Research Challenges for Business Process Models at Run-Time

David Redlich[1,2], Gordon Blair[2], Awais Rashid[2], Thomas Molka[1,3], and Wasif Gilani[1]

[1] SAP Research Center Belfast, United Kingdom
{david.redlich,thomas.molka,wasif.gilani}@sap.com
[2] Lancaster University, United Kingdom
{gordon,marash}@comp.lancs.ac.uk
[3] University of Manchester, United Kingdom

Abstract. Today's fast and competitive markets require businesses to react faster to changes in its environment, and sometimes even before the changes actually happen. Changes can occur on almost every level, e.g. change in demand of customers, change of law, or change of the corporate strategy. Not adapting to these changes can result in financial and legal consequences for any business organisation. IT-controlled business processes are essential parts of modern organisations which motivates why business processes are required to efficiently adapt to these changes in a quick and flexible way. This requirement suggests a more dynamic handling of business processes and their models, moving from design-time business process models to run-time business process models. One general approach to address this problem is provided by the community of *models@run.time*, in which models reflect the system's current state at any point in time and allow immediate reasoning and adaptation mechanisms. This paper examines the potential role of business process models at run-time by: (1) discussing the state-of the art of both, business process modelling and models@run.time, (2) reflecting on the nature of business processes at run-time, and (3) most importantly, highlighting key research challenges that need addressing to make this step.

Keywords: run-time models, business process models, business process management, adaptive systems, business process optimisation.

1 Motivation

Business processes and business process models play a central role in modern businesses. In the early years of computer-aided management of business processes it was assumed that business processes do not change frequently during their execution. While this might be true for static processes, e.g. at the strategic level, less rigid processes, mostly found on the operational level, can be the subject of frequent changes. Processes of the latter type might need to adapt to dynamic changes [44] in environment (e.g., lack of available resources) or in flow

N. Bencomo et al. (Eds.): Models@run.time, LNCS 8378, pp. 208–236, 2014.

of work (e.g. introduction of a new activity). In fact, today's businesses have to act in a highly competitive environment which comes with strict requirements with regards to fast adaptations and optimisation. If changes in the environment happen a business has to act accordingly and potentially also adapt their core business processes to stay competitive or even outmatch their competitors. One example is a hospital, in which, based on the qualifications of the current staff and demand of the patients, the treatment process has to be adapted in case of a sudden virus outbreak. Another example is the dynamic field of security, almost daily new threats arise and an organisation has to be prepared in order to protect its confidential assets.

As business processes are ultimately driving today's modern organisations they are likely to change and adapt at increasing speed. A late action can result in a Service Level Agreement (SLA) violation, leading to financial and legal consequences. To prevent this from happening businesses have to deal with the following two general challenges[1]:

- **Need to adapt to changing demands:** A business organisation and its processes have to be flexible and continuously adapt to changing demands exposed to by internal or external sources. Adaptations need to be accurate and reliable in order to actually improve the current situation.
- **Need to shorten the business process life cycle:** The process of designing, configuring, deploying, and analysing a business process [54] should become further simplified, i.e. more automated.

Hence, a more dynamic handling of business processes is desirable, moving from design-time business process models to run-time business process models. One approach to address this problem is provided by the community of models@run.time, in which models reflect the system's current status at any point in time and allow immediate reasoning and adaption mechanisms. This paper is a first attempt to raise the abstraction level of models@run.time to the domain of business processes. This will, for one, contribute to research in models@run.time by providing a valid use-case as well as further requirements for models@run.time of a high abstraction level and, secondly, help to address the general challenges of business adaptation and automation.

The remainder of this paper is structured as follows: In Section 2 background information of the business process management domain necessary for the understanding of this paper is summarised. In Section 3 the state of the art with regards to the topic of business process models at run-time and the general challenges identified earlier is reviewed: business process modelling standards, business process adaptation, and models at run-time. Section 4 presents the three main research challenges that arise from raising the abstraction level of run-time models to the domain of business processes. These challenges are then individually discussed in the following three sections, each challenge comprising related work and first findings. We conclude with a summary and outlook in Section 8.

