

# Hybrid Approach to Business Software Development in Terms of Current Needs and Requirements

Aneta Poniszewska-Marañda<sup>a</sup> and Martyna Tyran

*Institute of Information Technology, Lodz University of Technology, Lodz, Poland*

**Keywords:** Business Software Development Methods, Software Methodologies, Hybrid Approach, PRINCE2, Scrum.

**Abstract:** Business traditional software development methods rely on a work plan to extract complete sets of documentation, architectural, and high-level project development and control requirements. Agile has gained immense popularity in the field of computer science in recent years, although it combines acceptable software engineering practices with the controversial ones. The software industry has finally come to the conclusion that specific project characteristics such as purpose, scope, requirements, resources, architecture and size determine the methodology that best fits the project realities. The paper presents hybrid approaches, combining the PRINCE2 methodology and the Scrum method, in order to take maximum advantage of the methodologies and minimize their disadvantages by using the strengths of one methodology on the weaknesses of the other.

## 1 INTRODUCTION


The software development methodology appeared only in the 1960s. The main idea of the first method was to strive to develop information systems in a very purposeful, organized and methodical way, requiring each stage of the life cycle – from the emergence of an idea to the delivery of the final system – to be carried out rigidly and sequentially in accordance with the framework used. The main goal of these methodologies in the 1960s was to develop large-scale functional business systems in the times of large business concerns. Information systems activities focused on heavy data processing procedures and complex numerical calculations. Since then, software engineering has been developing very dynamically, and along with it – methodologies that had to keep up with the requirements and expectations that were set before them.

Traditional software development methods rely on a work plan to extract complete sets of documentation, architectural, and high-level project development and control requirements. Due to these serious aspects, this method has become known as severe. Over time, practitioners of these methodologies realized that such a software development process was frustrating in their daily work. Consequently, it has developed methods and practices to absorb and re-

spond to the inevitable changes.

Over time, agile methods gained immense popularity in the field of computer science, even though they combined acceptable software engineering practices with controversial ones. Eventually, it became clear that the specific characteristics of a project, such as the purpose, scope, requirements, resources, architecture and size, do indeed determine the methodology that best fits the realities of the project. Thus, dynamic, heavy, and possibly hybrid methodologies can be used. In recent years, when the uncritical admiration for agile methodologies is slowly fading away and their results are beginning to become visible, voices are heard more and more often that there is a lack of order and confidence that traditional methodologies provided. Hybrid approaches begin to combine the advantages of both types of methodologies – similarly to the solution proposed in this paper, to use the maximum advantages of the methodologies and minimize the disadvantages of these approaches by applying the strengths of one methodology to the weaknesses of the other (Fowler, 2020).

The paper presents proposals for a modern approach to software development in a hybrid system. This proposal was based on the analysis of methodologies in the traditional and agile approach, which included considering their specific implementations, analysing the research of these approaches by reputable organizations such as Standish Group (infoQ, 2020) and conducting literature research among sci-

<sup>a</sup>  <https://orcid.org/0000-0001-7596-0813>

entific works in the field of project management. The proposed approach was based in particular on the PRINCE2 method and the Scrum method. In addition, the advantages and limitations of both methodologies were presented and the risk assessment of the proposed hybrid method was made.

The paper is structured as follows: section 2 presents the overview of traditional and agile methods together with the limitations of both of them. Section 3 describes the proposal of hybrid software development based on PRINCE2 and scrum approaches.

## 2 TRADITIONAL AND AGILE METHODOLOGIES

Traditional methodologies are an approach that is considered to be a classic method of software development. They are based on a sequential series of steps such as defining requirements, building a solution, testing and installing. These methodologies require a set of requirements to be defined and documented at the very beginning of the project. They impose a disciplined process for software development to be more predictable and efficient (Fowler, 2020).

These methodologies are preferred for projects that result in a physical object (such as structural designs, equipment installation, and the like) and for projects where tasks must be completed in a specific order. Moreover, the design plans can be used in the future for other projects. On the other hand, heavy methodologies require much more planning work. It is estimated that approx. 20-40% of the entire project time is invested in the first two phases (collecting requirements and design). Due to the highly structured approach, any changes are very slow, which makes it inappropriate for projects where the client is not entirely sure what it really requires. One of the most popular implementations of heavy methodologies are: Rational Unified Process (RUP), PRINCE2, Project Management Body of Knowledge/Project Management Institute (PMBOK/PMI).

