How Can ITIL and Agile Project Management Coexist?

An Adaptation of the ITIL V.3 Life Cycle in Order to Integrate SCRUM

Bertrand Verlaine(✉), Ivan Jureta, and Stéphane Faulkner

PReCISE Research Center, Department of Business Administration,
University of Namur, Rempart de la Vierge 8, 5000 Namur, Belgium
{bertrand.verlaine,ivan.jureta,stephane.faulkner}@unamur.be

Abstract. A significant proportion of organizations delivering IT services follows and combines some IT management frameworks. At the organizational level, they often act in accordance with ITIL, the most used IT Service Management (ITSM) framework. At the project management level, a growing part of them are willing to work with agile methods. However, ITIL favours the Waterfall life cycle, such as in PRINCE2 or PMBoK, to the detriment of agile methods. Nevertheless, it is also assumed that ITSM best practices have to be adapted to, e.g., the environment, the kind of IT services and the culture of IT organizations. So there is a legitimate issue to raise: How can ITIL v.3 and agile project management coexist in an IT organization?

In this paper, we positively answer to this question by describing ***how*** to adapt ITIL v.3 when it is associated with SCRUM, the most popular agile method. First, we detail the current ITIL structure when a software implementation project is carried out. Then, we identify and explain which are the ITIL elements to modify in comparison with Waterfall-based project management methodologies. Lastly, we describe and illustrate eight interfaces between ITIL v.3 and SCRUM.

Keywords: It service management · ITIL v.3 · Agile methods · SCRUM

1 Introduction

Organizations delivering IT services often apply an IT Service Management (ITSM) framework. It describes how to manage quality IT services that meet the business needs "through an appropriate mix of people, process and information technology" [1]. In this paper, we refer to ITIL, which is often considered as the most used ITSM framework.

Besides the management of the organizational processes, IT organizations also have to manage their projects. Some of them consist of implementing software. In recent years, these kinds of projects are more and more often managed thanks to agile methods. This means that they respect the *agile manifesto* and its twelve principles [2]. Taken as a whole, *agile* corresponds to "the continual readiness of an information systems development method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value [...], through its collective components and relationships with its environment" [3]. There are several agile methods: eXtreme Programming (XP) [4], SCRUM [5] and Feature-Driven Development [6], to name a few of them. Each adheres more or less strongly to the agile manifesto and principles. Presently, SCRUM is the most popular agile method in the IT industry [7]; this justifies its use in the scope of this paper.

If we compare agile methods and ITIL v.3 from a conceptual and structural point of view, they could not be more different. Agile methods are seen as loose and very flexible, while ITIL is rather considered as bureaucratic and procedural. However, they share a common objective: both of them aim at providing business value to its customers and users. As first and main principle, the agile manifesto states that its "highest priority is to satisfy the customer through early and continuous delivery of valuable software" [2]; ITIL v.3 "is adopted by organizations to enable them to deliver value for customers through IT services" [8]. However, the way of doing is very dissimilar. The former provides value by quickly implementing what customers and users really want, while the latter provides value by delivering stable IT services respecting the negotiated set of functionalities and the quality levels. The agile manifesto favours informal interactions between the project stakeholders as opposed to a high formalism, which is advocated by ITIL v.3. Therefore, the following research issue is naturally raised.

How can ITIL v.3 and agile project management coexist in an IT organization?

Contributions. In this paper, we analyse the ITIL v.3 best practices by focussing on the relations between them and the structure of software project management methods. We conclude that ITIL favours methods which are based on the Waterfall life cycle or, at least, methods which recommend a full software design before its implementation. Based on four recommendations in order to adapt ITIL v.3 to the principles of the agile manifesto, we discuss the alignment of SCRUM and ITIL v.3. We identify, define and justify eight interfaces between them, as well as the modifications to the ITIL v.3 life cycle in order to enable agile project management. This work aims at helping people wishing to follow the ITIL v.3 best practices associated with SCRUM in their IT organizations.

Organization. First, we present ITIL v.3, its point of view on software project management, and SCRUM (Sect. 2). Then, in Sect. 3, we describe the relations between ITIL v.3 and SCRUM while emphasizing the adaptations to ITIL v.3. Finally, after the related work (Sect. 4), the conclusion and some directions for future research are discussed (Sect. 5).

2 Presentation and Discussion of the Reference Frameworks

In this section, we describe the two main references used, namely ITIL v.3 (Sect. 2.1) and SCRUM (Sect. 2.4). In Sect. 2.2, we describe the relations between the relevant ITIL processes and the project management structure advocated by ITIL. In Sect. 2.3, we make four recommendations in order to adapt ITIL v.3 to an agile project management method.

