Taking Advantage of Business Process Management Approaches in Requirements Engineering

An Assessment Model based on Process Characteristics

Matthias Lederer[1], Werner Schmidt[2], Albert Fleischmann[3], Remzi Avci[4]

Abstract: Many established techniques and modeling methods for business process management (BPM) are known. Since many of them are domain-independent, they can also be applied for requirements engineering within software development processes. In order to build a bridge between requirements engineering and the BPM body of knowledge, this paper first identifies 26 basic process characteristics (e.g., duration of cycle times). A classification is created, which recommends built-time methods for agile as well as traditional (e.g., waterfall model) software development processes. For both types, this contribution discusses the application of the most suitable BPM techniques (e.g., case management for agile processes vs. variants of BPMN for traditional development processes).

Keywords: Business Process Management, Software Development Process, Software Engineering, Pre-deployment Techniques, Process Characteristics.

1 Motivation

Software development processes (SDP) are used to handle a structured and professional implementation of requirements engineering (RE) [WK00]. The orientation towards business process management (BPM), understood as the organization of activities (e.g., requirements engineering), events (e.g., specification sheet approval), resources (e.g., change request databases) and individuals (e.g., developers), is an essential core characteristic of Software Engineering (SE) [Mü12]. In literature, it is widely accepted, that the explicit application of BPM techniques to SE makes a significant contribution to the professionalization required in business practice (e.g. [LT99;Vo10;HKW14;Mü12]). As [Vo10] point out in their trend study, it has always been the core of BPM to model processes that transfer the requirements of users (process customers) into the development of software (process participants). In particular for RE, there are numerous new methods discussed in the BPM field. Many approaches are domain-independent or transferable to other business areas. As a consequence, RE as essential part of SE can

[1] ISM International School of Management, Karlstraße 35, D-80333 Munich, matthias.lederer@ism.de
[2] Technische Hochschule Ingolstadt, Esplanade 10, D-85049 Ingolstadt, werner.schmidt@thi.de
[3] Albert Fleischmann, InterAktiv Unternehmensberatung, Oldenburger Str. 231, D-26203 Wardenburg, albert.fleischmann@interaktiv.expert
[4] Universität Erlangen-Nürnberg, Lange Gasse, D-90404 Nuremberg

benefit from two fundamental different groups of BPM techniques[5] [Fe05], which are also differentiated in known maturity models:

- **Pre-deployment**: Numerous techniques exist for the efficient assessment of processes (e.g., notations, workshop organization rules, role models) before it is actually implemented. Many case studies show that simple improvements as well as underlying resources may already be detected by modeling (“build”) the actual state of a process [Ko10].
- **Post-deployment**: Numerous methods (e.g., scorecards) as well as tools (e.g., test automation software) are used in SE during the execution (“running”) of the SDP. In BPM, however, many domain-independent best practices are known to support the organizational and technical handling of processes after deployment (e.g., workflow and groupware management systems).

In the lifecycle of a process [LSK17], these two groups distinguish recognized methods according to whether they are supportive in the planning of processes or its implementation. Studies (such as those by [LT99;Vo10]) show that many companies can already achieve significant professionalization by modeling workflows. Such findings are also known in publications on SE [LC06]. This paper therefore focuses on the first group mentioned because they are also relevant in requirements engineering.

In RE, two basic paradigms (traditional vs. agile development processes) are known [GSM06]. Coming from a process-oriented view on these paradigms, they have fundamentally different workflow characteristics (for example robustness vs. flexibility, pre-definition of processes vs. knowledge-based ad-hoc decisions). Also, their goals (e.g. communicated in an i*-model) result in fundamentally different process characteristics. Whether a user is allowed to ask for continuous adjustments (agile procedure) or whether an executable software exists only at the end of the development (traditional procedure) is known in advance as a basic decision. Accordingly, corresponding BPM techniques may not be equally well suited for both traditional and agile development processes.

In order to be able to assess which BPM techniques can be used to professionalize a procedure, often process characteristics are used. In evaluations, [Le16] and [LHB15] showed that such central process properties are generally known in advance (for example, because they are dictated by a classical or agile paradigm). Moreover, it proven for many situations that their consideration supports the selection of BPM techniques. For example, [LHB15] successfully showed scenarios for the selection of executing software for a given set of process characteristics. In addition, numerous studies show that essential process properties in the SE affect the suitability of BPM techniques [KN08;HLB14;BRZ05]. This is discussed in particular in the step of the RE.