Agile methodologies, as opposed to the traditional approach, focus on adding lightness to all their processes, leading to high-quality software and customer satisfaction. Above all, a linear sequential approach is avoided in favor of an incremental iterative approach. Instead of extensive planning and designing right from the start, agile allows requirements to change over time by using multi-functional teams. They include people responsible for planning, design, developers and testers who work on subsequent iterations of the project for a specific period of time (time-boxes). The work is organized through the Backlog

(a list of tasks to be done), which is ordered by the priority of the business value (or user) of a given task (Alliance, 2020). The best known agile methods are: Extreme Programming (XP), Scrum, Kanban, Feature Driven Development (FDD), Dynamic System Development (DSD), Adaptive Software Development (ASD), Lean Development.

Advantages of traditional methodologies include:

- Potential problems that would only be found during the development of the system can be found, investigated and listed during the design phase.
- The software development process tends to be better documented as this methodology places more emphasis on documentation such as requirements and project documents. Many organizations find this reassuring.
- Because the traditional process is linear, it can be easier to understand, especially for non-developers or those who are new to software development. They often feel more comfortable with this approach.

However, these methodologies also have several significant drawbacks:

- Very often, the people for whom software is being built do not know what exactly they will need at the very beginning and they do not know what is possible with the technology available.
- Solution designers are often unable to foresee problems that will arise during the implementation of their plans.
- Requirements changes are not easy to incorporate into traditional methods as they must go through heavy change control procedures if this is the case.
- The process does not have a tempo of its own.

In the case of agile methods, which were created as a response to traditional methodologies, they have many important advantages:

- Working software is delivered much faster over successive iterations, and can be delivered frequently and at a consistent tempo.
- There is very strong collaboration between developers and business.
- Changes in the requirements can be incorporated into the project at any time during the work – even in the final stage of development.
- They are highly transparent.

Unfortunately, however, like any solution, agile methodologies also have undoubted drawbacks:

- Agile implementations such as Scrum are often more difficult to understand than linear, sequential approaches – at least initially.

- Due to the emphasis on running software, documentation can sometimes become neglected. The focus should be on documentation appropriate for a given recipient, but if the chosen methodology is not properly implemented, this aspect may be neglected.
- When poorly implemented in an organization, agility can be a source of ineffectiveness or it can act against long-lasting organizational processes.

## 2.1 Limitations of Traditional Methods

The main difference between heavy and light methodologies is change acceptance. It means being able to respond to changes that could make a project successful or unsuccessful (Barry, 2016). Heavy methodologies freeze product functions and do not allow for changes. However, the key to making agile methodologies successful in the current market is responding quickly to changes at every stage of the project. This makes it difficult to implement a predictable process or ensure a set of persistent requirements in such an unstable and ever-changing environment. Michael Dell, Dell's founder and CEO, concluded that the only constant is change (Academia, 2020). Martin Flower and Jim Highsmith, the authors of the Agile Manifesto, found that allowing and facilitating change is much more effective than trying to stop it. Learning to trust your ability to respond to unforeseen events is more important and valuable than trusting your disaster planning ability (Barry, 2016). Moreover, both Barry Boehm and Capers Jones, two prominent software engineers, concluded that over the course of their project development experience, requirements changed by at least 25% (Barry, 2016; Jones, 2008).

The Standish group has researched 50,000 projects from all over the world, ranging from small extensions to large implementations of entire systems. The results of their research show that in 2015, 29% of projects were completed on time, within the agreed budget, with all designated functions. 52% of the projects were completed, but exceeded the agreed timeframe, budget and offered fewer functions than planned. In contrast, 19% of the projects were not completed at all because they were closed at some point in the development cycle. The trend of such distribution has remained almost unchanged for 4 years – the ratio of project statuses is almost the same (infoQ, 2020).

In the case of a more precise distribution with the division into methodologies used in the project, a rule can be observed that regardless of the size of the project, heavy methodologies perform much worse than agile methodologies. Although in the case of

small projects Waterfall is doing well, as as much as 44% of projects are successful, in the case of agile it is still 58%. The study reveals that the four most important factors that ensure project success are: executive management support, emotional maturity, user engagement, and optimization.