[1] These two relevant challenges have been extracted from a set of challenges for modern businesses identified by Simchi-Levi et al. in [52].

2 Background: Business Process Management

Processes accompany every human venture, from simply booking a holiday to manufacturing a car. In a similar way a business organisation is driven by its so-called "business processes": In order to achieve an organisation's objectives, tasks are usually carried out in certain ways, i.e. workflows are defined to express activities, the associated roles to perform them, and their order of execution. In [17] business processes are defined as *"...a series or network of value-added activities, performed by their relevant roles or collaborators, to purposefully achieve the common business goal."* Prominent examples of business processes are Order-to-Cash, Accounts Receivable, or Procure-to-Pay. Because of their central role in a business organisation they are considered to be *"...the most valuable corporate asset"* [1].

In order to deal with increasing complexity and respond to the arising importance of business processes, Information Technology (IT) was harnessed to manage business processes. This development lead to the rise of Business Process Management (BPM), as an IT-related discipline. In fact, BPM is a cross-discipline subject of "theory in practice" adopting a variety of paradigms and methodologies from computer science, management theory, philosophy, mathematics, and linguistic, just to name a few [17]. Perhaps because of its cross-disciplinary nature, even after a history of three decades, there are many duplicate, and contradictory publications trying to clarify definition and scope of basic BPM terminology [17], e.g. business process vs. workflow, BPM vs. Workflow Management (WfM) vs. Business Process Reengineering (BPR).

However, *Business Process Management* (BPM) is considered to be the next step after the workflow wave of the nineties [54]. Therefore, it is appropriate to use workflow terminology to define BPM. A *Workflow Management System* (WfMS) is defined as: *"A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications."* [19]. Based on that BPM is defined as follows: *"Supporting business processes using methods, techniques, and software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents, and other sources of information."* [54]. Software systems that support the management of operational business processes are called Business Process Management Systems or Business Process Management Suites (BPMS's) [18]. Although many other definitions of BPM exist, they are in most cases wrapped around *Workflow Management*(WfM).

In BPM a *process type* is a particular type of process with a defined business goal, e.g. Order-to-Cash. A process type is represented by a particular *process schema* which is captured in a *business process model* specifying business process aspects like activities, ordering, resources. A process type may be represented by more than one process schema expressing different versions or evolution steps of this type. Furthermore, a *process instance* is defined as a particular occurrence

of the business process, i.e. a particular sequence of executed activities in order to process a *work item*.

Part of the complete BPM definition is the BPM lifecycle. Here that of prominent BPM researcher van der Aalst et al. is adopted. It originates from the standard development life cycle and consists of 4 stages (see Figure 1) [54]:

1. **Process Design** - In this stage, business processes are modelled for the BPMS.
2. **System Configuration** - This stage configures the BPMS and the underlying system infrastructure (e.g., synchronisation of roles).
3. **Process Enactment** - The modelled business processes are deployed and executed in a BPMS.
4. **Diagnosis** - With analysis and monitoring tools, the BPM analyst can identify bottlenecks and improve the business processes.

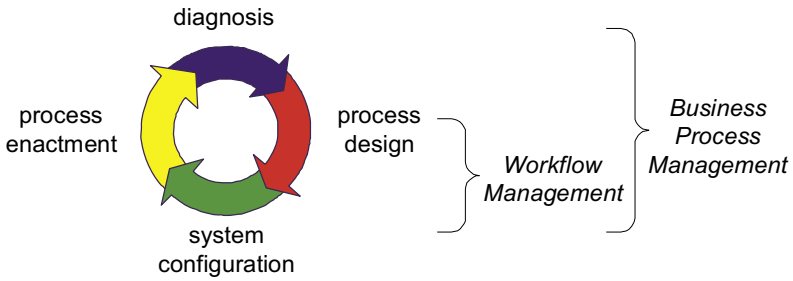

Fig. 1. BPM life cycle: Workflow Management vs. Business Process Management [54]

The viewpoint of Aalst et al. is that WfM covers only process design, system configuration, and process enactment, but BPM also includes the diagnosis phase to complete the BPM lifecycle [54]. This viewpoint makes WfM a logical subset of BPM. According to [17] *"...many BPMS are still very much workflow management systems (WfMS) and have not yet matured in the support of the BPM diagnosis."* However, recently the diagnosis phase started to gain more attention which is reflected in the high number of publications in the sub-topics of Business Process Analysis (BPA) and Business Activity Monitoring (BAM).