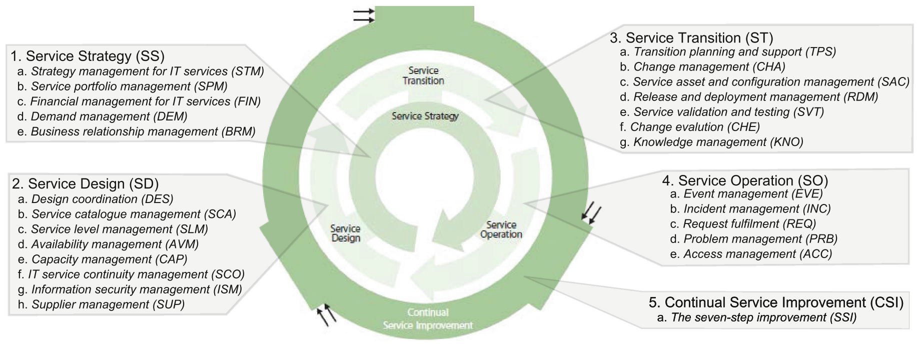
1. Service Strategy (SS)
a. Strategy management for IT services (STM)
b. Service portfolio management (SPM)
c. Financial management for IT services (FIN)
d. Demand management (DEM)
e. Business relationship management (BRM)
2. Service Design (SD)
a. Design coordination (DES)
b. Service catalogue management (SCA)
c. Service level management (SLM)
d. Availability management (AVM)
e. Capacity management (CAP)
f. IT service continuity management (SCO)
g. Information security management (ISM)
h. Supplier management (SUP)
3. Service Transition (ST)
a. Transition planning and support (TPS)
b. Change management (CHA)
c. Service asset and configuration management (SAC)
d. Release and deployment management (RDM)
e. Service validation and testing (SVT)
f. Change evalution (CHE)
g. Knowledge management (KNO)
4. Service Operation (SO)
a. Event management (EVE)
b. Incident management (INC)
c. Request fulfilment (REQ)
d. Problem management (PRB)
e. Access management (ACC)
5. Continual Service Improvement (CSI)
a. The seven-step improvement (SSI)
Service Transition
Service Strategy
Service Design
Service Operation
Continual Service Improvement

Fig. 1. ITIL v.3 life cycle (based on an official illustration of ITIL [9])

2.1 A Short Description of ITIL V.3

ITIL is a collection of ITSM best practices which requires to focus on the customers and users and, more specifically, on the value that they get by using IT services [10]. A recent empirical study [11] demonstrates that ITIL strongly helps companies delivering IT services to improve their processes and to increase their benefits. ITIL[1] v.3 is structured into five phases, each being composed of processes. An ITIL process is defined as "a structured set of activities designed to accomplish a specific objective. A process takes one or more defined inputs and turns them into defined outputs" [9]. These five ITIL v.3 phases along with their processes are depicted in Fig. 1 and described below.

– **Service Strategy:** This phase aims at guiding the whole IT service creation and delivery strategy through an effective management of its life cycle. Topics covered are mainly the creation and management of the IT strategy, the analysis and the development of the markets (including the internal customers), the feasibility analysis related to the conception of IT services and the financial management [9].

[1] The third version of ITIL was released in 2007 and revised in 2011. In this paper, each time we mention ITIL v.3, we refer to its last version.

- **Service Design:** This phase provides the guidance for the conception of services and their future management [1]. The conception lies in defining how the organization assets will be transformed to bring value to customers through the use of the future IT services. This phase also defines how to improve the existing IT services.
- **Service Transition:** This phase explains how to organize the implementation of the designed services and how to manage the changes applied in existing services [12].
- **Service Operation:** This phase focuses on the delivery and support of IT services [13].
- **Continual Service Improvement:** This phase aims at maintaining and improving the value delivered by IT services to its users and customers through the measurement and the analysis of the service solutions and the processes followed [14].

2.2 ITIL V.3 Viewpoint On, and Relations With, Project Management

ITIL defines the notion of project as "a temporary organization, with people and other assets, that is required to achieve an objective or other outcome. Each project has a lifecycle that typically includes initiation, planning, execution, and closure" [12, p. 324]. This definition is very similar to the ones proposed in the PMBoK [15] or in other related references (e.g., [16,17]). The notion of Project Management Office (PMO) is another important notion in the scope of this work. From the ITIL point of view, the purpose of the PMO is "to define and maintain the service provider's project management standards and to provide overall resources and management of IT projects" [1, p. 254]. Actually, it is a combination of two processes, namely, *Design coordination* and *Transition planning and support*. The former focuses on the creation of the service design package while the latter focuses on the effective implementation of IT services and of their modifications. A quick look to the PMBoK indicates that this notion is similarly understood: a PMO is "a management structure that standardizes the project-related governance processes and facilitates the sharing of resources, methodologies, tools, and techniques" [15, p. 10].

Concerning the project management, ITIL suggests to combine its best practices with a methodology such as PRINCE2 or PMBoK [12, p. 324]. Although ITIL aims at managing the whole life cycle of an IT service, the projects leading to the effective creation or modifications of IT services are not directly managed through the ITIL processes. This is consistent with the project notion; the PMBoK explains that "a project is a temporary endeavor undertaken to *create* a unique product, service[2] or result" [15]. Actually, ITIL supports this creation by, e.g., providing information, but it not *directly* manages projects. In the ITIL Service Design book [1, Sect. 3.11.3], the project management in an iterative,

[2] The notion of service in the PMBoK is different from the notion of service in ITIL v.3, but this discussion is out of the scope of the paper.

incremental and/or adaptive life cycle is briefly discussed. The Rapid Application Development (RAD)[3] [18] and XP [4] methods are mentioned. However, there is no explanation concerning the way for combining one of these two methods with ITIL. The structure of the ITIL v.3 processes, as illustrated by the left side of Fig. 2, does not fit with agile methods. Indeed, this structure corresponds to a Waterfall life cycle, also called "conventional development" in the ITIL literature: the requirements are elicited, then the system-to-be is specified, which leads to its implementation and, lastly, to its testing and deployment. This is illustrated by Fig. 2 and discussed hereafter. The legend of Fig. 2 is available in Table 1, and is also applicable to Figs. 3 and 4.