Another limitation of heavy methodologies is dealing with complexity. Former Visa International CEO Dee Hock stated that complex rules and regulations lead to simple dumb behaviour (Wlodarski et al., 2022). An approach where you plan everything first and then follow a plan works great for stable and less complex environments, but for larger and more complex environments this technique fails. The solution to this problem lies in simplicity. Dee Hock added that simple, clear purpose and rules result in complex, intelligent behaviour. Some companies follow simple rules to survive in complex and turbulent markets. Jack Welch, CEO of General Electric (GE), transformed his company to a market value of nearly 500 billion (from the original 12 billion). Agile methodologists promote exactly the same approach. Fowler and Highsmith confirm that in an agile project it is especially important to use the straightforward approach as it is easier to change. It is much easier to add something to a process that is too simple than to remove something from a process that is too complicated (Deemer et al., 2012).

Another study by the Standish group shows that almost 50% of the functions present in applications are almost always unused. This is another reason to make models and code as simple as possible, as half of the software produced and the complexity added to it is not needed. Another feature that limits traditional methodologies is sequential production. These methods postpone customer feedback and testing to the last stage of the project's life. People practising agile methodologies believe the opposite – it should be incorporated into everyday work. Fowler believes that the key to iterative software development is frequent production of working versions of the final system that contain a subset of the required functionality.

## 2.2 Limitations of Agile Methods

The biggest limitation of lightweight methodologies is the way they deal with large teams. Cockburn and Highsmith find that agile development is all the more difficult for larger groups as the communication interface grows in size and becomes a dominant problem (Highsmith and Cockburn, 2015). Martin Fowler also believes that agile face-to-face communication breaks down and becomes much more difficult and complex in teams of more than 20 people (Fowler, 2020; Con-

stantine, 2001). In contrast, heavy and scheduled methodologies scale much better to larger projects.

Barry Boehm disagrees to some extent with the Agility Manifesto's first rule, which states that customer satisfaction from the early stages and continuous product delivery is a top priority (Cunningham, 2020). Boehm believes that over-focusing on early results in large systems can later lead to a whole project revision (or even the need to start over) as the architecture and subsequent requirements are increased (infoQ, 2020). Boehm also maintains that a plan-driven process is most needed for highly reliable software development. Even the founder of agile modelling, Scott Amber, states that he would be very restrained when using agility in life-critical systems (Boehm, 2002). Traditional heavy methodology goals of predictability, repeatability, and optimization are often hallmarks of reliable and secure software.

It has been proven that such methods are insufficiently suited to the stringent regulation of critical systems. Alistair Cockburn questioned agile methodologies by asking how much agility there could be in a project, given that the software produced must be critical, reliable and secure (Fowler, 2020). In turn, Martin Fowler mentions that agile methodologies provide working solutions only for business.

The debate as to whether agile methodologies require creative and capable people to be effective leads to another argument. Using only the best people who can do almost anything makes using that particular type of methodology unimportant. This means that perhaps agile methodologies may owe their success to teams of very good people rather than to practices and principles. Alistair Cockburn agrees, but at the same time explains that if the people in the project are skilled enough, they can use any process to complete their assigned tasks. If they are relatively weak, no process will help and fix their inadequate qualifications. In other words, people take precedence over processes (Highsmith and Cockburn, 2015).

Another important thing that can cause problems is the interpretation of the dogma of the manifesto – working software over extensive documentation (Cunningham, 2020). Boehm questions the usefulness of the emphasis in agile techniques for simplicity. The YAGINI (*You ain't gonna need it*) concept based on extreme programming, one of the agile methodologies, believes in doing the extra work of removing architectural features not supported in the current version of the software to achieve simplicity. This approach can be useful when subsequent requirements are highly unpredictable (Boehm, 2002).

Project and product documentation is a topic that has attracted a lot of attention in Agile methodolo-

gies. On the one hand, agile advocates believe that organizations require more documentation than necessary, and that documentation is a poor form of communication (MacCormack, 2016). However, documentation is needed to retain critical information over time. Barry Boehm mentions that documented design makes it easier for external experts to diagnose the problem. Proposing no documentation on the project increases the risk when further maintenance, support and other aspects of use are considered, as agile methodologies assume that the development team will be together all the time until development is complete, which is unlikely in many cases (Boehm, 2002).