A more industry-based viewpoint on BPM and WfM is provided by Gartner [14]: *"Business process management (BPM) is a process-oriented management discipline. It is not a technology. Workflow is a flow management technology found in business process management suites (BPMSs) and other product categories."* Here BPM is a management discipline which is supported by WfM as a technology.

To put BPM terminology into one coherent picture, understanding the nested relationship of BPM theory (e.g., Pi Calculus [24] and Petri Nets [36]), BPM

standards (e.g., Business Process Model and Notation (BPMN) [31] or Business Process Execution Language (BPEL) [28]), and BPM systems (e.g. SAP Netweaver BPM [64] or Intalio BPMS Designer [15]) is of essence: BPM standards are based on established BPM theory and eventually adopted into software, i.e. BPMSs [18].

3 State of the Art

To address the topic of business process models at run-time and its associated general challenges (see Section 1), in the following sections state of the art is surveyed for the topics: (1) business process modelling standards, (2) business process adaptation, and (3) general models at run-time.

3.1 Business Process Modelling Standards

The previously mentioned challenges are part of a set of general challenges that are meant to be addressed by BPM standards. At the moment there are more than 10 formal groups creating BPM standards [66], many of them dedicated to definitions for business process modelling [13]. In order to get an overview about the state of the art for business process modelling standards it makes sense to categorise them into groups with similar functions and characteristics [18]. Many of the standards address at least one of the phases of the BPM life cycle. For this reason Ko et al. suggest a separation of features found in existing standards into four different types of standards [18]:

1. **Graphical Standards** allow users to express information flow, decision points, and roles for business processes in a diagrammatic way. Standards of this type correspond to the design phase of the BPM life cycle and are usually the easiest to understand, i.e. most human-readable. Prominent examples of graphical standards are Business Process Model and Notation (BPMN) [31], Event-driven Process Chains (EPC) [48], and activity diagrams of Unified Modelling Language (UML) [33].
2. **Execution Standards** are code-like and enable business processes to be deployed in a BPMS. Standards of this type correspond to the enactment phase of the BPM lifecycle. The most prominent example is Business Process Execution Language (BPEL) (sometimes also called Web Service Business Execution Language (WS-BPEL)) [28].
3. **Interchange Standards** are used to translate graphical standards to execution standards and exchange business process models between BPMS's [23]. One of the reasons these standards became necessary was the fragmented BPM landscape. Two prominent examples of interchange standards exist: Business Process Definition Metamodel (BPDM) [32] and XML Process Definition Language (XPDL) [65].
4. **Diagnosis Standards** provide monitoring capabilities. These standards are to support audit trails, real-time business process information, trend analysis, bottleneck identification, etc. Examples are initiatives of Object Management Group: Business Process Runtime Interface (BPRI) [30] and Business

Process Query Language (BPQL) [29]. Though, both of the projects failed to produce a standard (yet).

Most of the existing standards dealing with modelling languages can be assigned to one of these types. Of course, this a simplified view and there exist some exceptions which can be assigned to more than just one, e.g. Yet Another Workflow Language (YAWL) [57] can be regarded as graphical and execution standard, or BPEL which can have a graphical representation, too.