[5] The terms “pre-deploy” and “post-deployment” are used in this publication. Both are both known in the jargon of the BPM and the SE community. However, there are concepts and terms that are more appropriate for each of the disciplines (e.g., “build-time” vs. “run-time” or “modeling” vs. “implementation”).

It should be up to this contribution to clarify, which BPM pre-deployment techniques can successfully lead to added value for which type of SDP.

This paper will therefore answer the following questions:

1. Which basic process characteristics determine the suitability of business process management pre-deployment techniques?
2. Which pre-deployment techniques are suitable for the fundamental types of software development processes?

Section 2 briefly describes the related work. To answer the first question, section 3 outlines a general model, which links pre-deployment techniques with fundamental process properties. This model is then used in Chapter 4 to give an assessment of which software development processes should rely on which BPM findings. Chapter 5 expands the theses of this contribution by showing further steps for bridging between BPM and RE.

Both questions will be answered in one workshop contribution to provide researchers and practitioners with coherent arguments. Question 1 lays the theoretical and scientific foundation for a fundamental link between existing BPM and SE knowledge. The exemplary discussion on the results (question 2) gives practitioners the chance to benefit directly from the findings of question 1. Scientists receive a basis for argumentation in order to get into the detailing of the techniques and procedures.

2 Related Work

Software development is performed in processes [SL02]. While these are different in their basic requirements, it is agreed that small (e.g., prototyping) and large software developments (e.g., according to the waterfall model) follow a fundamental process model [SL02;RBP09].Nevertheless, the major part of science and research on SDPs is related to the classical project management [BS06;RBP09]. On the one hand, primarily methods of project management (e.g., personnel planning, time monitoring, quality measurement) are adapted to the requirements of SE. From a BPM perspective, on the other hand, individual software developments should be seen as instances of a development process [AS06]. When authors from the SE domain take this process-oriented view, their focus is mostly on technical or normative aspects. Technical publications show, for example, theoretical models for software quality, mathematical simulation possibilities for development processes or algorithms for the improvement of coding. The BPM literature pursues not only technical aspects but also mainly investigates organizational aspects. In terms of normative aspects, SE publications aim to implement process-oriented frameworks as best practices. For example, experts from the SE often describe ISO and CobIT processes to increase the software quality [SL02]. Hence, BPM best practices are usually seen isolated for individual processes (such as standards for change management, risk management, program management).

Fundamental BPM paradigms (e.g., adaptive case management), which could be applied to many tasks of the SE, are therefore not pursued [XK00;WK00;SL02].

There are, in fact, textbooks and studies that address the combination of BPM modeling techniques and RE. In most cases, these publications stem from SE (e.g.,[AS06]). It is noticeable that they were published especially in the years of 2000 to 2010. This could be due to the fact that in this time more aspects of open development processes (such as open source software) were addressed [Ko05]. This shift away from closed innovation – similar to the idea of human-centered computing [SGD06] – depends on the interaction between people. These can be customers or volunteers, who play an active role in the process (e.g., advisor or developer). BPM publications have been addressing such challenges of openness with specific solutions for a number of years under the slogan of Social BPM and BPM 2.0 [LSK17].

3 Domain-Independent Selection of Pre-deployment Techniques

3.1 Methodology

Based on a qualitative literature analysis, publications were identified and codified. Following the inductive approach by [Ma08], first relevant databases were selected, then search terms were used to come up with a comprehensive database. To answer the first research question, a comprehensive list of pre-deployment techniques was developed. Following the definitions of [Al05;La94], we subsume under this term all textual and graphical methods, notations, best practices, industry standards, and procedures to plan and design the actual state of a process [Wi05]. Several scientific databases were investigated, in which typical state of the art knowledge from the BPM discipline is suspected (e.g., Business Source Complete and SpringerLink). No results have been taken into account from before the year 2012. To find relevant results, the search term “BPM” (including synonyms) was combined with descriptions for pre-deployment (e.g., “modeling”, “acquisition”, “elicitation”, “identification”).

