Scrum integrated SDLC processes of Turkiye Finans IT in a COBIT compliant environment

Necmettin Özkan

Turkiye Finans Participation Bank, Istanbul, Türkiye
Necmettin.Ozkan@turkiyefinans.com.tr

Abstract. Turkiye Finans IT in 2014 managed to be the first in the Middle East and East Europe in terms of the size of the change by transforming all production and project teams and related processes to agile models from the waterfall model. The transformation included nearly 50 teams touching more than 200 people and had to take place in a highly regulated environment driven by COBIT (Control Objectives for Information and Related Technology) v4.1. Melting COBIT and Scrum in an organization is new and challenging (especially when a full compliance is required as in this case). Despite the prominent challenges, the charm of being agile pushed the Bank to go for the transformation. It has yielded the first coexistence of these two in practice. During the transformation, we have witnessed some issues coming from the integration of Scrum into Software Development Life Cycle processes while staying conformant to COBIT. This paper aims to discuss these challenges and provide solutions for them from the practice.

Keywords: Scrum, Agile, COBIT, SLDC, Banking

1 Introduction

The charm of being agile attracts many medium and even large-sized organizations. Türkiye Finans, as one of them, in 2014, started and finished the structural transformation of its IT covering nearly 50 teams with more than 200 people. Thus, the Bank managed to be the first in the Middle East and East Europe in terms of the size of this change by transforming all production and project teams and related processes to agile from waterfall models. While such a transformation touches many points of organizations as discussed in the work of [1], from people management to mindset changes the main subject of this study is alterations to Software Development Life Cycle (SDLC) processes that are inevitably at the core of the transformation and one of the most challenging parts.

The transformation had also to take place in a highly regulated environment driven by COBIT (Control Objectives for Information and Related Technology) v4.1. Melting COBIT and Scrum in an organization is new and challenging (especially when a full compliance is required as in this case). There are contrasting natures between them from the SDLC window due to the following reasons:

- As a result of co-occurrence and similarities of COBIT with rationalized, engineering-based approaches, COBIT has a potential to make Scrum's each sprint to resemble mini-waterfall in the flow.
- While agile methods discourage heavy documentation, COBIT regards documentation as a means of storing, sharing, conveying, replicating and backing-up knowledge, planning, codifying and standardizing for practice, and creating logs for further use.
- Agile teams are characterized by self-organization and Scrum trusts them to do their jobs well yet COBIT precludes this freedom inside the teams by the principle of segregation of duties.
- While COBIT is process-centric and designed to standardize people to the processes, Scrum relies on people and their creativity rather than processes.

The conflicts center on documentation requirements, their approvals, segregation of duties and the level of discipline promoted by the means of processes. Despite these prominent challenges, the need of being agile pushed the Bank to go for the transformation. Besides, it has yielded the first coexistence of these two in practice, to the best of our knowledge (i.e. based on what the literature review showed us) [1]. Thus, the main characteristics of the transformation are 1) being in a bank which is normally disciplined environment 2) being in a COBIT regulated environment which distinguishes the bank from other types 3) applying Scrum at a scalable size. During such a transformation, in particular, we have witnessed some issues coming from the integration of Scrum into SDLC processes while staying conformant to COBIT. This paper aims to deliver the description of these challenges and solutions that are already taken and on the road map.

2 Introduction to COBIT

COBIT (Control Objectives for Information and Related Technology) has domination in information technology with its more than 40 integrated standards worldwide and is itself a de-facto standard providing information technology (IT) governance model with international set of generally accepted IT control objectives to help in delivering value from IT and understanding and managing the risks associated with IT. Many countries facilitate COBIT for their public sector, governmental agencies and regulatory bodies. Specifically speaking for Turkey, on May 2006 the Banking Regulation and Supervision Agency of Turkey (BRSA) mandated that all banks operating in Turkey must adopt the COBIT's best practices when managing IT-related processes.