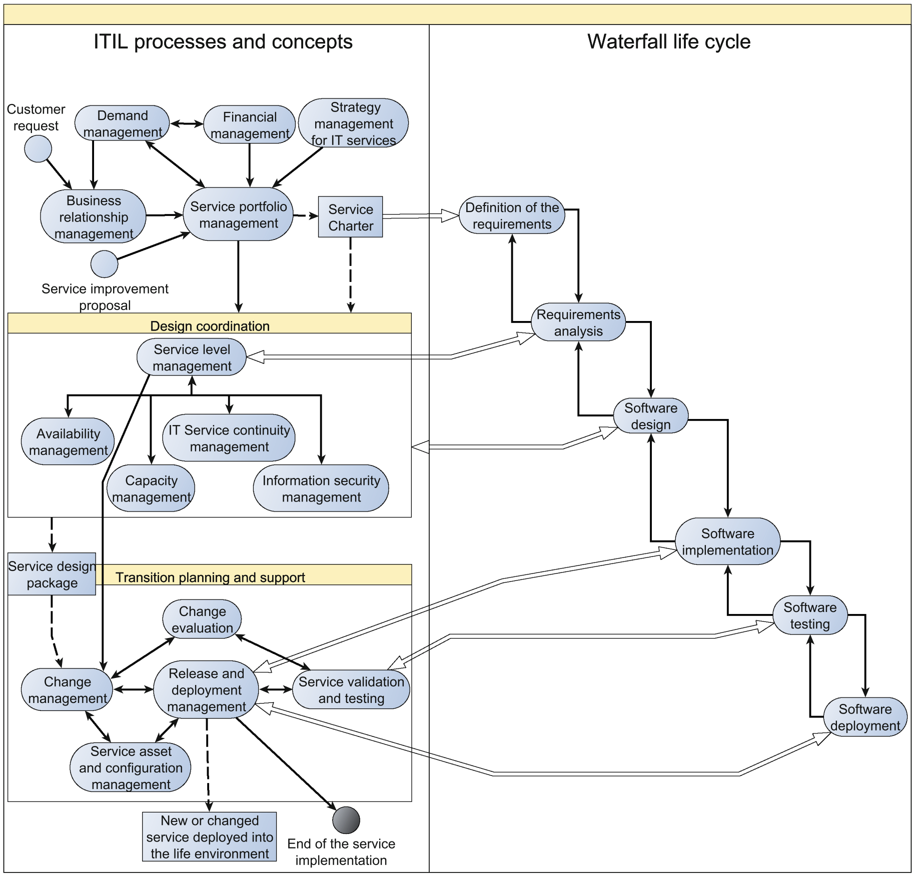
ITIL processes and concepts
Waterfall life cycle
Customer request
Demand management
Financial management
Strategy management for IT services
Business relationship management
Service portfolio management
Service Charter
Service improvement proposal
Design coordination
Service level management
Availability management
IT Service continuity management
Capacity management
Information security management
Service design package
Transition planning and support
Change evaluation
Change management
Release and deployment management
Service validation and testing
Service asset and configuration management
New or changed service deployed into the life environment
End of the service implementation
Definition of the requirements
Requirements analysis
Software design
Software implementation
Software testing
Software deployment

Fig. 2. Relations between the ITIL v.3 processes and the Waterfall life cycle

[3] RAD is not always considered as fully agile, but this discussion is out of the scope of the paper.

Table 1. Legend applicable to Figs. 2, 3 and 4

Form	Description
Nodes	
Light circle	It is a trigger of the flowchart
Dark circle	It depicts the end of the flowchart
Rectangle with rounded edges	It is a set of tasks to be performed, which corresponds to the rectangle label; this is a process in ITIL v.3, otherwise this is an activity in the scope of project management
Rectangle	It is a set of elements produced, which are grouped and labelled with a single name
Rectangle containing other nodes	It represents a coordinating process for other ITIL processes
Diamond	It denotes that an exclusive choice should be made.
Arrows	
Arrow with continuous line	It indicates the sequence of the set of tasks
Arrow with dashed line	It shows the transmission of sets of elements between activities and/or processes
Arrow with double line	It indicates a unidirectional or a bidirectional transmission of information between a ITSM process and a project management activity. It is named an **Interface** and is numbered for the relations between ITIL v.3 and SCRUM (see Fig. 4)

Description of the IT Service Implementation Flow. The left side of Fig. 2 depicts the ITIL processes and relations when a new IT service is created or when an existing IT service is reworked. As a reminder, these processes are mentioned in Fig. 1 within their belonging phase. There are two possible starts. The first one lies in a customer request managed by the *Business relationship management* process; the second one is a positive decision concerning an IT service improvement proposal. In both cases, the main starting process is *Service portfolio management*. It receives the information provided by the other processes of the *Service strategy* phase. Its main output is a *Service Charter*, i.e., "a document that contains details of a new or changed service" [9, p. 452], which is sent to the service design phase. The latter has to support the creation of the *Service design package*. It is a (set of) "document(s) defining all aspects of an IT service and its requirements through each stage of its lifecycle[, which] is produced for each new IT service, major change or IT service retirement" [1, p. 418]. At the organizational level, the design tasks are coordinated by *Design coordination*. The other processes are used to provide information about the expected service level and the four quality properties; these information exchanges are managed by *Service level management.*

Once created, the *Service design package* is transferred to *Change management* for its implementation, which is coordinated by *Transition planning and support.* The workflow is as follows. First, the *Service design package* is transformed into several requests for change, which is "a formal proposal for a change to be made [on a configuration item, which contains the] details of the proposed change" [1, p. 415]. Once authorized, they are grouped under a service release and, then, they are effectively implemented. *Service validation and testing* is in charge of validating the deployment of the service release. This leads to the *Service operation* phase, which is out of the scope of this paper.