Many standards already exist (perhaps too many) which address specific phases of the BPM life cycle. However, important is the relation to the system with regards to their time of validity. In practice, two types of business process models with regards to their time of validity could be identified (see Figure 2):

- **A-priori model** - Business process models at design-time: In this case business processes are documented *before* execution to define the execution of workflows in an organisation. This is either done informally via a document listing and describing the steps and their execution order or they are modelled via design-time languages. The most prominent business process model languages were developed to build design-time models, and focus on aspects like interoperability, or being a basis for reliable communication between different stakeholders [8]. Basically, every language that addresses the enactment phase or one of the preceding is considered a-priori model, e.g. BPMN or BPEL.
- **A-posteriori model** - In practice business process models are often extracted *after* execution to reflect the real execution of a process as part of the diagnosis phase of the BPM life cycle. This static a-posteriori analysis of business processes based on event logs is called process mining [59] or in the case of a performance analysis during run-time Business Activity Monitoring (BAM). A-posteriori models in the sense of process mining usually conform to languages of BPM theory, e.g. Petri-Nets [56]. In the case of recent BAM solutions, special modelling languages that address run-time challenges of the diagnosis phase, e.g. need for notification when detecting alarming behaviour, are common. One example for such a modelling language is presented by Friedenstab et al. [10]: it proposes an extension for

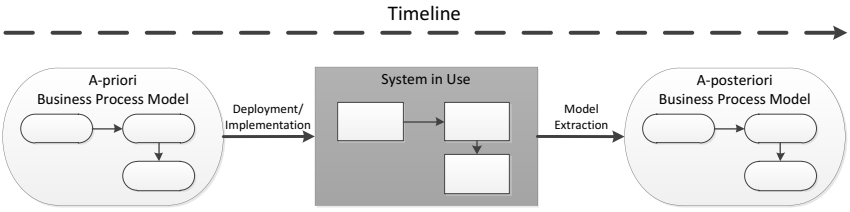

Fig. 2. Common Kinds of Business Process Models

BPMN to express process performance metrics. Both approaches, model extraction via BAM or via process mining, are considered run-time analyses in [59].

At the moment there is a shift towards a process diagnosis at run-time noticeable which is reflected in an increasing number of publications detailing approaches about how to make modelling of BAM more automated or even part of the business process modelling, e.g. [10,34,63,26,21]. In some cases even simple business process adaptation due to the monitoring results can be modelled. However, all of these approaches have limitations, one of which is that they are very much restricted to the purpose of traditional BAM: monitoring of Process Performance Indicators (PPIs) and Key Performance Indicators (KPIs) which are duration or frequency measures or aggregations of them.

3.2 Business Process Adaptation

Business processes need to be able to adapt to dynamic changes [44] in environment, e.g. because of a lack of available resources, or in flow of work, e.g. introduction of a new step. For instance, in domains like health care, Customer Relationship Management (CRM), or customised product manufacturing are process adaptations necessary or desirable to address changing demands.

In recent literature two different types of adaptation to dynamic changes could be identified: (1) *build-time flexibility*, i.e. the ability to pre-model flexible execution behaviour, and (2) *run-time flexibility*, i.e. in which an adaptation at run-time in the sense of exception handling or process evolution is carried out. In both cases the challenge is to balance flexibility and control [46].

Build-time flexibility is about leaving parts of the business process unspecified at design-time, i.e. the flexibility is modelled into the business process, and the missing information is added at run-time according to pre-specified constraints or rules. Different approaches to achieve this type of flexibility are by applying either general declarative processes [60,35], advanced modelling [53] or late-binding [46]. Pioneers of the more prominent latter approach are Sadiq et al. who introduced so called "pockets of flexibility" for workflow specifications [46]. The introduced workflow specification consists of [45]:

- core process consisting of pre-defined activities,
- pockets of flexibility within the process which in turn consist of
  - set of process elements, which can be a single activity or a sub-process,
  - set of constraints for concretising the pocket with a valid composition of process elements.

The definition is recursive and thus supports a hierarchical definition of flexibility pockets.

The other type of handling dynamic change: run-time flexibility, is about permanently or temporarily adapting the business process model at run-time. Permanent adaptation in the sense of process evolution is carried out by process

schema changes on the process type level and supported by adaptive process languages [40]. Temporary adaptations in the sense of ad-hoc changes is carried out on the process instance level and supported by exception- or case-handling [58].

