

Agile Software Engineering Practices and ERP: Is a sprint too fast for ERP Implementation?

Adnan Kraljić, Tarik Kraljić,

Ghent University, Ghent, Belgium
adnan.kraljic@ugent.be, tarik.kraljic@ugent.be

Abstract. The Enterprise Resource Planning (ERP) implementation is a complex and active process, one that involves a mixture of technological and organizational interactions. Often it is the largest IT project that an organization has ever launched and requires a mutual fit of system and organization. Concept of an ERP implementation supporting business processes across different departments in organization is not a generic, rigid and uniform process - it is a vivid one and depends on number of different factors. As a result, the issues addressing the ERP implementation process have been one of the major concerns in industry. Therefore, ERP implementation process receives profound attention from practitioners and scholars in its academic or industry papers. However, research on ERP systems so far has been mainly focused on diffusion, use and impact issues. Less attention has been given to the methods/methodologies used during the configuration and the implementation of ERP systems; even though they are commonly used in practice, they still remain largely unexplored and undocumented in Information Systems research domain. Furthermore, research on Agile Software Engineering Practices in ERP Implementations context is almost nonexistent. Many IT specialists find agile management frameworks positive, but ask themselves whether these can also cope with the complex adjustments of ERP systems. The answer is quite simple: agile management frameworks, in particular Scrum, were developed for the exact purpose of enabling successful execution of large and complex projects. Depending on the size of the company, an ERP project including preparation can take over a year – or it can become a long-runner that eats up resources. One reason for this: if projects are handled in line with traditional procedural models, lengthy integration and acceptance tests are not carried out until the end. So, at the end of the development the detected errors and change requests pile up and delivery is delayed. This study is a response to the frequent calls for industrial case studies on agile software development (Dingsøy et al. 2012; Dybå and Dingsøy 2008). This paper is useful to researchers who are interested in ERP implementation methodologies and frameworks. Paper also aims at the professional ERP community involved in the process of ERP implementation by promoting a better understanding of ERP implementation methodologies and frameworks, its variety and future development.

Keywords: ERP, ERP implementation, Agile implementation methodology, sprints, phases etc.

1 Introduction

Implementing an ERP system is a major project demanding a significant level of resources, commitment and adjustments throughout the organization. Often the ERP implementation project is the single biggest project that an organization has ever launched [1]. As a result, the issues surrounding the implementation process have been one of the major concerns in industry. And it further worsens because of numerous failed cases including a few fatal disasters which lead to the demise of some companies. In previous studies can be found that almost 70% of ERP implementations fail to achieve their estimated benefits [2]. Although ERP can provide many benefits for organization, goals are often changed to getting the system operational instead of realizing the goals [3]. Reflecting such a level of importance, the largest number of articles in literature belongs to this theme. It comprises more than 40% of the entire articles [4]. Many of these articles share implementation experiences from various companies. Also, various models of implementation stages and different implementation methodologies are presented and will be discussed more in next section.

1.1 ERP Implementation Methodologies in Literature

Research on ERP systems has so far been mainly focused on implementation CRF/CSF and impact issues [5]. Less attention has been given to the methods used during the configuration and the implementation of ERP systems [6], even though they are commonly used in practice [7], they remain unexplored in ISD research. Several models of ERP implementation methodologies are provided in literature and they vary according to e.g. the number of phases. The phases in ERP implementation frameworks are often counted as between three and six, according to Somers and Nelson [8]. However, the model of [9] includes 11 phases and it gives practical checklist-type guidance for an ERP implementation. On the other hand, the models of Markus and Tanis or Parr and Shanks are very general, and are merely used for analyzing ERP implementation projects. The models are useful in studying, analyzing and planning ERP implementation. [10]

The selection of ERP implementation method mentioned in paper is based on the degree of “institutionalization” in the scientific community. Livari and Hirschheim described six criteria to determine institutionalization: including 1) the existence of scientific journals, 2) scientific conferences, 3) textbooks, 4) professional associations, 5) informational and formal communication networks, and 6) citations [11].