The right side of Fig. 2 depicts the Waterfall model [19,20], which is the main project life cycle model behind the structure of PRINCE2 and of PMBOK: these methods recommend to design first the software and, then, to implement it. That is, if the software specifications are sufficiently detailed after an in-depth requirements analysis, it will be coded such as. In this context, we consider the software engineering as a linear process with possible steps backward. The first steps focus on the requirements engineering and their specifications (see the right side of Fig. 2). Then, the specified software is coded and tested. The last step is the software deployment. One should move to the next step only when the previous step is achieved, although a step back is always possible in order to improve or to remake a previous output. However, returning to a former step often involves additional costs and delay. The lack of contact with the stakeholders is also a drawback often underlined when using this software engineering model.

In Fig. 2, the arrows with a double line illustrate the relations between the ITIL processes and the steps of the project management workflow. The first information transfer occurs when the *service charter* is communicated to the step *Definition of the requirements*, so that the software engineers can elicit the requirements and design the future software during, respectively, the steps *Requirements analysis* and *Software design.* This specification work is facilitated and supervised by the processes of the service design phase, coordinated by the *Design coordination* process. The next step, *Software implementation*, lies in the coding of the designed software by respecting the service release previously defined. This leads to its test through the step *Software testing* and, lastly, to its deployment through the step *Software deployment.*

2.3 ITIL Associated with an Agile Project Management Method: Recommendations

To be considered as agile, a project management method has to respect the agile manifesto [2] and have to be incremental, iterative and adaptable [21]. Based on the observations made in Sect. 2.2, we make four recommendations in order to associate ITIL with an agile project management method. Note that ITIL v.3 adaptations are allowed if we respect its main principles and ideas. Indeed, the ITIL content is not prescriptive, in the sense that IT organizations are encouraged to adopt the main ITSM principles and "to adapt [the best practices] to work in their specific environments in ways that meet their needs" [12, Sect. 1].

1. The *Service design package* has to be created incrementally along with the software implementation and testing.
2. The software development team should be able to modify the *Service design package*, and thus the requests for change, even when the implementation has already begun.
3. The phases *Service design* and *Service transition* have to be conducted in parallel.
4. Service managers have to progressively release their IT services, on the same pace that the underlying software implementation is carried out.

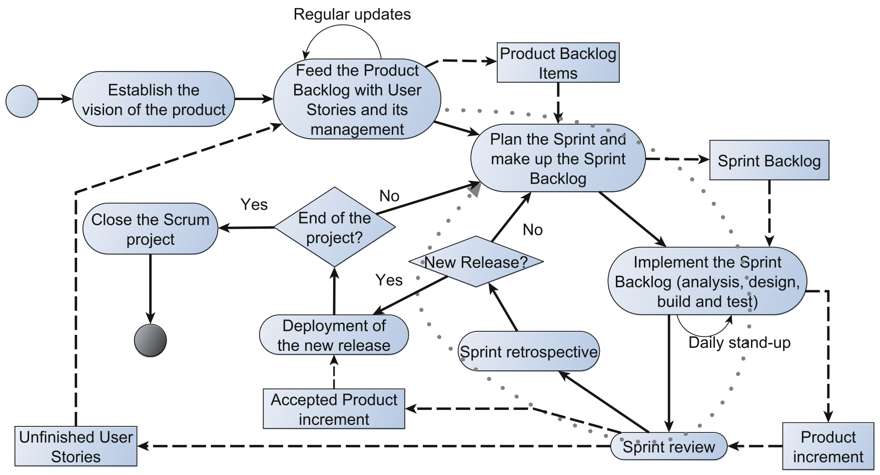
Regular updates
Establish the vision of the product
Feed the Product Backlog with User Stories and its management
Product Backlog Items
Plan the Sprint and make up the Sprint Backlog
Sprint Backlog
Close the Scrum project
Yes
End of the project?
No
No
New Release?
Yes
Implement the Sprint Backlog (analysis, design, build and test)
Daily stand-up
Deployment of the new release
Sprint retrospective
Accepted Product increment
Unfinished User Stories
Sprint review
Product increment

Fig. 3. The SCRUM process framework (based on [5,22,23])

2.4 An Introduction to SCRUM

SCRUM is an agile process framework for managing software development projects. It consists of roles, events, rules and artefacts [24]. Its main objectives are the transparency, the inspection and the adaptation during the software implementation [5,24]. Figure 3 illustrates the SCRUM framework and its steps, which are described hereafter[4].

When a SCRUM project starts, the vision of the product, which corresponds to the future software to be created, is determined. The client and the SCRUM team have to write down how the future product is going to support the client's organization strategy. This is the essence of the business value that will be got from the product use. In particular, this includes the definition of the targeted users and

[4] For more details about SCRUM, the reader can refer to [5,22].

customers of the product, the main use situations addressed by the product, the business model, its "must-have" characteristics, its desired qualities and a comparison with the possible competing products [22]. Note that the SCRUM team is composed of (i) the product owner—s/he is in charge of the product backlog management—(ii) the SCRUM master—s/he is responsible for ensuring that the SCRUM team correctly follows the SCRUM philosophy and its rules, without any hierarchical relation—and (iii) the development team—the latter is composed of professionals who carry out the work leading to the product implementation[5]. All SCRUM teams have to be self-organized and cross-functional.