For both types of flexibility, run-time and build-time, 18 change patterns[2] have been identified in [62] to facilitate formal validation for different adaptation approaches. The identified change patterns comprise a set of common process changes that could be applied to a business process. Though, all changes should be generally supported not all of them leave the business process after application in a valid state, e.g. removal of an activity can lead to a run-time error due to missing data. Hence, a number of changes is usually applied simultaneously, which emphasises the important challenge of change validation in the area of process adaptation, which is discussed in further detail in Section 6.2.

So far research in the area of process adaptation mostly focuses on the challenges of *how* adaptations can be carried out (modification policies) and *which* adaptations can be carried out (validation), but not so much on the challenge of *what* adaptation *should* be carried out (optimisation). The common constraint-based reasoning approaches with distinct adaptation solutions are limited with regards to the optimisation potential of business processes. A first practical approach which addresses automated process optimisation can be found in [49] where a business process optimisation loop including simulation as a mean for performance parameter computation and process adaptation is proposed. In this solution a simulation engine is included into the monitoring process with the help of which optimal solutions for a process change are determined. The business process is automatically adapted according to the suggestion. Although an evaluation has been carried out it seems that this work is still in a proof of concept stage as important definitions, e.g. for modification policies which are further discussed in Section 6.2, are missing.

3.3 Generalising Models at Run-Time

In Model-Driven Engineering (MDE), models are abstractions or reduced representations of a system. The combination of principles from MDE and reflective systems build the foundations of models@run.time. Here, models reflect the system's current status at *any* point in time as opposed to differentiate between a-priori and a-posteriori models. More specifically, a model at runtime (M@RT) *"... is a causal connected self-representation of the associated system that emphasises the structure, behaviour, or goals of the system from a problem space perspective"* [2]. Run-time models are used in different domains and serve different purposes, i.e. are problem oriented. Depending on the model's purpose is its properties. Still, similarities can be found that are more or less existent in most of the run-time models.

One approach of classifying model elements of M@RT is presented in [4] in which an analysis of model dynamics and executability has been carried out.

[2] 14 for run-time flexibility and 4 for build-time flexibility.

Therein the following classification of elements of *executable* run-time models has been identified:

- Definition part: the static part of the model which is defined at design-time
- Situation part: describing the dynamic state of a system during execution
- Execution part: specifying the transitions from one state to another

Because of the classification's focus on executable models it does not fully apply to *general* run-time models [20], i.e. not every run-time model is an executable model: E.g. run-time models with the purpose of monitoring do not necessarily have to have a definition part; some are built completely at run-time (e.g. by data mining algorithms). The inapplicability for general run-time models of this element classification motivated Lehmann et al. [20] to focus on classifying run-time model elements based on the causal connections of the model. The causal connections in a M@RT are either of a descriptive or prescriptive nature [51]:

- A model is descriptive if all statements made in the model are true for the System Under Study (SUS), i.e. every relevant change of the system is captured in the descriptive part of a run-time model.
- A specific SUS is considered valid relative to a prescriptive model if no statement in the model is false for the SUS, i.e. the space of possible system states is defined by the prescriptive part of a run-time model.

In general, the specification ratios of descriptive and prescriptive parts in a run-time model differ dependent on its purpose. That is, a M@RT that focuses, for instance, on monitoring has a strong focus on descriptive parts (e.g. [47]) and a M@RT that focuses on executability has a dominating prescriptive role (e.g. [27]). In addition to the prescriptive and descriptive parts of the model, Lehmann at. al identified that valid model modifications for both, descriptive and prescriptive, and the actual information flow of the causal connection are part of a general run-time model, too. The resulting classification to define elements of meta-models for general run-time models is the following [20]:

- prescriptive part - how the model should be
- descriptive part - state of the SUS at run-time
- valid modifications of descriptive part during run-time
- valid modifications of prescriptive part during run-time
- causal connections - modelling the information flow between the model and its SUS