The same methodology and the same databases were used to identify the process characteristics. However, the steps of an inductive category building were carried out for the collection and listing following the methodology by [EK08]: In contrary to the techniques, which are exposed to be part of trends and also hypes, the literature recognizes that at a high-level view (e.g., strategy formulation) processes already have a fixed set of known characteristics. They serve as the basis for the selection of analytical methods, BPM tools and also control mechanisms [HLB14]. Such characteristics of a process prior to its modeling have been the subject of various studies (see section 2). A list could be created by systematically searching for combinations such as "BPM" and keywords like "properties", "structure", "mark", "feature" and "requirement". A classification scheme is generated to connect the two building blocks of the model. Using the idea of a Target Activity Grid [BW09], it was evaluated which technique can be used for which process characteristic in a table-like representation. For this purpose, two experienced experts were asked for individual assessments. Both experts have a practical background in the usage of the techniques or can estimate their range of application by own research and teaching practices. A boolean assessment is used to evaluate whether the technology can adequately satisfy a processes characteristic or not.

3.2 Results

Based on 15 publications, which give statements on basic process traits, a total of 23 *process characteristics* could be identified in the analysis (see Table 1, more names are given in Table 2). The values of the properties can be assessed partially on nominal and partially on ordinal scales (see cells in columns of Table 1). As a rule, a process needs to fall into at least one value of the process characteristic properties, but can hold more than one. For the 9 process characteristics in the row at the bottom of Table 1 no scale is provided by literature. In such cases an ordinal 3-scale was assumed, using 'high', 'medium' and 'low' as properties. This scaling is usually sufficient for use in a theoretical model since the meaningfulness of the process characteristic can be evaluated quick enough by the model user. For further processing each property value is assigned a scaling number from 1 to 5. Key findings on characteristics and their scaling are taken from [LSK17;Va03;NP11;Jo12;LHB15;Ci15;Br11]

1	2	3	4	5
PC_1 Structuring				
Structured	structured with ad hoc exceptions	structured with predefined exceptions	loosely structured	unstructured
PC_2 Process Representation				
Activity oriented	Rule oriented	Artefact oriented	-	-
PC_3 Process Implementation				
Workflow engine	Rule engine	Program control flow	-	-
PC_4 Trend orientation				
Data-driven	Case-driven	Social-driven	-	-
PC_5 Process participants				
Person to person	Person to application	Application to Applic.	-	-
PC_6 Knowledge intensity				
Knowledge-intensive	Automated/Repeatable	-	-	-
PC_8 Interrelatedness				
Linked explicitly	Inferred link	-	-	-
PC_9 Collaboration intensity				
Collaborative	Semi-collaborative	Non-collaborative	-	-
PC_11 Value repetition				
Ad hoc	Administrative	Collaborative	Production	-
PC_16 Implementation				
Big bang	Step by step	-	-	-
PC_17 Process instantiation				
Automated	Semi-automated	Manual	-	-
PC_23 Process cycle time				
Long-running	Medium	Short running	-	-
PC_7, 10,12, 13, 14, 15, 18, 19, 20, 21, 22				
High	Medium	Low	-	-

Tab. 1: Process characteristic properties and their scaling [LSK17;Va03;NP11;Jo12;LHB15;Ci15;Br11]

Among the identified *pre-deployment techniques* there are classical notations (e.g., BPMN), but also approaches that do not focus on the formal design of processes in the first place. There are established and frequently discussed approaches (e.g., BPM 2.0), but also techniques that are in the early stages of development and application (e.g., BPMN-D). A total of 31 BPM techniques is included in the model. Table 2 presents the evaluation table, which combines pre-deployment techniques (columns) with fundamental process characteristics (rows). An entry in the table cells indicates that the technique can be used when a process holds this value of the characteristic.