The second SCRUM step is the feeding of the product backlog, which is a prioritized list of requirements. It is managed by the product owner. It is continually updated even after the start of the product development, hence the fact that SCRUM is an adaptive software project management method. Product backlog updates come from the customer or from the Sprint review (see further for a description of this SCRUM event). These updates include modifications, additions and removals of product backlog items.

Once the product backlog is defined and prioritized, a first Sprint may be organized. The dotted line in Fig. 3 represents its life cycle. This SCRUM iteration is a fixed period of time (about four weeks) during which a subset of the product backlog, called the Sprint backlog, is analyzed, designed, implemented and tested. A Sprint begins with its planning and ends with the Sprint review and, after that, the Sprint retrospective. The Sprint review is an inspection of the product increment created during the Sprint. This inspection is achieved by the customer and the product owner with the help of the development team. The second meeting, i.e., the Sprint retrospective, is held by the SCRUM team to inspect and to improve their way of working for the next Sprints.

After the Sprint review and the Sprint retrospective, the customer and the product owner decide if the product increment is deployed as a new product release, and thus made immediately usable. Then, the customer has to determine if the current version of the product meets his expectations, or if an additional Sprint is needed in order to integrate some more backlog items to the product and/or in order to improve it. The closing of the SCRUM project corresponds to, i.a., the creation of the documentation.

3 Coexistence of ITIL and SCRUM: Alignment and Description of the Interfaces

In this section, we describe the alignment between SCRUM and ITIL v.3 as well as all the interfaces between them through which information exchanges occur. In Table 2, we sum up these interfaces, which are numbered in Fig. 4. In this picture, we illustrate how to combine ITIL v.3 with SCRUM. The left part of Fig. 4 contains some differences compared to the left part of Fig. 2 illustrating

[5] In the scope of this paper, we consider that the development team includes the SCRUM master.

the structure of ITIL v.3. These differences, detailed hereafter, are due to the recommendations made in Sect. 2.3. The right side of Fig. 4 is exactly the same than Fig. 3 illustrating the SCRUM method.

At the beginning, there is not difference compared to the traditional life cycle of ITIL v.3 detailed in Sects. 2.1 and 2.2: the *Service strategy* phase is covered as usual. Afterwards, there is a key difference in comparison with the traditional structure of ITIL v.3 (cf. Fig. 2). There is no more a direct relation with the phase *Service design* before the start of the software implementation projects. In other words, the *Service charter* is sufficient to start a SCRUM project. The *Service charter* document is used to establish the vision of the product in SCRUM (see **Interface** 1 in Fig. 4 and its summary in Table 2). Of course, it is also transferred to the next ITIL phase and, more precisely, to *Service level management*. This difference is explained by the fact that the service design will be progressively achieved along with the carrying out of Sprints[6].

Then, the product backlog has to be created and fed based on the *Service charter* and on the communication between the product owner and the identified stakeholders. For utility and quality considerations, information provided by *Service level management* should be used (e.g., service improvement opportunities, service quality plans, reports on operational level agreements and underpinning contracts, and so on [1, p. 121]). This is captured by **Interface** 2, which is referenced by its number in Fig. 4 and in Table 2. After the next SCRUM step, i.e., the Sprint planning, the first Sprint starts.

During a Sprint, the Sprint backlog is implemented; this includes the analysis, the design, the coding and the testing of the objective(s) introduced in the Sprint backlog. There are several information exchanges with ITIL processes during a Sprint. The first main concerns *Design coordination* (cf. **Interface** 3). Indeed, for analysis and design tasks achieved during the Sprints, this ITIL process coordinates the transfer of knowledge about the design of a service—or of a service modification—in order to reach the adequate service level. The second main information exchange involves *Change management* (cf. **Interface** 4). Indeed, the designed changes made during a Sprint have to be assessed and authorized by the Change Advisory Board (CAB), which is "a group of people that support the assessment, prioritization, authorization and scheduling of changes" [12, p. 306]. In the context of our work, the scheduling of changes is automatically determined: the authorized changes are carried out during the current or the next Sprints. Similarly, it is not relevant to prioritize changes achieved during projects seeing that it is an ITIL artefact applied for operational changes.

A second key difference is the removal of the ***direct*** relation between *Service level management* and *Change management*, i.e., between the ITIL phases *Service design* and *Service transition*. In the common ITIL structure, the phase *Service design* ends with the completion of the *Service design package*, which is

[6] If other types of configuration items than software, activities or processes have to be modified, and thus specified in a *Service design package*, a parallel project should be carried out or it could be included in the agile project. This possibility is left for future work (see Sect. 5).