There are number of different ERP implementation methodologies mentioned and described in literature. However, there is an issue with methodology scope, context and its ambiguity. For example, some methodologies treat the phases before the acquisition of an ERP system (and are focused on it), while some methodologies put stress on phases after the ERP system has started to be used (production phase) [12]. A board concept for an ERP implementation also covers these after and before phases. Different authors provide different sequence of phases and diverse naming practice. The preliminary phases are, for example, initiation and requirements

definition defined by Kuruppuarachchi project chartering by Markus [13] and initiative and selection by Makipaa [14]. Verviville and Halingte even present a Model of the ERP Acquisition Process (MERPAP). The phases after the ERP system is put into use are described as termination, onward and upward), exploitation and development enhancement, acceptance, routinization, and infusion and stabilization, continuous improvement and transformation. In some cases, an ERP implementation concept may cover only phases between the acquisitions and beginning of usage of a system, for example, “go live” phase. For instance, Ross proposed a five-stage model for ERP: implementation, stabilization, continuous improvement and transformation (covering only phases between the acquisition and beginning of usage of a system). Markus & Tanis suggested a model named enterprise system experience cycle, which has four phases: charter, project, shakedown and onward and upward etc. [15].

It is obvious that there is no ground based ERP implementation methodology, widely accepted and tested. Even though they are commonly used in practice (ERP implementation methodologies) they still remain largely unexplored and undocumented in Information Systems research domain. Next table summarize list of proposed implementation methodologies followed by the degree of institutionalization in scientific community [16].

Author(s)	ERP implementation model
Bancroft et al. (1998)	(1) Focus, (2) Creating As – Is picture, (3) Creating of the To-Be design, (4) Construction and testing and (5) Actual Implementation
Kuruppuarachchi et al. (2000)	(1) Initiation, (2) Requirement definition, (3) Acquisition/development, (4) Implementation, and (5) Termination
Markus and Tanis (2000)	(1) Project chartering, (2) The project, (3) Shakedown, and (4) Onward and upward
Makipaa (2003)	(1) Initiative, (2) Evaluation, (3) Selection, (4) Modification, Business process Reengineering, and Conversion of Data, (5) Training, (6) Go – Live, (7) Termination, and (8) Exploitation and Development
Parr and Shanks (2000a)	(1) Planning, (2) Project: a. setup, b. reengineer, c. design, d. configuration and testing, e. installation (3) Enhancement
Ross (1999)	(1) Design, (2) Implementation, (3) Stabilization, (4) Continues improvement and (5) Transformation
Shields (2001)	Rapid implementation model of three phases and 12 major activates
Umble et al (2003)	(1) Review the pre-implementation process to date, (2) Install and test any new hardware, (3) Install the software and perform the computer room pilot, (4) Attend system training, (5) Train on the conference room pilot, (6) Established security and necessary permissions, (7) Ensure that all data bridges are sufficiently robust and the data are

	sufficiently accurate, (8) Document policies and procedures, (9) Bring the entire organization on – line, either in a total cutover or in a phased approach, (10) Celebrate, and (11) Improve continually
Verviell and Halington	(1) Planning, (2) Information search, (3) Selection, (4) Evaluations, and (5) Negotiation

Figure 2.1 illustrates the three generations of software engineering. The diagram on top of the figure schematically depicts the ability of each particular generation to handle software complexity in terms of cost per unit [17]. As the complexity of software development projects has constantly risen over time, new approaches were introduced which better suited the given project contexts.