The classification of elements for run-time models by Lehmann et al. [20] is shown in an example in Figure 3. Assuming there is only a finite number of states the system can be in then the prescriptive part would reflect all these states and the descriptive part would consist of the single state the system is in at the moment. The valid modifications of the descriptive part would determine the transition from one state to another, it represents the execution logic of the system. Additionally, through the notion of modifications of the prescriptive part the run-time model would be available from within the run-time model itself,

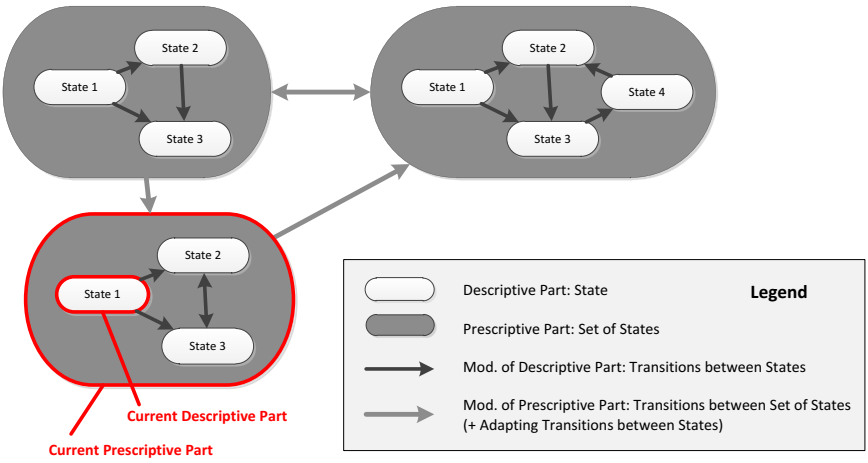

Fig. 3. Descriptive vs. Prescriptive Parts

i.e. be self-representative. Models that have the properties of self-representation and causal connection are called reflective [7]. However, per definition a run-time model does not necessarily has to have the property of self-representation, e.g. monitoring models are causally connected to the system sufficient for their purpose without having the ability to change the system. Plus, as pointed out earlier, the ratio of prescriptive and descriptive parts are dependent on the purpose of the model: For instance, prescriptive parts of a monitoring M@RT can be descriptive in a M@RT for dynamic adaptation.

However, there is one general issue that makes this classification only partly suitable for a general M@RT: The classification captures the self-representation property only partly because valid modifications for the descriptive parts should be able to change at run-time as well in order to support full self-representation. Assuming we are adding a state to the prescriptive part of the model, we would also have to define transitions describing how to reach this state (see Figure 3), i.e. add valid modifications of the descriptive part. We argue that the logical adaptation of the classification to overcome this issue is to declare the valid modifications of the descriptive part to be a part of the prescriptive part of the model. A good example of this fact are business processes models: They are generally prescriptive but also already define a workflow, i.e. the state transitions of the system.

3.4 Summary

As identified in Section 3.1 common business process model languages focus mostly on design-time aspects like interoperability, or being a basis for reliable communication between different stakeholders. For this reason important run-time aspects, like dimensions of change, are either only insufficiently supported

or not regarded at all. A simple example is that simple state of a process instance cannot be expressed with languages like BPMN.

If the concepts of M@RT are applied to the domain of business process models the result is business process models at run-time (BPM@RT). The current classification into a-priori and a-posteriori business process models does not support the purpose of BPM@RT because this type of model is neither provided *before* nor extracted *after* the system was in use. A BPM@RT is in fact a model causally connected *while* the system is in use and therefore represents a new type of model. Technically, a performance model derived from BAM is for instance already a BPM@RT, i.e. it is a process performance representation of a System Under Study (SUS) based on business processes and therefore to some extent emphases the goals and behaviour of the processes from the problem space perspective of performance analysis. However, there are more perspectives on business processes than performance analysis, e.g. path prediction and optimisation.