Characteristics \ Pre-deployment techniques	P_1: S-BPM	P_2: uBPMN	P_3: BPMN Plus	P_4: Deontic BPMN	P_5: BPMN Gamification	P_6: Ad-hoc BPMN	P_7: BPMN light	P_8: AC-BPM	P_9: ArC-BPM	P_10: BPMN4RS	P_11: BPMNDiffViz	P_12: BPMN choreography diagram	P_13: Hierarchical BPMN Miner	P_14: BPMN	P_15: BPMN (flat) Mining	P_16: BPMN-D	P_17: simulation-based BPMN	P_18: BPMN4CP	P_19: EDBPM	P_20: RrBPMN	P_21: BR + SWS	P_22: BPM Reuse	P_23: iBPM	P_24: Collaborative BPM	P_25: Social BPM	P_26: BPM 2.0	P_27: Design Thinking	P_28: ACM	P_29: CCM	P_30: ECM	P_31: BPMN-Q
PC_1 Structuring	1	1	3, 4, 5	4, 5	1 to5	4, 5	1, 2	1	1	1, 2, 3	1 to5	1	1, 2, 3	1, 2	1	3	1	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	3, 4, 5	1 to5	3, 4, 5	1 to5	4, 5	4, 5	4, 5	1
PC_2 Process Represen-tation	3	1, 2, 3	1, 3	1, 2	1, 2	1	1	1	3	1, 3	1, 2	1	1, 2, 3	3	1, 2	1, 2	1, 2	1, 2	1, 2	2, 3	2	1, 3	1, 2	1	1	1	1	1	1	1	3
PC_3 Process Implemen-tation	3	1, 2, 3	1, 3	1, 2	1, 2, 3	1	1	1, 3	3	1	1, 2, 3	3	1, 2, 3	3	1, 3	1, 2	1, 2, 3	2, 3	1, 2	2, 3	2, 3	1, 3	2	1	1	1, 3	1	1	1	1	3
PC_4 Trend orientation	1, 3	1, 2	1, 2	2	2, 3	2	2	1	1	1, 2	1	1, 2	1	1, 2	1	2	1	1, 2	2	1, 2	1	1, 2	1	3	3	3	3	2	2	2	1
PC_5 Fixation of the process participants	2, 3	2, 3	2, 3	2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2 ,3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	2, 3	1	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	2, 3
PC_6 Knowledge intensity	2	2	2	2	2	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1, 2	1	1	1	2
PC_7 Diversity of Information	2	2	1	2	2	1	2, 3	1, 2	1, 2	2	2	3	2, 3	1, 2	3	2, 3	1, 2	1, 2	1, 2	2	2	1	1	1	1	1	1	1	1	1	1
PC_8 Interrelatedness	1	1	2	2	1, 2	2	1	1	1	1	1, 2	1	1, 2	1	1	2	1	1	1	1	1	1	1	2	1, 2	2	1, 2	2	2	2	1
PC_9 Collabo-ration intensity	1	1, 2, 3	1, 2, 3	1, 2, 3	1, 2	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2	1, 2, 3	1, 2, 3	1	1, 2	1, 2	1, 2	1, 2, 3	1	1, 2	1	2	3
PC_10 Value	1	1	1	1	1	1, 2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
PC_11 Value repetition	4	4	4	4	4	1, 3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	3	3, 4	3, 4	3, 4	3	3, 4	3, 4	4
PC_12 Pre-dictability	1	1	1	1	1, 2, 3	3	1, 2	1	1	1, 2	1	1	1, 2	1	1	1	1	1	1	1	1	1	1	3	1, 2, 3	3	1, 2, 3	3	3	3	1
PC_13 Flexibility	3	3	1	1	1, 2, 3	1	2, 3	3	3	2, 3	3	3	2, 3	3	3	3	3	3	3	1, 2, 3	1, 2, 3	3	3	3	1, 2, 3	1	1, 2, 3	1	1	1	3
PC_14 Model-ability, control and automation	1	1	1, 2	2	1, 2, 3	3	1, 2	1	1	1, 2	1	1	1	1	1	1	1	1	1	1	1	1	1	2, 3	1, 2, 3	3	1, 2, 3	3	3	3	1
PC_15 Complexity	3	2	2	3	1, 2, 3	1	2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	2, 3	2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2	1, 2	1, 2	1, 2, 3
PC_16 Implementation	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2
PC_17 Process instan-tiation	1	1, 2	1, 2	1, 2	1, 2	2, 3	1, 2, 3	1, 2	1, 2	1, 2	1	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	1, 2	2, 3	1, 2	1, 2	1, 2	2	2, 3	2	2	2, 3	3	2	2, 3	1, 2
PK_18 Robustness	1, 2, 3	1, 2	1, 2, 3	1, 2, 3	1, 2	3	1, 2, 3	1	1	1, 2	1	1	1	1	1	1	1	1	1	1	1	1	1	3	3	2	3	3	3	3	1
PC_19 Adaptability	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	2, 3	1, 2	2	1	1	2	1	1, 2	1	1	1	1	2	1	3	2	1	1	2	1	2	2	1	1	1	1	2
PC_20 Adaptivity	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	2, 3	1	1	1, 2	1	2, 3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2
PC_21 Selection	1, 2	1, 2, 3	1, 2, 3	1, 2, 3	2, 3	1, 2	2, 3	1, 2	1, 2	2	1, 2	2, 3	1	1, 2	2	2, 3	1	1	2	1	1	1	1	1	1	1	1	1	1	2	2
PC_22 IT needs	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1, 2	1	1	1	1	2, 3	1	1, 2	1, 2	1, 2	1, 2	1, 2, 3	1
PC_23 Process cycle time	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3	1, 2, 3