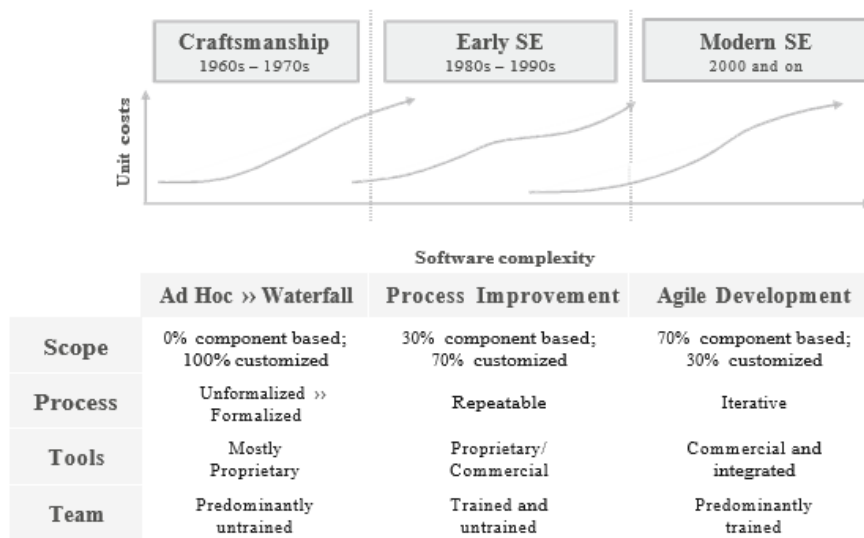


Fig. 1.1 Trends in software engineering (based on Royce et al. 2009)

1.2 Agile Software Development

In February of 2001, 17 prominent software development figures (Person A, Person B, Person C, Person D, Person E, etc. — just to name a few) met in The Lodge at Snowbird ski resort in the Wasatch mountains of Utah to share their ideas on software development methods known as “lightweight ‘methods at the time. [18] The result of the meeting was the Agile Manifesto. These agile values were derived from previous light- weight methods introduced by these agilists in the 1990s and early 2000s. [19] The four values constitute the essence of agile software development:

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

Instead of formalizing the development process with detailed specification of software requirements, agile software development meant a distinct move towards continuous, informal, and close customer collaboration . Unnecessary documentation was avoided as much as possible emphasizing a “lean” mentality adopted from lean manufacturing [20].

Following the publication of the Manifesto, the Agile Alliance was created to promote and evolve agile software development methods. The manifesto was derived from the ideas of many iterative development methods including Extreme Programming (XP), Scrum, DSDM, Crystal, Adaptive Software Development, and several others. It is rather difficult to disagree with the Manifesto once it is fully understood. In fact, most people in the software development space nod their heads when they hear the above statements. These principals resonate in high-quality organizations that have great talent and pride themselves on collaboration, client service, and adaptability to change. The fact remains that without talent, no software development process will ever lead to success. Incorporating agile methodologies into our playbook puts us in good company along industry leaders (Google, IBM, Lockheed Martin, and many others) and positions us to provide our clients with the right type of SDLC transformation based on their culture and business needs. [21]

Agile characteristic	Description	Benefits
Time-boxed iterations	<ul style="list-style-type: none"> • Scope (in the form of user stories) is broken up in small chunks based on priority and/or complexity • Iterations are time boxed in short timeframes (generally one to six weeks) • Each iteration includes the entire SDLC life cycle • The result of each iteration is a working solution with minimal bugs • Multiple iterations may be required before releasing a system to end users • Scope is generally fixed during iterations, but can change in between iteration based on new business needs 	<ul style="list-style-type: none"> • Improved quality due code integration and test within each iteration • Early return on investment as working code is produced early on in the process

Client collaboration	<ul style="list-style-type: none"> • Client lead (product owner) is an active team member who is engaged in iteration planning and review. • Product owner is always aware of progress being made and the direction of system development • Product owner is available to answer the team 's question 	<ul style="list-style-type: none"> • Quick resolution of issues • Adaptive evolution of requirements from high-level user stories to detailed requirements through continuous client discussion • Trust-based relationship with the client
Self-organizing teams	<ul style="list-style-type: none"> • Agile teams are typical assembly of cross-functional team members capable of carrying out all aspects of the SDLC • Team members are empowered to make decisions, estimate effort, decide roles and responsibilities, and are protected from outside influences • All team members are given the opportunity to interact with the client • Collocated teams with emphasis on face-to-face communication. If team members are not collocated, then the use of video conferencing and conference calls to keep the team connected 	<ul style="list-style-type: none"> • Improved productivity • Increased motivation • Clear responsibility and accountability • Staff development and growth • Improved team chemistry
Adaptable to change	<ul style="list-style-type: none"> • Agile methods embrace change as inevitable. Business needs are changing rapidly and agile methods do not assume that requirements can be frozen • Thus, agile methods work closely with the client to adapt the solution to their changing requirements • Scope and requirements are reviewed between iterations, modified, and prioritized for the next iteration • In addition to adaptability to requirements, the design of the system is also adaptable to changing needs. Team members are encouraged to refactor code and improve their quality regularly. The team is encouraged to adapt any aspect of the system as new information becomes available. 	<ul style="list-style-type: none"> • Increased client satisfaction as the system is adapted to their needs • Improved system design that is simple and easy to understand (due to frequent refactoring of code)
Product	<ul style="list-style-type: none"> • Working software is the main objective 	<ul style="list-style-type: none"> • Working code