To name a point in COBIT in this study, the corresponding process is remarked as in the example of "AI2", control objectives as in the example of "AI2.1", and control practices as in the example of "AI2.1.3".

3 History of Agile Transformation

From process point of view, by the transformation, the changed COBIT processes are as following while the rest of the processes maintain the version before the transformation. The main changes to the Portfolio Management process are delivered in the corresponding places of this title below. When it comes to the Project Management process, we have maintained the basic body of classic project management functions considering the requirements of COBIT yet reorganized the project management roles, tools and meetings aligned with the Scrum events, roles and relevant project management tools. Among these listed COBIT processes, the focus is not on the first two, dealing with the next four SDLC related ones.

- Define a Strategic IT Plan (Portfolio Management in relevant)
- Manage Projects
- Identify Automated Solutions
- Acquire and Maintain Application Software
- Enable Operation and Use
- Install and Accredit Solutions and Changes

From the organizational point of view, with the transformation, hierarchical manager layer was removed and instead of directorates of waterfall structures in the past, a leaner organizational structure was established with agile teams. Teams are designed as containing qualifications from every profile within themselves. And they are supported by a separate enterprise architect from outside when needed. The development teams were reorganized, trained, got PSM I certificate. The training process was also conducted for all C-levels, relevant business unite and outsource resources.

The teams by default are the domain Service Scrum Teams that deal with small projects (less than 100 m/d) and problem records. For these teams, the product owners are from IT side and Scrum Masters are from the development teams. When a project is to be initiated, project members from these domain teams are selected by department managers and a dedicated project-specific project team is thus created. These created Project Scrum Teams work on master plan projects (>100 m/d), the product owner role is taken on by the business units and Scrum masters are from PMO (Project Management Office). While this type of Scrum Masters are the former project managers working in the Project Management Office for the Project Scrum Teams, they have Scrum master skills and qualifications.

IT project roadmap for the year has begun to be planned on a quarterly basis, instead of yearly in as in the past. During the prioritizing phase of projects they are classified according to their area of specialization and high-level effort estimation is performed by the relevant expert teams (T-Shirt sizing). In terms of the coverage, we apply the portfolio management to master projects (>100 m/d), excluding requirements out of projects to be aligned with the set objective of “concentrating on Master Projects with Agile Scrum teams”.

There is a committee in place in which business and IT participate to determine the prioritization of IT projects in line with business needs. While COBIT does not identify who constitutes such a committee and who directs them how, yet in our case product owners are not a member of these committees, for the sake of enterprise-wise-portfolio management.

When a selected master project starts, a product owner and Scrum master are assigned to the project and loops of Scrum come in. Thus we mainly use Scrum inside a project, at work breakdown structure level, and run Scrum events with Scrum roles inside this cycle.

4 Challenges

This part is the description of challenges faced in SDLC domain. Among the challenges, we have overcome some and some of them still remain to work on. We can basically categorize the source of challenges into two: ones from COBIT and ones from agile-way-doing-work.

For COBIT side, a rationalized, engineering-based approach has dominated software development almost since its inception and has a co-occurrence and similarities with COBIT. By this effect, in COBIT processes related to SDLC, certain document contents must be produced in a certain order and must be approved by related parties. Consider that iterative and incremental development of Scrum also means the iterative and incremental development of such software development design documents for many times including requirement specifications, high-level design, detailed design and test plans etc. It requires also formal approval cycles of these iteratively and incrementally developed documents by related business process owners and IT stakeholders. Here are the corresponding verses from COBIT side for this type:

- (AI1.3) Feasibility Study and Formulation of Alternative Courses of Action: “Define and execute a feasibility study that clearly and concisely describes the key alternative courses of action… Review the alternative courses of action with all stakeholders, and select the most appropriate one…“
- (AI1.4) Requirements and Feasibility Decision and Approval, Control Practices: “Obtain sign-off from the business sponsor and technical authority for the proposed approach, and gather feedback requiring further feasibility analysis.”
- (AI2.1) High-level Design: “Ensure that the high-level design is approved and signed off on by IT stakeholders… Submit the final high-level design after QA sign-off to the project sponsor/business process owner, and obtain approval and sign-off.”
- (AI2.2) Detailed Design: “Conduct a design walk-through with IT and business stakeholders before development is initiated, as a part of the sign-off process for the design specifications.”
- (AI2.3) Application Control and Auditability: “Confirm the design specifications for all automated application controls with IT technical authorities and business process owners, and obtain their approval and sign-off.”