Tab. 2: Classification scheme combining process characteristics with pre-deployment techniques

4 Application for Software Engineering

After the general classification approach was developed, it is now used to answer the second research question.

4.1 Methodology

For company-independent application, general assessments must be filled into the model. This means it is necessary to assess the extent to which the two fundamentally different SDPs (traditional and agile) follow the process characteristics developed in the last section. In order to avoid company-specific peculiarities in the user input, which might influence the overall assessment, two scientists have made the assessment. With a total of more than 70 publications and more than 30 years of research as well as teaching experience in the field of process management, the two researchers have a broad experience in the BPM and SE field. The mini Delphi study serves as method. After their assessment, individual deviations were subject of a face-to-face discussion. Although it is a qualitative research, the tabular classification is also suitable for the application of a simple scoring algorithm. The number of hits (h) per technique does not give any proof in a quantitative sense, but it is intended to support the discussion as a general assessment.

4.2 Results

As a result of the methodology, the characteristics of the two SDPs stereotypes can be obtained. The order of the evaluations presented can be understood as a vector of the 23 properties from Table 2.

- *Traditional Process Development Processes*: The expert assessment for the traditional SDPs (e.g., waterfall model) is as follows[6].

  [(2);(3);(1,3);(1,2,3);(3);(2);(3);(1);(3);(1);(2);(1);(3);(1);(3);(1);(1);(1);(1);(2);(3);(1);(3)]

- *Agile Development Processes*: For agile SDPs (for example according to the idea of SCRUM), the expert assessment led to the following values.

  [(4);(3);(2);(1,2,3);(1);(1);(1);(2);(1);(3);(1);(3);(1);(3);(1);(2);(3);(3);(3);(1);(1);(3);(1)]

[6] The values in brackets refer to the scaling number of the property (cell entry in each row of Table 1), identified as valid for 'traditional'.

5 Discussion

In the following paragraphs, an excerpt of the resulting recommendations for or against specific pre-deployment techniques is discussed. Table 3 outlines the top 3 recommendations. The number of hits results from counting.

Agile		Traditional	
Techniques	Hits	Techniques	Hits
Ad-hoc BPMN	18	BPMN choreography diagram / Hierarchical BPMN Miner	19
Collaborative BPM / Design Thinking / Adaptive Case Management (ACM)	17	uBPMN / BPMN light / BPMN-D	18
Social BPM / Collaborative Case Management (CCM)/ Emergence Case Management (ECM)	16	BPMN Gamification / BPMNDiffViz / Modelling Software Processes using BPMN / Process mining (flat) / BPMN4CP / BR+SWS / BPM Reuse	17

Tab. 3: Classification scheme combining process characteristics with pre-deployment techniques

The discussion is intended to give first indications[7], which areas of application and possibilities result from essential BPM techniques in the two SDPs. This section will in particular show that the BPM techniques support the general mind-set behind the different approaches in the SE with special attention to RE.