### 3 PROPOSAL OF HYBRID SOFTWARE DEVELOPMENT

The Scrum process is about delivering. Fast and efficient product manufacturing is the key. In PRINCE2, the delivery process is a black box that leaves a lot of freedom in the way you create. On the other hand, the methodology is very rich in guidelines for managing the project process. This makes Scrum naturally fit into the product development management process in PRINCE2.

#### 3.1 Seven Principles of PRINCE2 and Scrum Value

First of all, integration begins with Scrum's compliance with the basic 7 principles of PRINCE2, as shown in table 1. Scrum's ideology fits perfectly into the seven themes that form the basis of PRINCE2. This means that creating a hybrid of these two methodologies is possible while maintaining their advantages.

#### 3.2 Seven Topics of PRINCE2 and Scrum Philosophy

Below is shown how to modify the seven themes that affect project management. Important elements from the Scrum ideology have been added to achieve a better result.

**Organization.** PRINCE2 introduces the organizational structure for projects. The chairman is the key person making decisions about the project and the supreme supervisor. The prime user represents the end user and the prime vendor represents the project vendors. These three form a body called the steering committee. In Scrum, the main roles are the development team, the Scrum Master, and the product owner.

Table 1: Scrum integration to the seven PRINCE2 principles.

PRINCE2 principle	Scrum integration
Further <b>business case</b> for the project must exist throughout the project life-cycle.	Scrum encourages the Product Owner to review the <b>Product Backlog</b> throughout the process to ensure that the requirements are prioritized according to the business case, and updates the list based on it.
<b>Learning from experience</b> – lessons from previous projects are documented and knowledge is passed on.	Scrum recommends a <b>Sprint Retrospective</b> at the end of each Sprint, during which the team asks themselves – what could we have done better? This knowledge is saved for the next Sprint or Project. Adaptive cycles based on prior learning are also recommended.
<b>Clearly defined roles and responsibilities</b> – chairman, main user, lead vendor, project manager, project support etc.	Scrum has its <b>own set of roles and responsibilities</b> , ie Scrum Master, Product Owner and Development Team. Although the two methodologies have different roles, there are <b>no obstacles to combine one role with the other</b> . Although some roles will be removed, such as the team leader role, his responsibilities will be split into others.
Managed by <b>phases</b> – each project must have at least 2 management phases (checkpoints).	Scrum delivers the product in <b>iterations</b> , called sprints, which can be considered similar to stages.
<b>It focuses on products and their quality</b> . Documents are created where the design and product descriptions contain detailed acceptance criteria and quality expectations for each delivery.	Scrum ensures that the team reviews the Product Backlog and converts it into the Sprint Backlog. It contains a list of tasks to be completed, however, before this happens, the team must consult the Product Owner on the user stories and <b>acceptance criteria</b> used to deliver the product through quality checks during the Sprint review.
Tailored to <b>suit</b> your design environment	Scrum can be used in the PRINCE2 methodology as long as its <b>7 principles are not violated</b> .

Project manager and team leader roles have been removed.

All of the above roles have different responsibilities, and they all have an excellent justification for why they are necessary in the project. How best to combine them depends on the skills and personality of the individuals throughout the development team:

- The role of **project manager** can be fit to the role of **Scrum Master**, but only if the person understands this role as supporting and strengthening the team, and not as managing their style of work. Therefore, the best solution is for the Scrum Master role to be taken by a person who has no knack for traditional management – common solution is to naturally select a Scrum Master from the team or hire a dedicated person.
- The **project manager** takes a supportive role, delegating his responsibilities related to delivering the product to Scrum Master, who focuses on making the team understand Sprint goal and try to achieve it, removing all obstacles reported by this team along the way.
- The role of **Product Owner** is ideally combined with the role of main user, as they know the product best. It is also possible for the project manager to assist the main user – the Product Owner, in the early phases of the project to create a complete product description, or in this case, the Product Backlog.
- The role of Product Owner can be taken over by the entire **steering committee**. This adds positive value to the overall process, as Scrum is not very good at being the vendor taking the commercial risk. because it is in no way represented

in the trial. Additionally, the steering committee is responsible for the success of the project, not the project manager, scrum master or the development team. He becomes a huge ally of the above roles, e.g. in removing difficult obstacles in the project.