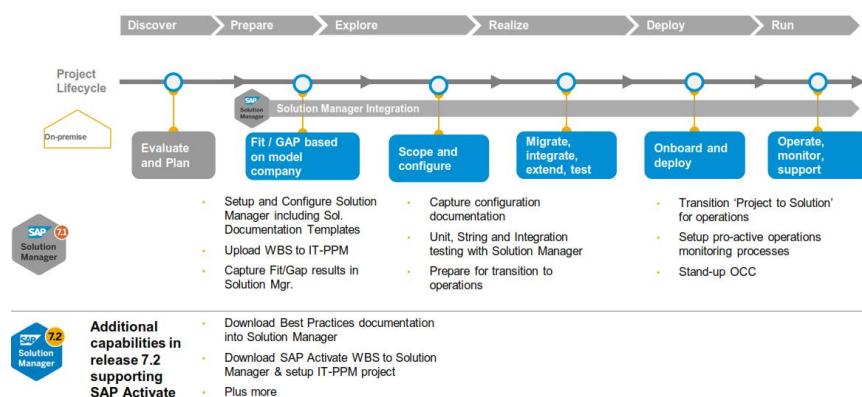
focused/results oriented	of the agile team <ul style="list-style-type: none"> • Every activity must be beneficial to the main objective • Client needs related to software documentation is treated as requirements 	(early progress demonstrated) <ul style="list-style-type: none"> • Less distractions
--------------------------	--	---

Nevertheless, a satisfying understanding in what circumstances agile development should be used and reasons for its effectiveness are still missing. This is reflected in the repeated calls for more theory-based investigations and more rigorous industrial studies on agile software development. The agile approach follows the general trend in new product development with teams as the core building block of the development organization. Collaborative development work in teams promises greater adaptability, productivity, and creativeness as compared to individuals. Moreover, it provides better solutions to organizational problems. Every good concept needs a strong underlying logic serving as a “theoretical glue”. In ISD research, however, it seems that “almost every piece of research adopts a unique interpretation of agility”. This conclusion was confirmed by who found 17 different definitions of the term after reviewing agile studies in the main outlets of software engineering in information systems research streams. [22]

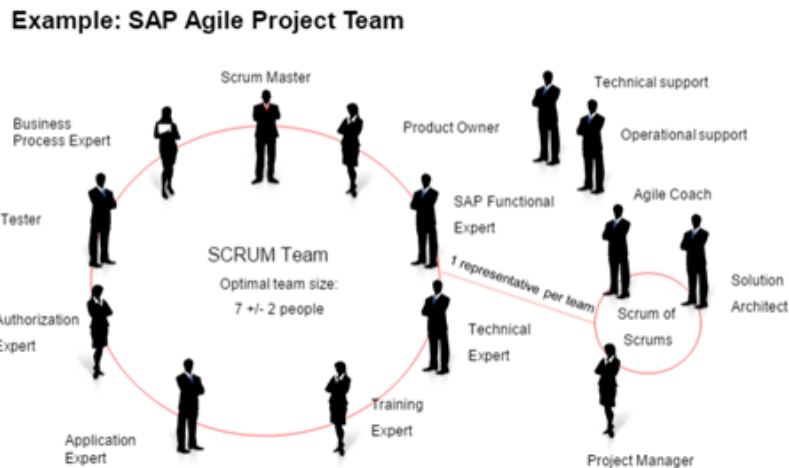
2 Agile Software Engineering Practices and ERP: SAP Activate Methodology