5.1 Traditional Software Development Processes

Since the classical software development requires many departments to collaborate both professionally and technically, the *BPMN choreography diagram* (h=19) is particularly suitable. It models the exact and complete control flow across department or team boundaries (e.g., specification, testing) with a particular focus on communication. In the stages of the waterfall model, the process manager has to have an overview of the stages of development, but the internal behavior of subjects are not that relevant in his/her view. Since it is not stipulated in the traditional view of software development to step backwards (for example development only after the complete specification), this technique supports collaboration in SDPs with a particularly formal technique. If this explicit pre-modeling is not possible, the technique of the *Hierarchical BPMN Miner* (h=19) is suitable. It is known that the pre-definition of fundamental development stages (e.g., as milestones) is possible. However, many further developments of traditional SE (for example, the V-model XT) have resulted in particular from the realization that not all activities can be determined exactly (e.g., which requires all stakeholders to meet to come up with the requirements). The recommended pre-deployment technique supports this weakness by automatically analyzing and comparing BPMN models. In this sense, it is understandable why *BPM Reuse* (h=17) is recommended as a pre-deployment technique as well. This is due to the fact that satisfying process control flows (e.g.,

[7] In further research (see also section 6), it is necessary to examine the specifications for individual combinations (SDPs including sub-categories and BPM technique including further developments), how the co-operation (e.g., use of requirements specifications) can be applied in practice.

development instances that produced high-quality software within the given budget) must be recognized and imitated. By doing so, this technique supports one of the major challenges of SE since its discipline foundation. Following the same consideration, the recommendation of *uBPMN* is very suitable (h=18). The idea behind this further development of BPMN lies in the continuous technology implementation in many processes activities. Ubiquitous computing technology is currently a much-debated form of IT-enabled information processing, which is entering more and more areas of daily life and work (trend of “digitization”). In software development, many steps have always been PC-supported. Also, typical human-centric steps (e.g., a story telling workshop) result in digital information (e.g., requirement lists), which is passed on to the next development stages. An interesting recommendation is the notation *BPMN-D* (h=18). Once again, the BPMN standard is being further developed. However, the normative character of the notation is successively reduced by the inclusion of declarative elements. This means that both imperative (i.e., fixed control flows, events, and communication flows) as well as optional and soft content can be modeled. This technique paves the way for agile methods, which allow for the clearance and implementation of activities (e.g., the order of implementation of requirements). *BPMN light* (h=18) follows the same tendency, when it aims at that not all contents of the development process can or must be modeled in detail. Rather, this technique focuses on the fact that the process model is well understood by users. This simplification is an important point in classical software development: process models are often powerful and large. However, they must be understood by the teams (e.g., developers) to secure compliance.

5.2 Agile Software Development Processes

As many properties for agile methods fundamentally differ from those of the traditional SDP, the recommendations are almost inverse. The handling of individual requirements (e.g., written in user stories) in short sprints goes along with the core idea of case management. Instead of defining all necessary steps to achieve a final result (i.e., the executable software) in advance, individual requirements (e.g., features, masks, data streams) are implemented one after another. Agile approaches (e.g., SCRUM) can therefore apply the idea of *Adaptive Case Management* (h=17), in which the modeling (i.e., planning of the team organization) is combined with the execution (i.e., actual software development) of workflows. The peculiarity of the adaptivity is to provide good workflow parts (e.g., good activity combinations) as reusable templates. They can be accessed again if the requirements of another instance (e.g., sprint planning) are similar. This case approach, which has been laid down in medicine for decades, has also received increased popularity in SE in recent years. The same applies to aspects addressed within *Emergence Case Management* (h=16) as a sub-form. *Ad hoc BPMN* receives the highest value in the recommendation (h=18). The application of the technique is certainly not useful for overall development (e.g., steps of sprint planning, daily SCRUM, development and a retrospective). Rather, this technique can be applied in the individual sprint parts, if the exact way to implement a feature is not yet known - similar to cases.