Also, a number of adaptation approaches presented in Section 3.2 can be considered as causally connected business process models at run-time, e.g. [40,46], as they capture the current state and/or allow for run-time adaptations. However, automated optimisation of business processes is by neither of the reviewed approaches supported and stays a current challenge that needs to be addressed. Only one very initial optimisation approach [49] could be identified in which several adaptation concerns have not yet been addressed.

We generally agree with the notion of business process models being handled at run-time to sufficiently address the need for adapting to changing demands and for shorter BPM life cycles. But the current state of the industry in which business process models are mostly regarded as either a-priori or a-posteriori models is too static and does not fully meet the requirements of systems in which business processes are highly volatile with possible changes over time. In some cases, however, business process models at run-time already exist, but do neither fully leverage model driven concepts nor support important problems like business process optimisation.

4 Research Challenges

Through the application of principles of the models@run.time discipline we certainly expect the view on business process management to become more structured and thus promotes a much needed separation of concerns. The assumption is that if the abstraction level of models@run.time can be raised to the domain of business processes, this can make business process management more automated and business processes more flexible and easier to adapt. Future research in this area will provide valuable contributions to areas of BPM and M@RT alike.

In an attempt to generalise future research challenges we identified topics that have to be further addressed and are subject of the remainder of this paper. The topics are conceptually depicted in Figure 4 and further described in the following list:

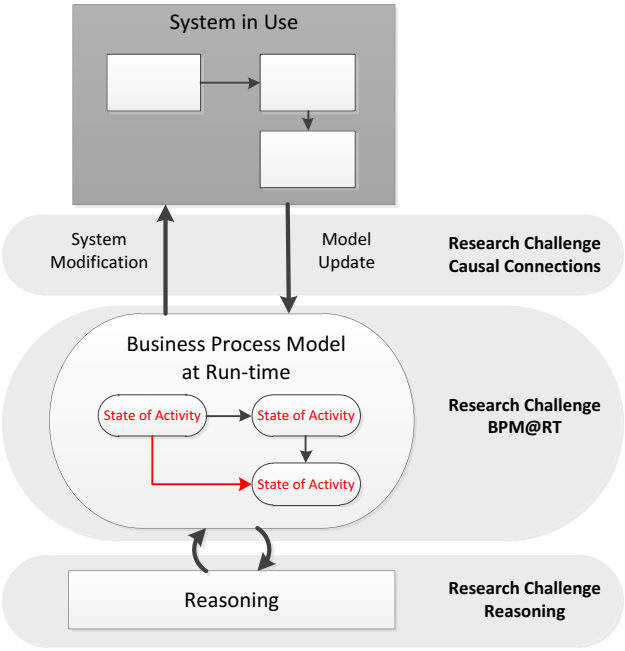

Fig. 4. Challenges for Run-time Business Process Models

1. **Run-Time Characteristics of BPM@RT:** First of all, BPM@RT has to deal with additional concerns as opposed to common business process models, such as capturing the current status information of the SUS's processes or the current performance of the process/system, depending on the problem space of the run-time model. Most business process models are a-posteriori models that only capture prescriptive information and this is why they need to be extended by certain run-time characteristics. One major challenge will therefore be to identify these run-time characteristics and elaborate a complete specification of a BPM@RT. A special emphasis has to be put on the dimensions of change the BPM@RT has to be able to express. The classifications of elements for general characteristics of run-time models reviewed in Section 3.3 is a good starting point to determine necessary parts for business process models at run-time. A review of related work and initial findings for this challenge are discussed in Section 5.
2. **Causal Connections between BPM@RT and the Associated System:** A second step towards BPM@RT is the identification of existing causal connections between the business process model and the SUS. Causal connections are either applied in the form of *Model Updates*, i.e. if the SUS changed the model has to be updated accordingly, or in the form of *System Modification*, i.e. the SUS has to be modified according to the associated model (see Figure 4). Due to the abstract nature of business process models this problem is particularly challenging as stronger requirements for the

causal connections between system and model are necessary, e.g. policies of how a BPMS can be updated during run-time. With regards to this challenge related work is reviewed in Section 6 and put into relation to the findings for the run-time characteristic challenge.

