of IT install MS SQL Server 2008 R2 and Master Data Services on a virtual machine with standard configuration</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MDS</td>
<td>Create a HR Master data model, on MDS system, for HR</td>
<td></td>
<td>90</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>ETL</td>
<td>Create an ETL for transfer Master data from HR system into MDM database</td>
<td></td>
<td>60</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>MDS</td>
<td>Create a Customer Master data model on MDS system</td>
<td></td>
<td>120</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ETL</td>
<td>Create an ETL for transfer Customer Master data from the ERP system</td>
<td></td>
<td>80</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
The points that are discussed during the first phase are: delivery date, the number and functionality of release, the most important releases, the list of packets or objects for backlog items, the project team structure, necessary tools, risk control, release cost (including also the cost of training) etc. At the end there are designed review meetings during which the teams exchange information, present problems and progress of the project.

4.2 MDM Project Implementation

Implementation of MDM solution is an iterative cycle performed in sprints, which may last from 2 to 4 weeks. The results of those iterations (sprints) are new versions of the MDM system which include added functionalities from the product backlog. However these requirements are not strict and sometimes the time of each sprint is different. Unlike waterfall approach, where project manager quantify work in terms of time, in Scrum team member use effort and degree of difficulty to estimate their own work by giving Story points based on predefined scale.

In the sprint planning meeting, the team estimate its effort for the user stories in the product backlog, so that product owner can effectively prioritize items, and, as a result, forecast releases based on the team’s velocity. However, even when teams possess a shared understanding of their scale, they can still estimate differently, so usually numerous rounds of estimation are required. On daily short stand-up meeting team members inform each other, for example, if a team member says: “Today I am going to finish SSIS packages for transferring clients address data from the ERP system to the MDM database“, the others know that, on the next meeting, he will say whether the task has been finished.

During the sprint, the realization of specific tasks from the Sprint backlog reduces the number of points. That process is known as the burn down. At the end of every sprint the team actualize a burn-down chart, and take a part in sprint review meeting where realized tasks are presented to the Product Owner and the MDM system owners.

During every sprint, the Scrum team perform all types of testing, create user and maintenance manuals. Before deployment, team should train middle and lower management who will train end users. This hierarchical training gives best results at the lowest cost.

**Rule 6:** MDM functionalities deployed in production have to be in accordance with the requirements defined at the beginning of the sprint, tested and users trained.

4.3 MDM Project Closure and Further Challenges

During closure phase, the Scrum team ends the development process and the MDM system is prepared for a release. Activities that should be done are integration, testing, user training, define organizational procedures; prepare user and product documentation, etc.

### a. Organizational Procedures and Instructions

The success of any MDM implementation depend primary on the adoption of a governance model across the enterprise. Governance implies ownership and accountability within and across an enterprise created with clear roles and responsibilities, and set of policies and procedures that provide the foundation for improving master data quality. Establishing master data governance is a major initiative within an organization. In collaboration with organizational departments, the product owner should build or redefine existing components of governance model. Governance model components include governance organization structure (clear roles and responsibilities), governance policies (executive and governance directives), and governance procedures (workflow needed to make various decisions) [13]. For successful MDM implementation, the policies, procedures and results need to become a part of everyday operations within an enterprise.

**Rule 7:** Define master data governance model and mechanisms with clear decision making structure, working procedures and instructions.

### b. MDM Critical Success Factors

Establishing MDM is a major initiative within an enterprise, since it crosses departmental, regional, geographic and cultural barriers. As any other business initiative, MDM implementation has several success factors:

- Scalable MDM system - in terms of geography, departments and business functions
- Master data quality must be part of the MDM – the fundamental objective of a MDM effort is to improve the quality of mater data
- MDM decision making structure – governance model adapts to varying levels of decision making within an enterprise
- MDM architecture – the choice of architecture depends on specific business problems with processes or data domains, so organization has to remain flexible and select those that best fit their business needs
- Metrics – necessary for measuring MDM efficiency and effectiveness. Metrics manage and improve the success of MDM. They help in undertaking corrections, to report on business and operational performance, to review and analyze business value of MDM.

### c. Challenges Related To Managing MDM Projects Using Scrum

MDM project typically require the formally orchestrated efforts of People (Data Governance Council, Data Stewards, etc.), Processes (business rules, data creation and maintenance workflows, data governance procedures, etc.) and Technologies (tools for collecting, consolidating, cleansing and synchronizing data). There are challenging differences between MDM projects and Scrum. Scrum assume a very simple customer/developer relationship, however major challenges in MDM projects
are people, processes and other organizational issues and considerations, and even when MDM is part of a larger data governance initiative. So communication is complex. MDM projects are not standalone software products; they have strong dependencies on other projects like BI. The project interactions and dependences need to be taken into consideration in the project planning. While Scrum focus on using time for delivering a working product, MDM has a stronger focus on meeting the requirements. With these challenging differences, there are factors that make MDM project using Scrum successful. Scrum developers closely work with final users, however in MDM project there are project owner like mediator between developers and final users. Both MDM and Scrum aim at delivering value to the customer, but using different approaches.

The answer to the question why Scrum in MDM should be used is that Scrum provides good communication, social integration, control, and coordination mechanisms that are especially useful for project such as MDM. MDM project includes enterprise wide collaboration, supports a data governance initiative, needs visible results for the business, and fast respond to ever changing business needs. The Scrum principles almost align with the principles guiding MDM. Scrum helps MDM team to understand very clearly what work needs to be done within the whole project and the specific sprint, and how to coordinate work [7]. Although Scrum provides opportunity to develop trust between team members and to maintain their motivation during the project. The trust is based on knowledge and performance. Developing knowledge about each other through the daily Scrum meetings allow team members to interact and get to know what the others are doing. Performance-based trust is supported by daily Scrum meetings, burn-down charts, and deliveries as the result of each sprint.

5. CONCLUSIONS

Business managers and IT are already familiar with the high cost and expenses incurred by low quality data. Well defined MDM system will provide: synchronization of master data across all enterprise systems, unique master records, an integrated view of a master data across the organization, improved operational efficiency and accuracy, quantitative knowledge, efficient staff that spend more time analyzing not verifying data etc. However, the question is how to implement the MDM solution successfully. Managing MDM using Scrum deploys MDM releases in increments, rapidly develop and validate (no) functional requirements, and provide proof points to the top management. More specifically, it comprises data governance model, master data quality and MDM system architecture assessment, and alignment among stakeholders. Given guidelines and defined rules can faster development of MDM system and provide more efficient usage of resources, improved operational efficiencies, improved customer satisfaction, and increased competitive advantages. However, despite the rules defined through work and practice, when it comes to MDM development and implementation, every MDM system should be customized to the specific business system.

REFERENCES