For example, ad hoc BPMN allows changes in the parallelization of tasks. It fits well into the basic conditions of agile development (e.g., on-site customer, team in a room, pair programming). What is striking is that, with *Social BPM* (h=16) and *Collaborative Case Management* (h=16), techniques are recommended that rely heavily on the competences of the process teams. The common idea of both techniques is that people involved in the process (for example developers, testers) should influence the design of the processes. This means that these individuals no longer merely assume the execution of the processes, but can also provide an important input for improvements in the pre-deployment phase. It therefore corresponds precisely to the core idea of the agile manifesto. With concrete SCRUM methods, such as the retrospective, many agile development processes follow this social BPM view. The recommendation of the relatively new method of *design thinking* (h=17) for agile SDPs is easily comprehensible. In fact, there is a large interaction between many agile methods to generate and document customer needs (such as story-telling for requirements engineering) and techniques from design thinking (such as customer journeys for understanding a customer’s view).

6 Summary and Outlook

Based on the idea that numerous good practices and methods from the field of process management can be used for SE, 31 pre-deployment techniques for the design of processes have been identified. Since traditional and agile SDPs are fundamentally different, a general classification scheme for an assignment has been developed. Based on 28 typical process characteristics, the most appropriate methods for each SDP type were discussed. While BPMN including various further developments can be used for example in the waterfall model, case- or social-driven techniques are suitable for agile approaches such as SCRUM. Although we used a common approach for literature analysis [Ma08] and relevant databases as well as comprehensive search words, there might be more material to exploit. It is up to future work to enhance the database in order to further test the hypothesis. An ongoing evaluation by the authors, which uses recommended techniques in real-life scenarios, may provide further practical guidance in further publications. In addition, it should be the subject of further research how the actual and internal collaboration may look between the recommended BPM techniques and the SE steps (e.g., how, when, and where exactly functional or non-functional requirement statements can be used in which BPM step). This contribution provides the general scope of such investigation. In further research, also attempts can be made to extend the methodology for post-deployment techniques. In addition, further studies may provide quantitative evidence for the findings of this research.

7 References

[Al05] Allweyer, T.: Geschäftsprozessmanagement, Controlling. W3l, Herdecke, 2005.

[AS06] Acuna, S.T.; Sánchez-Segura, M.I.; New Trends in Software Process Modeling. World Scientific, New Jersey, 2006.

[BR11] Brambilla, M.; Fraternali, P.; Ruiz, C.V.:A model-driven approach to Social BPM applications. Social BPM Handbook, BPM and Workflow Handbook Series 2011, pp. 95-112, 2011.

[BRZ05] Boehm, B.; Rombach, H.D., Zelkowitz, M.V.: Foundations of Empirical Software Engineering. Springer, 2005.

[BS06] Bittner, K.; Spence, I.; Managing Iterative Software Development Projects. Addison-Wesley, Boston, 2006.

[BW09] Best, E.; Weth, M.: Geschäftsprozesse optimieren. Gabler, Wiesbaden, 2009.

[Ci15] Di Ciccio; D.; Marrella, A.; Russo, A.: Knowledge-Intensive Processes: Characteristics, Requirements and Analysis of Contemporary Approaches. Journal on Data Semantics 4, pp. 29-57, 2015.

[EK08] Elo, S.; Kynglas, N.: The qualitative content analysis process. Journal of Advanced Nursing 62, pp. 107-115, 2008.

[Fe05] Ferstl, O.; Sinz, E.J.; Eckert, S.; Isselhorst, T.: Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety. Wiesbaden, Springer, 2005.

[GSM06] Geras, A.; Smith, M.; Miller, J.: Configuring Hybrid Agile-Traditional Software Processes. In: (Abrahamsson, P; Marchesi, M.; Succi, G. Eds.): Proceedings of the 7th International Conference Extreme Programming and Agile Processes in Software Engineering. Springer, Oulu, pp. 104-113, 2006.

[HKW14] Heinrich, R.; Kirchner, K.; Weissbach, R.: IEEE 1st Int. Workshop on the Interrelations between Requirements Engineering and Business Process Management (REBPM). Proceedings, IEEE, 2014.

[HLB14] Huber, S.; Lederer, M.; Bodendorf, F.: IT-enabled Collaborative Case Management: Principles and Tools. In (Smari, W.W.; Fox, G.C.; Nygard, M. Eds.): Proceedings of the 2014 International Conference on Collaboration Technologies and Systems. IEEE, Minneapolis, pp. 259-266, 2014.