The SAP ASAP 8 methodology comprises of six phases as highlighted in Figure 2-3, which is a disciplined approach to managing complex projects, organizational change management, solution management, & industry specific implementations. The SAP ASAP 8 methodology is the enhanced Delivery model with templates, tools, questionnaires, and checklists, including guide books and accelerators. ASAP 8 empowers project teams to utilize the accelerators and templates built in to SAP solutions. The Agile add-on is available in SAP Solution Manager. Figure 2.1 explains various phases of SAP ASAP 8 Methodology. [23]

SAP Solution Manager in SAP Activate on-premise project



Prepare - This phase encompasses the entire project preparation and planning activities with infrastructure, hardware/network sizing requirements completed. It involves setting up the infrastructure, team, project goals, charter, and agree upon schedule, budget, risk baseline, proof-of-concept planning if applicable with implementation sequence. The project manager on the ground will discuss with the customer project manager to identify risks early on with a mitigation plan. The PM will be responsible for drafting a high-level project plan with all milestones with a detailed task level plan chalked out with critical dependencies. Each phase deliverable should be agreed between both parties. Finally, a project organization, steering committee is organized with assigned resources. Example of SAP Agile Project Team is shown in Fig. 1.2. [23]



Explore - This is the most crucial phase of the project for a project manager as he just about to steer the ship, like a captain. The objective of this phase is to be on a common platform on how the company plans to run SAP for their business operations. Thus, a PM is responsible for analyzing the project goals and objectives and revise the overall project schedule if required. In simple terms, it is the critical requirements gathering phase, A PM might use appropriate tools to collect requirements with required traceability. The result is the Business Blueprint, which is a detailed flow of business process AS-IS, how they run the business operations with a TO-BE mapped in SAP, on how these business operations will run in SAP. Depending on the implementation complexities, number of business process, Blueprint workshops might span for a few days or weeks or even months, in a complex environment. The output of this phase is the baseline configuration in SAP with detailed custom code requirements analysis done. [25]

Realize - In simple terms, realization is the actual development phase of the project, where you'd configure, develop custom code and conduct required testing. It involves

coding-unit testing-integration testing-User acceptance testing (UAT). As per the business blueprint and mapping the SAP system as agreed with business, all the business process requirements will be implemented. In reality, there are two major work packages: (a) Baseline (major scope); and (b) Final configuration (remaining scope). The success of any implementation project relies on how closely you're able to develop custom code, test and release it to the UAT phase, in order to support adequate testing by the users. Also, the challenge is to adopt changes as indicated during the UAT. This phase is resource intensive and the team is at peak team size to ensure all deliverables are met and sign-off. Often times Integration fail due to lack of test data, and testing in a "PRD" like environment to be able to test all critical business scenarios. A good practice is to copy a "PRD"-like environment and start testing if the system already exists. If it is GreenField environment, ensure adequate test data is available to test it rigorously. [26]

Deploy - Final preparations before cutover to production ensure that that the system, users, and data are ready for transition to productive use. The transition to operations includes setting up and launching support, then handing off operations to the organization managing the environment. [27]

3 Pilot SAP Agile project within the company: Inventory Aging Solution in SAP without implementation of material batch management

Javno preduzeće Elektroprivreda Bosne i Hercegovine d.d. Sarajevo (Public Enterprise Electric Utility of Bosnia and Herzegovina) is a joint stock company in which 90% of the capital is owned by the Federation of BiH, and 10% is owned by minority shareholders. Since 2009 Elektroprivreda BiH d.d. Sarajevo has had a status of the parent company in the EPBIH Concern, which is connected with several companies in the field of mining and manufacturing of equipment. [28]

Electric utility activities performed by JP Elektroprivreda BiH dd Sarajevo are:

- Generation and distribution of electricity
- Supply of Electricity
- Trading, representation and mediation on the local electricity market
- Export and import of electricity, including the management of electricity system [37]

Electric utility activities performed by the Company as public services are:

- Generation of electricity for unqualified (tariff) customers
- Electricity distribution
- Electricity supply for unqualified (tariff) customers.