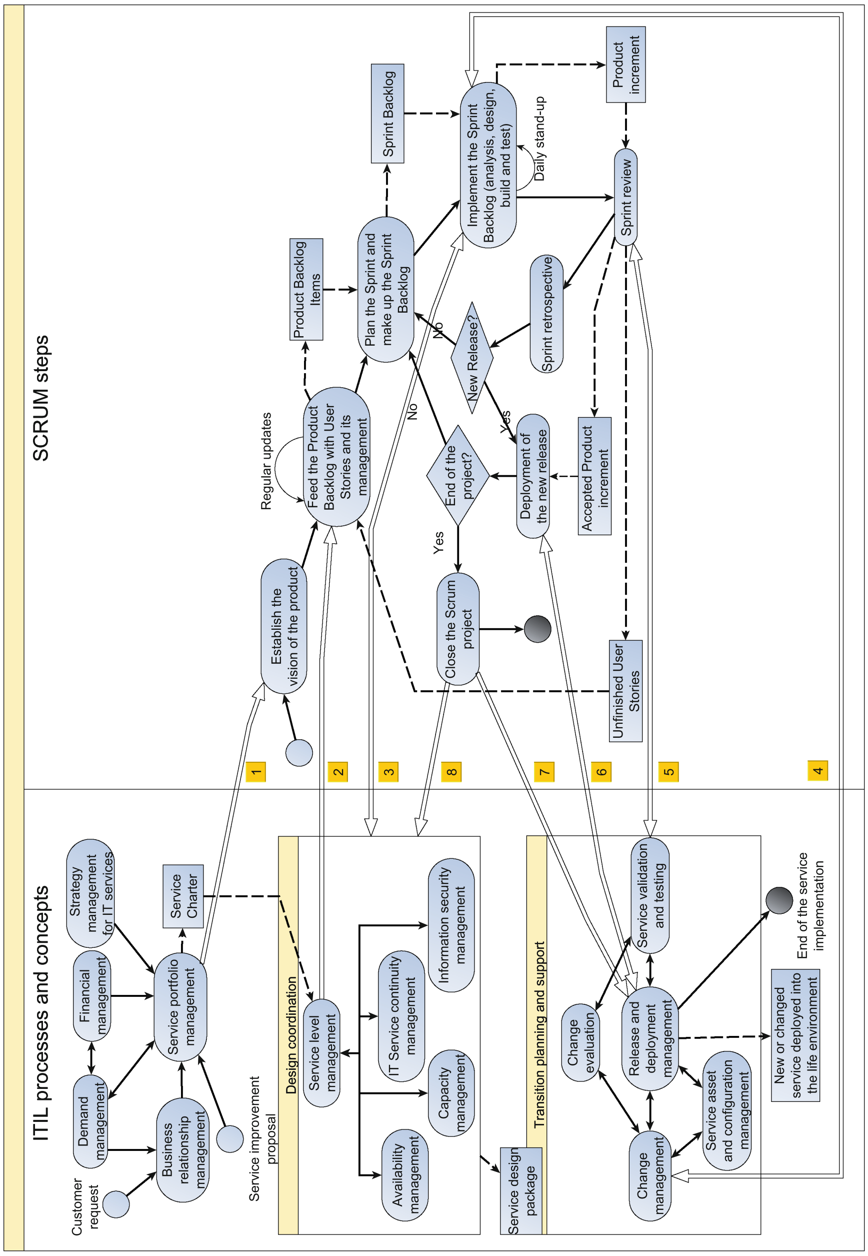
ITIL processes and concepts
SCRUM steps
Customer request
Demand management
Financial management
Strategy management for IT services
Business relationship management
Service portfolio management
Service Charter
Service improvement proposal
Design coordination
Service level management
Availability management
Capacity management
IT Service continuity management
Information security management
Service design package
Transition planning and support
Change management
Change evaluation
Release and deployment management
Service validation and testing
Service asset and configuration management
New or changed service deployed into the life environment
End of the service implementation
1
2
3
4
5
6
7
8
Establish the vision of the product
Regular updates
Feed the Product Backlog with User Stories and its management
Product Backlog Items
Plan the Sprint and make up the Sprint Backlog
Sprint Backlog
Implement the Sprint Backlog (analysis, design, build and test)
Daily stand-up
Product increment
Sprint review
Sprint retrospective
New Release?
No
Yes
End of the project?
No
Yes
Deployment of the new release
Accepted Product increment
Close the Scrum project
Unfinished User Stories

Fig. 4. Alignment of, and interfaces between, ITIL v.3 and SCRUM

Table 2. Description of the interfaces between ITIL v.3 and SCRUM; the numbers (#) correspond to the digits used to identify the interfaces depicted in Fig. 4

#	*From*: Information transferred	*To*
1	*Service portfolio management*: Service charter	*Establishment of the vision of the product*
2	*Service level management*: Service quality plan & Service improvement opportunities	*Product backlog management*
3	*Design coordination*: Service portfolio updates & Revised enterprise architecture	*Sprint*
	Sprint: Documentation obtained from the analysis and design tasks & Identified constraints	*Design coordination*
4	*Change management*: Authorized and rejected RFCs & Changes to services and infrastructure	*Sprint*
	Sprint: Software modifications to assess & Sprint planning and updates	*Change management*
5	*Service validation and testing*: Configuration baseline of the testing environments & Analysis of tests results	*Sprint review*
	Sprint review: User stories considered during the Sprint & Software modifications achieved	*Service validation and testing*
6	*Release and deployment management*: Deployment plan, Service notification, Tested environments and facilities & Updates for the release and deployment activities	*Deployment of the new release*
	Deployment of the new release: Reviewed and accepted product increment	*Release and deployment management*
7	*Scrum project closing*: Final version of the product created	*Release and deployment management*
8	*Scrum project closing*: Functional, technical and user documentation created	*Design coordination*

then transferred to the *Change management* process in order to be transformed into *Requests for change*. When associating ITIL v.3 with SCRUM, the activities of *Service design* and *Service transition* are conducted in parallel. The *service design package* is thus gradually created as the number of Sprints increases. As a reminder, one of the agile principles favors "working software over comprehensive documentation" [2], but this does not mean that there is no more documentation produced [24]. This remains indispensable for, e.g., the IT service operation management or its further modifications [13].

Once a Sprint is ended, the next SCRUM step is the *Sprint review* during which the development team, the product owner and the stakeholders inspect the newly created product increment. This step is supported by the process *Service validation and testing* (cf. **Interface** 6). In particular, it provides automated testing solutions to support the *Sprint review*. If the product increment is validated for release, it is deployed thanks to the support of the process *Release and deployment management* (cf. **Interface** 5). It is important to automate the deployment of software seeing that this activity proves to be more frequent with an agile project management method.