[Hu04] Huo, M.; Verner, J.; Zhu, L.; Babar, M.A.: Software Quality and Agile Methods. In (SC Eds.): Proceedings of the 28th Annual International Computer Software and Applications Conference. IEEE, pp. 1-6.

[Jo12] Joachim, M.: Process Measurement in Business Process Management: Theoretical Framework and Analysis of Several Aspects. KIT Scientific Publishing, Karlsruhe, 2012.

[KN08] Koch, S.; Neumann, C.: Exploring the Effects of Process Characteristics on Products Quality in Open Source Software Development. Journal of Database Management 19, pp. 3008-3035,2008

[Ko05] Koch, S.: Free/open Source Software Development. Idea Group, Hershey, 2005.

[Ko10] Kobler, M.: Qualität von Prozessmodellen. Logos, Berlin, 2010.

[KT05] Kasi, V.; Tang, X.: Design Attributes and Performance Outcomes: A Framework for Comparing Business Processes. In (SAIS Eds.): In Proceedings of the 10th Southern Association for Information Systems, 2005.

[LA94] Leymann, F.; Altenhuber, W.: Managing business processes as information resources. IBM, 1994.

[LC06] Lee, M.C.; Chang, T.: Applying TQM, CMM and ISO 9001 in Knowledge Management for software development process improvement. International Journal Services and Standards 2, pp. 101-115, 2006.

[Le16] Lederer, M.: Business Process Transparency Management. University Erlangen-Nürnberg, Nürnberg, 2016.

[LHB15] Lederer, M.; Huber, S.; Bodendorf, F.: BPM-Tools im Überfluss – Wie strategische Kriterien die Auswahl des richtigen IT-Werkzeugs beeinflussen. In (Berglehner,F.; Wilbers, K. Eds.): Schulisches Prozessmanagement, Nürnberg: Texte zur Wirtschaftspädagogik und Personalentwicklung, 2015.

[LK05] List, B.; Korherr, B.: An Evaluation of Conceptual Business Process Modelling Languages. In (ASC Eds.): Proceedings of the 2006 ACM symposium on Applied computing. SAC, Dijon, pp. 1532-1539.

[LSK17] Lederer, M.; Schott, P.; Knapp, J.: The Digital Future has Many Names - How Business Process Management drives the Digital Transformation. In (IEEE Eds.): Proceedings of the 6th International Conference on Industrial Technology and Management. IEEE, Cambridge, pp. 22-26, 2017.

[LT99] Luo, W.; Tung, Y.T.: A framework for selecting business process modeling methods. Industrial Management & Data Systems 99, pp.312-319, 1999.

[Ma08] Mayring, P.: Qualitative Inhaltsanalyse. Beltz, Weinheim, 2008.

[Mü12] Münch, J. et al.: Software Process Definition and Management, Berlin: Springer, 2012.

[NP11] Niehaves, B.; Plattfaut, R.: Collaborative business process management: status quo and quo vadis. Business Process Management Journal 17, pp. 384-402, 2011.

[RBP09] Royce, W.; Bittner, K.; Perrow, M.: The Economics of Iterative Software Development: Steering Toward Better Business Results. Pearson, 2009.

[SGD06] Seffah, A.; Gulliksen, J.; Desmarais, M.C.: Human-Centered Software Engineering. Springer, Heidelberg, 2006.

[SL02] Stiller, E.; LeBlanc, C.: Project-based Software Engineering. Addison Wesley, Boston, 2002.

[Va03] van der Aalst, W.: Business Process Management. ISRN Software Engineering, 2013, pp. 1–37, 2003.

[Vo10] vom Brocke, J. et al.: Current and Future Issues in BPM Research. Communications of the Association for Information Systems 28, pp. 393-412, 2010.

[Wa09] Wang, Q. et al.: Trustworthy Software Development Processes: International Conference on Software Process. Springer, Vancouver, 2009.

[Wi05] Wittges, H.: Verbindung von Geschäftsprozessmodellierung und Workflow-Implementierung. DUV, Berlin, 2005.

[WK00] Wang, Y.; King, G.; Software Engineering Processes. CRC, London, 2000.