- JP Elektroprivreda BiH d.d. - Sarajevo is the largest electric utility company in BiH. [38]

Key indicators are:

- Installed generating capacities 1682 MW;
- Distribution lines 27.405 km;
- Distribution substations 2.825 MVA;
- Number of customers 744.029;
- Number of employees 4.789. [39]

Internal audit asked for a custom business intelligence report to track the aging of her products in quarterly buckets (1-3 months, 4-6 months, etc.). Auditors could not afford to assume that quantity of stock (and average price) on hand is actively moving due to one recent goods receipt, as suggested by SAP Standard Content. [29]

While the standard content for SAP ERP is immensely useful for quickly implementing common solutions, and getting critical reports into the hands of business users, the out-of-the-box material do not address specific issues due to the different nature of each industry.

Internal SAP Consultants did not have any experience with Agile implementation methodology, so the external company specialized in Scrum provided its expertise and organize project teams and procedures according to agile premises.

Some of the fears that SAP Internal consultants expressed:

- Implementation teams are not programmers
- Minimum viable product is “monster”

The stock age calculation is load-intensive because it happens at the lowest level and is a back calculation. BW is meant for this type of reporting, while R/3 could bog down on the calculation. With BW, you can also change the age buckets and, with some formula variables, leave it to the user to decide which age buckets to use.[30]

This query works at the level at which material movements take place. If batch management is implemented, then the lowest level is plant/material/batch. Without batch management, the lowest level is plant/material. With this query, you can trace a material back based on the material documents. If a material document changes the identity of the material in any way — for example, a material-to- material transfer — then this query won't work for that material as the documents cannot be traced back. However, this approach is difficult to implement at the programming level because of the book-keeping overhead. There isn't a direct relationship between a goods receipt and a goods issue in the logistics module. You will need to keep an account of the movements manually. [31]

Those 8 points were “acting rules” during the project:

1. Get to initial prioritization faster.
2. Improve prioritization using economics.
3. Pull work from a dynamic prioritized list.
4. Reduce the size of requirements.
5. Get to the point of writing ABAP code quickly.
6. Actively manage the work in progress.
7. Enable a smooth sustainable flow of work.
8. Enable faster feedback cycles.

4 Recognized benefits by the agile team

More flexibility during implementation - The agile approach enables you to react decisively and effectively to change. In contrast to other methods there is no blueprint that serves as the basis for the entire implementation. Agile is based on the assumption that requirements will change with time. Therefore, changes can still be proposed and effected during the implementation. [32]

All necessary knowledge in the team. Two closely linked, cross-functional Scrum teams were brought together to address this initial situation in an optimum manner. Each team consisted of ERP developers as well as SAP consultants from both SAP systems, a representative from the manufacturer of the third-party software and a tester with knowledge of eCATT. This meant that each Scrum team was able to implement functional user stories end-to-end and ensure these with automated tests. [33]

Early integration. Focusing on the minimum viable product/report enabled early integration of all systems. The interface between the two SAP systems was already implemented in the first sprint using a minimum data set. In the following sprints the interface was expanded step by step and the third-party software was integrated. [34]

Communication. In order to integrate the many - especially operatively affected - persons, the managers of the departments had regular info-exchange meetings with the product owners of the two teams that took the form of backlog grooming. In order to integrate users “key user groups” were formed that covered a representative cross-section of the users. The key users not only took part in sprint reviews, they also worked regularly with the Scrum team during the sprint. [35]

The Results. This mini project was completed perfectly on budget and in time. But even more important: all those involved recognized what they really needed thanks to the iterative approach and many test runs. Many of the requirements originally

specified either disappeared completely from the “wish list” or were replaced by new ones. The risk of an incompatible interface also went down drastically thanks to timely integration of the modules. The department responsible for the project was so surprised and impressed by the fast and “correct” results that it intends to carry out all its commissioned projects with Scrum in future.

Improved progress monitoring & coordination - During a typical agile implementation, daily meetings are held with the team and the interested parties. These ‘Scrum Meetings’ keep you in constant contact with the project and allow you to follow the progress closely. Various checkpoints (demos) ensure that the SAP system links in with your requirements. After each clearly defined sprint, and the working software it delivers, it is clear what progress has been made and what still needs to be done. In this way risks and problems are identified at an early stage of the project.