After the *Sprint retrospective* and, if so decided, after the deployment of the product increment, a new Sprint can be planned. Note that, along with the Sprints execution, the product backlog is regularly updated, always with the support of the process *Service level management* (cf. **Interface** 2).

Once the last version of the product fulfils the stakeholders' expectations about the software-related utility and warranty provided, the SCRUM project may end by its final deployment (cf. **Interface** 7). During this last SCRUM step, the *Service design package* is finalized according to the operational conventions prescribed by ITIL (cf. **Interface** 8). The objective is to make accessible the IT service description, including the documentation created during the Sprints, in a single and comprehensive source of information. It is indispensable for further support and maintenance activities carried out during the operation activities structured around the ITIL phase *Service operation*.

4 Related Work

Several research works deal with ITSM frameworks and agile principles in order to integrate the latter into the ITSM processes and environments (e.g., [25,26]). These works are quite far away from the topic tackled, seeing that we study the consequences of associating an ITSM framework with an agile project management method.

In [27], there is a description of how to manage projects and services in an agile way. To do so, they explain how to combine three frameworks: ITIL v.3 edited in 2007, PRINCE2 and DSDM Atern (Dynamic Systems Development Method). The latter is an agile project method for delivering software. This book does not directly describe the relations between ITIL and an agile project management framework, seeing that PRINCE2 is used as an intermediary between them.

In [28], Pollard et al. argue for a deeper focus on ITSM issues, including the consequences of new project management methods, such as agile methods, on the common ITSM structure and operations. They underline the need for solutions taking into account both highly structured ITSM frameworks and agile frameworks, which provide minimal guidelines according to the authors. In the same line, [29] is a claim for combining ITSM frameworks with agile methods in order to improve the business and IT alignment (BIA). However, the author does not explain how. He focuses on the technological issue in BIA through an agile service provisioning system based on the principles of the service-oriented paradigm.

In [30], the authors describe how to apply SCRUM in the IT support, which is one of the components of an ITSM. This work covers the operational processes of ITIL v.2 and discusses the use of SCRUM during the maintenance and incident management, and when a small modification to an existing service has to be carried out.

Another related paper contains a large discussion about ITIL v.3 edited in 2007 and software implementation methodologies [31]. A part of the discussion is about agile software implementation methodologies. They stay at a very high level without describing practically how to integrate ITIL v.3 with an agile method. Nevertheless, the observations made in this paper are similar to our four recommendations (see Sect. 2.3).

Lastly, [32] reports an experiment about the use of XP combined to ITIL v.2, and concludes that agile methods share more similarities with ITIL than often believed.

5 Conclusion and Future Work

How can ITIL v.3 and agile project management coexist in an IT organization? By tackling this research issue, we argue for an adaptation of the most used ITSM framework, ITIL v.3, in order to facilitate its coexistence with an agile project management method, SCRUM. Although ITIL and agile methods share the same main objective, i.e., providing business value to customers and users, the way of doing is very different. This is explained by their respective structure. Basically, ITIL favours the complete design and specifications of an IT service before starting its implementation. Unlike ITIL v.3, agile methods favour a parallelism of the design, specifications, implementation and testing activities, which are indeed carried out at each SCRUM iteration, i.e., at each Sprint. The main contribution of this paper is the identification and the description of eight interfaces, i.e., information exchange channels, between ITIL v.3 and SCRUM, which can be put into action thanks to some described adaptations in the structure of the ITIL v.3 life cycle.

This paper also opens the way for several future works. Applying SCRUM is only possible for the management of *software* implementation project and not for other kinds of projects, such as the installation of hardware. Studying the structure of ITIL v.3 in order to organize both agile and traditional project management is relevant. Indeed, many IT organizations face both software and hardware projects, sometimes mixed.

Another future work is the generalisation of the ITIL adaptations for other agile project management methods. In this context, the integration of this work with the alignment between ITIL v.3 and the service implementation life cycle proposed by the same authors in [33] is an interesting research direction. The objective would be to map the life cycle of the ITSM procedural structure of an IT organization, of the service implementation life cycle in a service-oriented system, and of the agile management of software implementation projects.

A last future work is the execution of a case study in one or several IT organizations working with ITIL v.3 and willing to conduct their software implementation projects with SCRUM. Based on this work, the theoretical propositions made in this paper would be improved thanks to the comments provided and the observations made. Note that this could be a good opportunity to conduct a validation of the SCRUM method seeing that, taken as a whole, the agile research lacks of empirical validation [34,35].

References

1. Hunnebeck, L., Rudd, C., Lacy, S., Hanna, A.: ITIL V3.0 - Service Design, 2nd edn. TSO (The Stationery Office), London (2011)
2. Beck, K., Beedle, M., Bennekum, A.V., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al.: Manifesto for agile software development (2001). http://www.agilemanifesto.org/
3. Conboy, K.: Agility from first principles: reconstructing the concept of agility in information systems development. Inf. Syst. Res. **20**(3), 329–354 (2009)