**Business Use.** This topic answers the question of whether the project is worthwhile, feasible, desirable and achievable. The **business case** must be valid for the entire duration of the project. Detailed analyses are made of what benefits the project will bring, the reasons for undertaking it, as well as cost estimates, business options, time frame and major risks. The rationale may evolve over the course of the project. In the hybrid approach, it is recommended to address agility and start with a high-level business case, then revise the case after each Sprint.

Additionally, a project can be started with a single, high-level definition of the project, e.g. in the form of a **vision statement** – a project summary that shows the change achieved through the project, which forms the basis for communication with stakeholders. This is to ensure a common understanding of the goals of the program, motivate clients to actively engage, and facilitate focusing activities on achieving the desired change (MacCormack, 2016).

In Scrum, the **business case** is determined by the Product Owner. He presents a prioritized list of requirements to the development team. Then, they work together to create a Backlog of the upcoming Sprint, which includes user stories and acceptance criteria. It is recommended to use the technique known from PRINCE2, called Product breakdown structure. It involves creating a hierarchy of all products to be pro-

duced under the plan. The aim is to understand the composition and function of all products to be produced (MacCormack, 2016). The top-down diagram shows the breakdown of all products. External products must also be included but clearly distinguished. Thanks to the use of such a technique, it is possible to introduce order in the Backlog, which can have a rather chaotic structure. The PRINCE2 business case is a high-level document in the organization and should contain the original Product Backlog produced by the Product Owner.

**Quality.** The purpose of this topic is to ensure that the product that the project is delivering is fit for the purpose for which it is being created. PRINCE2 puts a lot of emphasis on quality, just like Scrum. In PRINCE2, the project manager creates a Quality Management Strategy that defines what quality standards will be applied to the project. It clarifies which controls in the process itself will be performed (for example, internal testing) and beyond using Quality review techniques (MacCormack, 2016). Scrum details the use of automated tests as an internal control process during each Sprint and Sprint Review, which follows a very similar scheme to the Quality Review Technique. Incorporating these practices into the quality process shouldn't be a problem as both Scrum and PRINCE2 are very similar in their approach to quality.

**Plans.** Both PRINCE2 and Scrum have Stage / Sprint plans and both use product-based planning techniques. The management style in PRINCE2 is management by exception. At the beginning of the project, set limits (tolerance) are set, in the centre of which the team can move freely. The steering committee is only involved when there is a risk that the project will go out of bounds. This works well with an agile approach as well. The cost and time tolerance may be predetermined (zero tolerance), while the scope of work tolerance may be the minimum scope that must be delivered, for example all user stories with the necessary business value.

However, if there is a need to change the basic requirements, the project manager still has to submit an extraordinary report to the chairman and the steering committee. This is also the case if the project takes longer than expected or the established tolerances are compromised. The steering committee together with the product owner (or the steering committee as Product Owner) can give a positive opinion on the report or reject it and ask the project manager to create a new emergency report or a new baseline plan. This plan can be used to update the product backlog.

Scrum Sprints follow one another, so there is no time to get formal approval from the steering committee to start the next Sprint. On the other hand, ac-

ording to PRINCE2, approval by the committee is mandatory until the next stage begins. The nonconformity at this point can be resolved by consenting to the implied acceptance. This means that Sprint Planning meetings are an informal way of obtaining approval. The Sprint Backlog created during this meeting automatically becomes the Stage Plan. During the first few days of the next Sprint, the project manager delivers a stage report to the steering committee and product owner describing the results of the previous Sprint and the consequences of the planned delivery. Moreover, this is where predetermined tolerances help – when the results of a Sprint or scheduled delivery are within the project boundaries, consent can be obtained automatically.

**Risks.** In PRINCE2, the project manager creates a risk management strategy and opens a risk register. This means establishing, identifying, estimating, planning, implementing and communicating risk together with detailing risk-related activities. The risk register should contain the risks and detailed prevention plans.

Scrum does not provide any specific risk management methods. However, this is largely not a problem as most risks are managed by default by delivering the product in short time intervals. It is a very effective approach to dealing with obstacles within the sphere of influence of a project. The problem is the risk that lies outside the team. The Scrum Master is responsible for managing such problems, but the Scrum Master himself has limited authority. It is also not a proactive approach to problems.