- (AI2.4) Application Security and Availability: “Confirm the design of security, availability, Access management, authentication and protection of transaction integrity with IT technical authorities and, as appropriate, subject matter experts. Obtain their sign-off on and approval of the design.”

Not as a pure COBIT requirement, but as a natural result of the iterative and incremental way of agile-way-doing-work, we have witnessed some of the challenges when the system is changing organically. And COBIT, too, has points leaving no room to overlook them. Thus, we had to consider the following verses of COBIT as well:

- (AI4) Enable Operation and Use calls for knowledge transfer to business management, end users, operations and support staff and adds that “Create all the required user instructions, documentation, procedures and training materials on a timely basis to enable efficient and effective use of the new system…. and integrate any procedures with existing end-user procedures.” Similarly (AI7.8) Promotion to Production says “Promptly update all copies of system documentation and configuration information, including”.
- (AI7) Install and Accredit Solutions and Changes: “Train the staff members of the affected user departments and the Operations group of the IT function …, including business end users, IT operations, support and IT application development training, and service providers…Confirm that all test plans are approved by stakeholders, including business process owners and IT, as appropriate…Establish an implementation and fallback/back-out plan. Obtain approval from relevant parties.”

While discussion of its necessity is a separate subject, as a clear COBIT requirement is segregation of duties and what COBIT say on this is as:

- (AI2.8) Software Quality Assurance:”...ensuring that reviewers are independent from the development team”. Similarly in (AI7.6) Testing of Changes: “Ensure that the testing is designed and conducted by a test group independent from the development team.”

Each of these subjects was carefully handled during the transformation of the processes, and the general approach taken to how these challenges are addressed is communicated in this paper.

4.1 Solution: SDLC of the Bank Integrated with COBIT

As mentioned before, the tension between the two frameworks mainly come from the different approach to documentation requirements of SDLC, their approvals, segregation of duties and the level of discipline promoted by the means of processes.

Segregation of duties in quality review and testing is a clear requirement of COBIT in a way that would leave any place for a comment. Considered such clear distinction between Scrum and COBIT we are literally to follow COBIT with no options. Now, a coder is not allowed to make the functional test for her code.

For documentation requirements of SDLC and their approval, great effort was made and many things were accomplished. We had a large number of SDLC documents prepared according to the waterfall structure, resulting from the walls in communication and planning for long-terms. Thanking to building cross-functional teams, short-term iterations and resuming and handling current documents from scratch, the number of SDLC documents was reduced from 12 to 5, the number of titles in them from 175 to 38. Addition to this improvement, here are some points that are useful and leading us to the solution, as:

- It is possible to approach differently to the risk of the creation of the value and the risk of disrupting the current production environment. In this respect, the approval of design documents given by the business and IT can be taken to the end of the sprint. Because the lost value can maximum be as much as a sprint length. From this point of view, we have moved and take the approvals of design documents at end of the sprint.
- It is helpful to distinguish document requirements and their frequency according to project, release and sprint base. We prepare one high level design document per project, functional and technical design documents on sprint based and production related ones on released based. However, all design documents are iteratively and incrementally created and maintained per project.
- We have placed Sprint Zero step to identify and remove possible uncertainties around project scope, cost, schedule, and technical strategy. There, adequate grooming is conducted that is required for launching Sprint 1. Apart from getting the big picture in design, this step has fortunately reduced the documentation overhead at a level.