4. Beck, K., Andres, C.: Extreme Programming Explained: Embrace Change, 2nd edn. Addison-Wesley, Boston (2004)
5. Schwaber, K., Sutherland, J.: The SCRUM Guide. http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf
6. Palmer, S., Felsing, J.: A Practical Guide to Feature-Driven Development. Prentice Hall, Upper Saddle River (2001)
7. Rubin, K.S.: Essential Scrum: A Practical Guide to the Most Popular Agile Process. The Addison-Wesley Signature Series. Addison-Wesley Professional, Boston (2012)
8. Addy, R.: Effective IT Service Management: To ITIL and Beyond. Springer, Heidelberg (2007)
9. Cannon, D., Wheeldon, D., Lacy, S., Hanna, A.: ITIL V3.0 - Service Strategy, 2nd edn. TSO (The Stationery Office), London (2011)
10. Hochstein, A., Zarnekow, R., Brenner, W.: ITIL as common practice reference model for it service management: formal assessment and implications for practice. In: IEEE International Conference on e-Technology, e-Commerce, and e-Services (EEE 2005), pp. 704–710. IEEE Computer Society (2005)
11. Marrone, M., Kolbe, L.: ITIL and the creation of benefits: an empirical study on benefits, challenges and processes. In: Proceedings of the 18th European Conference on Information Systems (ECIS), P. 66 (2010)
12. Rance, S., Rudd, C., Lacy, S., Hanna, A.: ITIL V3.0 - Service Transition. TSO (The Stationery Office), London (2011)
13. Steinberg, R., Rudd, C., Lacy, S., Hanna, A.: ITIL V3.0 - Service Operation, 2nd edn. TSO (The Stationery Office), London (2011)
14. Lloyd, V., Wheeldon, D., Lacy, S., Hanna, A.: ITIL V3.0 - Continual Service Improvement, 2nd edn. TSO (The Stationery Office), London (2011)
15. PMI Standards Committee: A Guide to the Project Management Body of Knowledge (PMBOK®R GUIDE). 5th edn. Project Management Institute (2013)
16. Jurison, J.: Software project management: the manager's view. Commun. AIS **2**(3), 2–57 (1999)
17. Royce, W.: Software Project Management: A Unified Framework. Addison-Wesley Professional, Reading (1998)
18. Martin, J.: Rapid Application Development. Macmillan Publishing Company, New York (1991)
19. Benington, H.D.: Production of large computer programs. Ann. Hist. Comput. **5**(4), 350–361 (1983)
20. Royce, W.W.: Managing the development of large software systems: concepts and techniques. In: Proceedings of the 9th International Conference on Software Engineering (ICSE), pp. 328–339. ACM Press, New York (1987)
21. Cohn, M.: Succeeding with Agile: Software Development Using Scrum. The Addison-Wesley Signature Series. Addison-Wesley Professional, Boston (2010)
22. Pichler, R.: Agile Product Management with Scrum: Creating Products that Customers Love, 2nd edn. Addison-Wesley Professional, Boston (2010)
23. Schwaber, K.: Scrum development process. In: Sutherland, J., Casanave, C., Miller, J., Patel, P., Hollowell, G. (eds.) Business Object Design and Implementation, pp. 117–134. Springer, London (1997)
24. Schwaber, K., Beedle, M.: Agile Software Development with Scrum, 1st edn. Prentice Hall PTR, Upper Saddle River (2001)

25. Göbel, H., Cronholm, S., Seigerroth, U.: Towards an agile method for itsm self-assessment: a design science research approach. In: Proceedings of the International Conference on Management, Leadership and Governance (ICMLG 2013), pp. 135–142. Academic Conferences and Publishing International (ACPI), Sonning Common (2013)
26. Komarek, A., Sobeslav, V., Pavlik, J.: Enterprise ICT transformation to agile environment. In: Núñez, M., Nguyen, N.T., Camacho, D., Trawiński, B. (eds.) Computational Collective Intelligence. LNCS, pp. 326–335. Springer, Switzerland (2015)
27. Office of Government Commerce (OGC): Agile project and service management: delivering IT services using ITIL, PRINCE2 and DSDM Atern. The Stationery Office, London (2010)
28. Pollard, C.E., Gupta, D., Satzinger, J.W.: Integrating SDLC and ITSM to 'servitize' systems development. In Nickerson, R.C., Sharda, R., eds.: Proceedings of the 15th Americas Conference on Information Systems (AMCIS 2009), Association for Information Systems, pp. 3306–3314 (2009)
29. Chen, H.: Towards service engineering: Service orientation and business-it alignment. In: Proceedings of the 41st Hawaii International International Conference on Systems Science (HICSS-41 2008), p. 114. IEEE Computer Society (2008)
30. Shalaby, M., El-Kassas, S.: Applying scrum framework in the IT service support domain. J. Convergence **3**(1), 21–28 (2012)
31. Hacker, W.: Intersection of software methodologies and ITIL v3. In: Proceedings of the IASTED International Conference on Software Engineering, pp. 232–236. ACTA Press (2008)
32. Hoover, C.: ITIL vs. agile programming: Is the agile programming discipline compatible with the ITIL framework?. In: Proceedings of the 32nd International Computer Measurement Group Conference, Computer Measurement Group, pp. 613–620 (2006)
33. Verlaine, B., Jureta, I., Faulkner, S.: Towards the alignment of a detailed service-oriented design and development methodology with ITIL v.3. In: Nóvoa, H., Drăgoicea, M. (eds.) IESS 2015. LNBIP, vol. 201, pp. 123–138. Springer, Heidelberg (2015)
34. Dingsøyr, T., Nerur, S.P., Balijepally, V., Moe, N.B.: A decade of agile methodologies: towards explaining agile software development. J. Syst. Softw. **85**(6), 1213–1221 (2012)
35. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: a systematic review. Inf. Softw. Technol. **50**(9–10), 833–859 (2008)