The PRINCE2 approach has clearly defined risk management and active involvement of the steering committee. It shouldn't take too much time and effort, and it can certainly add a lot of value to the process. The Scrum Master can deal with risks/issues within the team process and the steering committee on other issues. The project manager can support both the Scrum Master and the steering committee in this and act as an intermediary between them.

Additionally, if the risk materializes during the project, the countermeasure plan should also be embedded in the Product Backlog so that the team can commit to it in the next sprint. It will be given the appropriate priority by the Product Owner.

**Change.** Both methodologies recognize change if it is inevitable, but deal with it in completely different ways. PRINCE2 can approve a phase scope, duration and budget change, while in Scrum the Sprint duration is inviolable and the change can only be scope or budget. PRINCE2 approach to change is more formal, involving more stakeholders, and may take more time and work. Agreeing to add and remove

items may require changes to the budget and other important project parameters. Therefore, in this case, the Project Manager together with the Product Owner must document the change in the form of an Emergency Report and then, when approved by the steering committee, implemented by the team in the upcoming sprint (if there is a high priority for the change). When the product is owned by the entire steering committee, the steering committee should have the final opinion on changing the requirements.

The topics in PRINCE2 cover all the problems related to product development. If the tolerance is exceeded, an exception is thrown. In this case, the current step is solved and an impact analysis is performed to determine if a new plan is needed. In Scrum, the issues that influence product development are called obstacles. When such an obstacle threatens the completion of a Sprint and cannot be removed, the Sprint is interrupted. In both methodologies, exceptions are used to return control to the managing authority so that appropriate decisions can be made and corrective action taken.

The project manager should be able to manage the PRINCE2 issue log and thus capture and deal with change. The Scrum Master can also maintain this document by entering into it all obstacles in the project, which include problems, concerns or errors. This allows the project manager and the Scrum Master to work together to remove obstacles and deal with different issues.

**Process.** The team in Scrum does not report to the project manager, the Product Owner or the Scrum Master. This eliminates the checkpoint report known from PRINCE2. However, the same information can be read from scrum charts – Sprint burn, product burn, and speed. The project manager can use this information to create an interim report to the steering committee. Additionally, he or she can use other tools provided by scrum – Product Backlog, Sprint, Retrospection or Planning. It also allows the project manager to follow the progress of the work.

PRINCE2 experience report and logs are very similar to a Sprint flashback and capture the same information. Therefore, all valuable insights and experiences should be recorded by the Scrum Master and passed on to the project manager so that he can manage the above documents. Additionally, the advantage of Scrum is that there are flashbacks after each sprint, which makes there a lot of opportunities to try out and learn from different solutions as the process and people stay the same in the project all the time. It is also worth making this process visible to the client by reporting the results of the retrospective in each report at the end of the Sprint.

After presenting how the seven principles and topics of PRINCE2 have been modified, view at how the seven processes in Scrum hybridization will change are given in figure 1.

## 4 CONCLUSIONS

The paper presented a proposal to create a hybrid methodology that combines the advantages of traditional and agile methodologies. A specific method of incorporating the implementation of the agile methodology – Scrum, into the implementation of the traditional methodology – PRINCE2, was presented, bearing in mind the inviolability of the most important pillars of these methods.

After analysing all aspects of Scrum and PRINCE2, it seems that Scrum perfectly complements PRINCE2. The software development team works in Scrum, while the management team works mainly in PRINCE2. Both methods are compatible with each other. The real and most visible benefit of adding a PRINCE2 layer over Scrum is to help many organizations move steadily towards agile methodologies and a more successful project outcome.

When using the PRINCE2 and Scrum hybrid solution or other agile project management approaches, pay attention to the following "traditional management behaviours":

- A so-called contract is created through a specification known from traditional methodologies, and then delivered through the project manager who becomes the contract negotiator between the customer and the supplier. An agile team consists of customers (product owner) and suppliers (development team) working together to achieve clear goals for Sprint and product delivery.
- The project manager does not lead and does not control. Instead, it facilitates and enables the effective work of the development team and protects it from outside influences that are beyond the team's domain.
- The steering committee does not command and control. Instead, it facilitates and enables the effective work of the project manager and protects him from outside influences that are outside the manager's domain.
- Estimates are estimated forecasts based on the best information available (such as baselines); forcing delivery against a pre-agreed contract will result in significant errors if the delivery environment changes.