Based on these, the following pictures (Fig.1, Fig.2 and Fig.3) depict the resulted solution from different angles. Fig.1 points out the COBIT approach and our solution in terms of SDLC document requirements flow (B: Business, T: IT, PO: product owner, DA: domain architect, EA: enterprise architect, DT: development team).

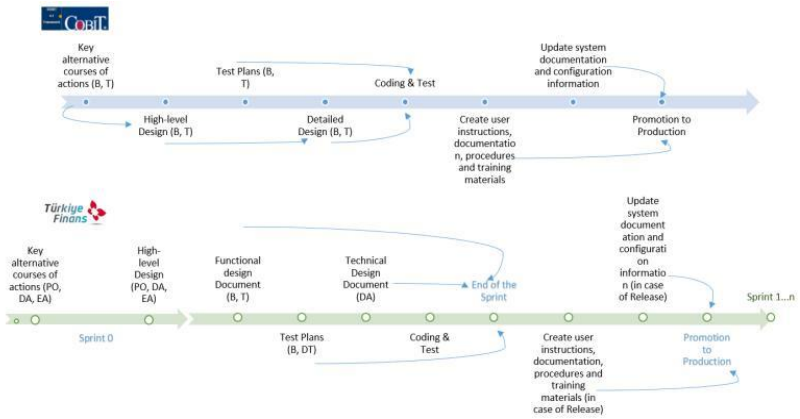

Fig. 1. SDLC document requirements of the bank and COBIT

Fig.2 illustrates documentation and approval details in the bank. It shows that small projects (0<x<100 m/d) are divided into two categories: 0<x<10 m/d and 10<x<100 m/d. And they are regarded differently in terms of design documents as shown. We use the same templates for master projects and for small projects in the range of 10-100 m/d (SBL: Sprint Backlog List, PBL: Product Backlog List).

High-level Solution Design	High-level Design	Project based	Domain Expert, PO, Enterprise Architect, IT Information Security Expert ve Development Team	x	-	-
Sprint0 presentation	-	Project based	-	x	-	-
PBL	-	Project /Team based	In Sprint0, approved within High-level Solution Design In Sprint Planning Meetings agreed upon by Development Team and PO	x	x	x
SBL	-	Sprint based	agreed upon by Development Team, (IT) PO	x	x	x
Functional Design Document	Detailed Design	Document is project based/approval is sprint based	**For Master Projects:** Domain Expert, PO and related business units **For Small Projects** Domain Expert or IT PO and related business units	x	x	-
Technical Design Document		Document is project based/approval is sprint based	**For Master Projects: Domain Experts,** or IT Department Vice Presidents, (if data model changed) Enterprise Architect **For Small Projects** ; Domain Expert or IT PO	x	x	-
Software Quality Assurance Plan	Software Quality Assurance Plan	Sprint based	Scrum Master (for master projects) IT PO (for small projects)	x	x	x
Code Review Check List	-	Sprint based (if coded)	Anyone different from the coder	x	x	-
Implementation Plan	Implementation Plan	Release based	Relevant parties	x	x	-
Fallback/backout plan	Fallback/backout plan	Release based	Relevant parties	x	x	-
End-user manuals	End-user procedure manuals	Release based	-	x	x	x
Training	Knowledge Transfer to Business Management, End Users and Operations and Support Staff	Release based	-	x	x	x

Fig. 2. Documentation and approval details in the bank

Fig.3 is the class diagram of SDLC design documents in order especially to visualize entity relations. In our case, in each sprint, useable and potentially releasable product increment is not created especially for scaled sized projects. Thus, it is possible not to have a full SDLC iteration, let's say a particular sprint may be full of coding and testing only. And, this is why design documents are shown with 0-1 relation with a sprint.