Conditions for success

Agile is based on a short implementation cycle and high speed. This requires constant feedback and attention from the ‘Process Owner(s)’ who represent the business stakeholders. Important conditions for success are therefore the involvement from the business, a clear picture of the requirements and priorities for the project and good technological preparation. If the correct setup is not yet present then the project focuses on this first.

Involvement of business via ‘Process Owner(s)’

Decision-making and internal coordination and communication with the various business stakeholders concerning the requirements and priorities; these are the most important tasks of a Process Owner (PO). In this way, the PO represents the customer and the requirements and links these to the implementation team. [48] The PO also administers the Delta List. This is a list of the differences between the possibilities of the baseline system and the processes and functionalities required from the point of view of the business. With this the PO gives the implementation team clear priorities for the various Delta requirements. Finally, it is crucial that the availability of the PO is frequent enough during the project. If there are several Process Owners, we ask for one chief PO for mutual benefit who can take a final decision if interests differ.

Development and implementation standards - Agree on development and implementation standards with your implementation team. At the very beginning of the project define your standards for code version control, configuration of reporting and input layout templates, proper use of modelling capabilities, proper use of different types of logic (VBA, Script Logic, Business Rules, ABAP, Dimension Formulas, Measures), authorization modelling, transport system. We spent some time on defining these standards and it saved lots of time during the project. We used the same library of script logic functions, ABAP functions and VBA library. [1] As both templates and functions were shared between all consultants and developers there was


no need to build something new but rather adjust the existing functions and templates to fit in new requirements and requested features. [36]

5. Conclusion

This study confirms previous findings that agile software development positively influences the performance of ERP development process [51] Agile software engineering evolved from the knowledge of experienced consultants and industry best practices [52]. While the number of scientific publications demonstrates a clear interest of academic scholars, theory-based research on agile software development remains limited [53]. Despite its popularity, a theoretical understanding of agile software development is still in its infancy [54]. This study is a response to the frequent calls for more theory-based, industrial case studies on agile software development [55].

It is clear that Agile transformation requires a serious mind-set change and strong focus and commitment. With this change, Agile approach give your teams a sense of accomplishment throughout whole project and keep things transparent to the project stakeholders. Agile practice is the future (and the present already) of ERP implementation process. In the next figure 5.1 you can find some key benefits provided by Accenture. Accenture's current clients include 94 of the Fortune Global 100 and more than three-quarters of the Fortune Global 500. [56]

Key Benefits of Agile for SAP

				
Increased business adoption and engagement	Decrease in rework and wasted effort	Higher-quality end product	Reduced risk of delayed timelines	Speed to Value