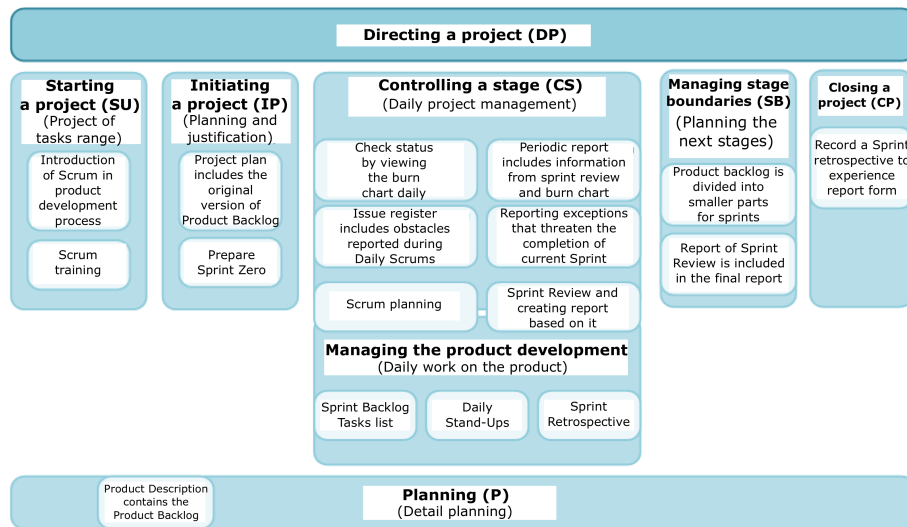


Figure 1: PRINCE2 processes integrating the Scrum method.

- Delivering products should be done by continuously adding new pieces to a working software (or product), delivered over a period of weeks or months, not through longer phases.

Values and principles outlined in agile manifest should also be enabled and consistently encourages:

- PRINCE2 and Scrum overlap in some areas. However, agility is usually compared to and contrasted with traditional approach, and PRINCE2 is itself a project management framework.
- Scrum includes the Product Backlog; typically user and defect stories defining what to do and this is created and maintained by the Product Owner. The project team then defines tasks that define how each User Stories will be performed to meet the stories within the short Sprint time span, which typically takes two to four weeks.
- Development team always has a goal of delivering a work product at the end of each Sprint as well as focusing on quality that can be implemented in a production environment. However, if it is not possible to deliver a product from a given Sprint to the production environment, the Sprint steps can be grouped together and delivered at a later release.
- PRINCE2 can include any approach to product delivery, so Scrum is no exception. Scrum provides the development team with discipline and strictness in delivering the product. On the other hand, PRINCE2 gives you a complete project management framework. Therefore, Scrum and PRINCE2 complement each other perfectly.

PRINCE2 and SCRUM can be integrated by creating a hybrid approach. While there are several chal-

lenges in this regard, there are also some benefits. Areas where the two methodologies complement each other, such as management, documentation, structure, can reduce risk and increase success rate of project.

## REFERENCES

Academia (2020). *Agile Software Methodologies Strength and Weakness*. Online, <http://www.academia.edu/18693780/>.

Alliance, A. (2020). *What is Agile Software Development?* Online, <http://www.agilealliance.org/the-alliance/>.

Barry, B. (2016). *Understanding and controlling software costs*.

Boehm, B. (2002). *Get ready for agile methods, with care*.

Constantine, L. (2001). *Methodological agility*.

Cunningham, W. (2020). *Agile Manifesto*. Online, <http://www.agilemanifesto.org/>.

Deemer, P., Benefield, G., Larman, C., and Vodde, B. (2012). *The Scrum Primer: A Lightweight Guide to the Theory and Practice of Scrum*. Dymaxicon.

Fowler, M. (2020). *The New Methodology*. Online, <https://www.martinfowler.com/articles/>.

Highsmith, J. and Cockburn, A. (2015). *Agile Software Development: The People Factor*. Addison-Wesley.

infoQ (2020). *Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch*. Online, <https://www.infoq.com/articles/>.

Jones, C. (2008). *Applied Software Measurements*. McGraw-Hill Education.

MacCormack, A. (2016). *Product-development practices that work: How internet*.

Wlodarski, R., Poniszewska-Maranda, A., and Falleri, J.-R. (2022). *Impact of software development processes on the outcomes of student computing projects: a tale of two universities*.