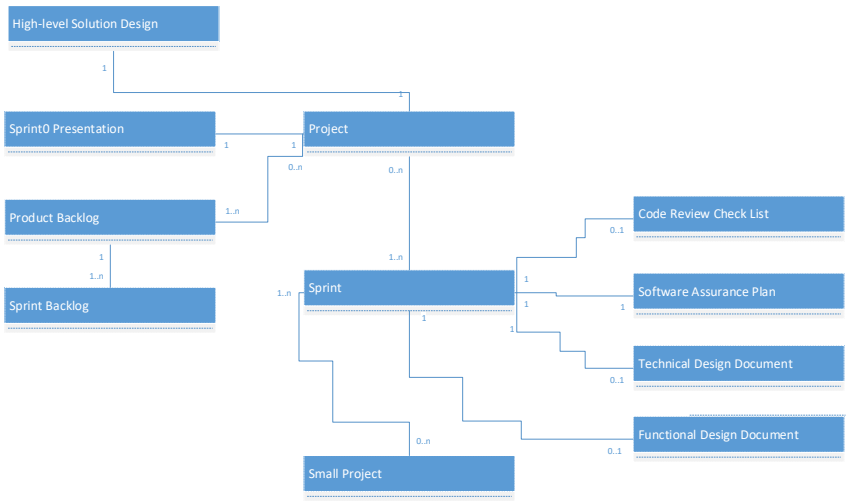

Fig. 3. Class Diagram of SDLC Design Documents

Organic growth of the system means the creation and integration of complete, accurate and usable supporting documents (AI4.2) with promptly updates to the existing environment in production for the use of end users (AI4.3), operations and support staff (AI4.4), and business management (AI4.2). If required, adequate training must be provided to the related people including business end users, IT operators, support and IT application development teams, and service providers (AI7.1.2) to learn how to use the new and continuously changing system.

For the sake of the nature of work in any promotion to the production and for COBIT for sure, we have included all COBIT requirements. It is clear these types of requirements are free from any kind of methodology as the nature of work calls for it. However, different from the development design documents, these are organized with release frequency that must be done only in corresponding sprints before the transition rather than each sprint.

Organic growth of the system also requires continuous integration. In this case, without a right balance between automated scripted tests and interactive user testing this iterative, incremental and continuous testing can be hard, time-consuming and risky. This is why the need of test automation that is in the Bank's short term plans manifests for the regression tests of increments that are part of large and interdependent applications.

In a rapid and more frequently promotion to the production, establishment of secure and sanitized test environments as a representative of the future operating landscape with the deployment frequency, which Scrum anticipates is important. For this reason, we have increased the frequency of updating the test environments and have made improvements with more frequently updated data. And, for the time being, we have started work on “dev-ops” to fasten and smooth this process.

5 Summary and Discussion

We live with more dynamic processes compared to the past. Additionally, we had to comply with COBIT and as a result, we did so substantially. These factors resulted in more documentary transactions and keeping documents fresher and updating more frequently compared to the past. We know the use of tools can be helpful to manage such frequent behavior on documents side. This is the main reason for the Bank's short term plans in the drifting of the Team Foundation Server (TFS) Agile Center into widespread use in this context. One way or another, the nice side is that the employees are fresher and more up-to-date in terms of document preparation knowledge and awareness.

The room for flexibility in such an adaptation has been in tailoring Scrum with customizations and extensions such as Sprint Zero and managing risks in a Scrum ways-of-doing work. In doing so, it is useful to distinguish between the risk of losing potential value and the risk of disrupting the production environment. Meanwhile, we continue to be in touch with the government regulatory body to apply COBIT in a more flexible way.

We hope that practitioners will benefit from the challenges, issues witnessed, solutions created during our transformation experience and that our findings will highlight the need for the coexistence of agile and disciplined approaches, and be another input into efforts to do so.

References

1. Ozkan, N., Tarhan, A., Kucuk, C.: Scrum at Scale in a COBIT Compliant Environment: The Case of Turkiye Finans IT. In: 18th International Conference on Agile Software Development, XP, agilealliance.org, (2017)
2. Ozkan, N.: Risks, Challenges and Issues in a Possible Scrum and COBIT Marriage. In: 22nd Asia-Pacific Software Engineering Conference, APSEC, December 1 - December 3, pp. 111-118. IEEE Computer Society, Washington, DC (2015)