References

1. Moon, Y.B. "Enterprise Resource Planning (ERP): a review of the literature," *International Journal Management and Enterprise Development* (4:3) 2007, pp 235-264.
2. Al-Mashari, M., Al-Mudimigh, A., and Zairi, M. "Enterprise Resource Planning: A Taxonomy of Critical Factors," *European Journal of Operational Research* (146:2) 2003, pp 352-364.
3. Scheurwater, M., De Swaan Arons, B., "ERP & Performance," *Compact KPMG IT advisory (2009_0)* 2009, pp 10-16.
4. Al-Mashari, M., Zairi, M., et al. (2006) 'Enterprise Resource Planning (ERP) implementation: a useful road map', *International Journal of Management and Enterprise Development*, Vol. 3, Nos. 1–2, pp.169–180.
5. Dantes, G.R., Hasibuan, Z.A., "Measurements of Key Success Factors (KSFs) on Enterprise Resource Planning (ERP) Adoption", *IBIMA Business Review Journal*, 2010.
6. Davenport, T.H. "Putting the enterprise into the enterprise system," *Harvard Business Review* (76:4), 1998, pp. 121- 131
7. Ross, J. W. and Vitale, M. R. (2000). The ERP revolution: Surviving versus thriving, *Information Systems Frontiers* 2(2): 233-241.
8. Somers, T.M., Nelson, K. (2001) "The Impact of Critical Success Factors across the Stages of Enterprise Resource Planning Implementations", In Proc of the 34th Hawaii International Conference on Systems Sciences, Vol.8, 8016, IEEE Computer Society, Washington, DC, USA.
9. Hevner, A. R. et al. 2004. Design Science in Information Systems Research. *MIS Quarterly*. 28, 1 (2004), 75-105.
10. Janssens G., Kusters, R., & Heemstra, F. (2008). Sizing ERP implementation projects: An activity-based approach. *International Journal of Enterprise Information Systems (IJEIS)*, 4(3), 25-47.
11. Møller, C., Kræmmergaard, P., & Rikhardsson, P. (2004). A Comprehensive ERP bibliography - 2000-2004. Department of Marketing, Informatics and Statistics, Aarhus School of Business, IFI Working paper series (12), 54.
12. Somers, T.M., Nelson, K. (2001) "The Impact of Critical Success Factors across the Stages of Enterprise Resource Planning Implementations", In Proc of the 34th Hawaii International Conference on Systems Sciences, Vol.8, 8016, IEEE Computer Society, Washington, DC, USA.
13. E.J. Umble et al. (2003), "Enterprise resource planning: Implementation procedures and critical success factors, *European Journal of Operational Research* 146 241–257
14. Markus, M.L. and Tanis, C. (2000) The enterprise systems experience – from adoption to success. In *Framing the Domains of IT Research: Glimpsing the Future Through the Past*, 264 Markus et al. Past, Zmud, R.W. (ed.) (Pinna ex Educational Resources, Cincinnati, OH), 173–207.
15. Dolmetsch, R., T. Huber & Fleisch, E. (1998). Accelerated SAP 4 Case Studies, Institute for Information Management, University of St. Gallen.
16. Bhattacharjee, A. (2000). Beginning SAP R/3 Implementation at Geneva Pharmaceuticals, *Communications of the AIS* Vol. 4,
17. NoJarvis, C. B., MacKenzie, S. B., & Podsakoff, P. M. (2003). A critical review of construct indicators and measurement model misspecification in marketing and consumer research. *Journal of Consumer Research*, 30(2), 199–218.
18. Jugdev, K., & Müller, R. (2005). A retrospective look at our evolving understanding of project success. *Project Management Journal*, 36(4), 19–31.

19. Jung, D. I., & Sosik, J. J. (2003). Group potency and collective efficacy examining their predictive validity, level of analysis, and effects of performance feedback on future group performance. *Group & Organization Management*, 28(3), 366–391.
20. Kan, S. H. (2003). *Metrics and models in software quality engineering*. Boston, MA: Addison-Wesley.
21. Kang, H.-R., Yang, H.-D., & Rowley, C. (2006). Factors in team effectiveness: Cognitive and demographic similarities of software development team members. *Human Relations*, 59(12), 1681–1710.
22. Karekar, C., Tarrell, A., & Fruhling, A. (2011). Agile development at ABC: What went wrong? In *Americas Conference on Information Systems*.
23. <https://support.sap.com/ja/support-programs-services/methodologies/implement-sap/asap-implementation.html>
24. <https://blogs.sap.com/2013/09/17/the-all-new-asap-8-methodology/>
25. <https://archive.sap.com/documents/docs/DOC-8032>
26. <http://www.scmfocus.com/saprojectmanagement/>
27. <https://blogs.sap.com/2013/09/17/the-all-new-asap-8-methodology/>
28. Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B., & Slaughter, S. (2001). How internet software companies negotiate quality. *Computer*, 34(5), 51–57.
29. Baskerville, R., & Pries-Heje, J. (2004). Short cycle time systems development. *Information Systems Journal*, 14(3), 237–264.
30. Baskerville, R., Ramesh, B., Levine, L., Pries-Heje, J., & Slaughter, S. (2003). Is internet-speed software development different? *IEEE Software*, 20(6), 70–77.
31. <https://www.linkedin.com/pulse/20140826102817-5677495-sprints-agile-approach-in-sap-implementations>
32. <http://www.prosoftnearshore.com/5-types-of-scrum-meetings/>
33. <https://www.handshake.com/blog/erp-implementation/>
34. <https://blogs.sap.com/2014/10/24/agile-for-sap-implementation-yes-of-course/>
35. <http://www.r3now.com/agile-software-development-for-sap-erp-projects/>.