3. **Reasoning:** Models@run.time has been defined as a *"... causally connected self-representation of the associated system ... from a problem space perspective."* [2]. The problem space of business processes is diverse and dependent on what problem is to be regarded. Examples of this are: (1) Determining the current performance of a business process, (2) Predicting the future behaviour or performance of a current business process based on its current state and its historical behaviour and performance, and (3) Optimisation and adaption of a running business process according to given objectives and constraints. With respect to the actual problem in consideration appropriate reasoning methodologies have to be analysed and developed for BPM@RT. As opposed to common BPM reasoning methodologies like process mining and BAM, the reasoning will not be based on state change events but on the current state and historical states. As this is a change of paradigm which has, to the authors' best knowledge, not been addressed yet, this challenge can only be briefly discussed in Section 7.

After intensive literature review the authors claim that applying principles and theory of models@run.time to BPM has not yet been carried out to this extent. The expectation is to unify the BPM approaches towards the models@run.time paradigm, i.e. having a model express the current state and its history which is the basis of reasoning algorithms that can in turn change the model and eventually the system. Further research following this approach can initiate a shift from separate tools for modelling, execution, and diagnosis towards one framework comprising all of them. Already now the shift towards combining phases of the BPM life cycle are addressed by some approaches in industry (e.g., the existence of interchange standards to transform design models into execution models = design + enactment) and research (BAM solutions that can influence the business process execution = enactment + diagnosis).

Three different challenges towards BPM@RT have been identified: (1) Identifying characteristics for BPM@RT, (2) Identifying requirements for the causal connections between system and models, and (3) Reasoning upon BPM@RT. In the next three sections these challenges are individually discussed in further detail. That includes review of related work if available, first findings, and proposed next steps.

5 Research Challenge: Run-Time Characteristics of BPM@RT

A language for BPM@RT has to support specific run-time characteristics in order to deal with the requirements of a run-time model. The classifications of elements for general characteristics of run-time models, reviewed in Section 3.3, is

a good entry point to define requirements for such a language. As pointed out in that section, the ratio of prescriptive and descriptive parts are dependent on the purpose of the model, e.g. monitoring M@RT vs. execution M@RT. But not only the ratio of these parts can be different also the run-time aspect, prescriptive or descriptive, can vary for the same model element types depending on the purpose. That is, an element type, e.g. an activity, can be of a prescriptive nature in one BPM@RT, e.g. execution standards like BPEL, but of a descriptive nature in another BPM@RT, e.g. a run-time model extracted via process mining. Though, in both cases it is important that changes on the activity level can be captured. Hence, a special emphasis has to be put on the dimensions of change a BPM@RT has to be able to express. This is discussed in the remainder of this section, surveying related literature that deals with process flexibility.

5.1 State of the Art: Process Flexibility

An extensive taxonomy for dimensions of process flexibility is presented in [50]:

1. *Flexibility by design* is the ability to model alternative execution paths within the process definition at design-time. Dependent on the circumstances, the most appropriate execution path for a process instance can be chosen at run-time. This dimension is supported by almost any business process modelling language to some extent.
2. *Flexibility by deviation* is the ability for a process instance to deviate at run-time from the prescribed execution path of the business process model. The deviation does not allow for changes in the process definition, i.e. the business process model.
3. *Flexibility by underspecification* is the ability to execute an only partially defined business process at run-time. The full specification of the model is made at run-time and can be unique for each process instance.
4. *Flexibility by momentary change* is the ability to modify the execution of one or more selected process instances. This change is performed at the process instance level and does not affect any future instances.
5. *Flexibility by permanent change* is the ability to modify business process model at run-time such that the process definition is permanently modified. All currently executing process instances need to be transfered to the new process definition.

Whereas the first three dimensions leave the prescriptive part of the business process model unchanged, the last two encompass modifications in the prescriptive part of the business process model (either momentarily or permanently) at run-time. We can find that most of the flexibility dimensions of this taxonomy correspond to adaptation approaches presented in Section 3.2: Item 2 from the list above corresponds to exception handling approaches, item 3 corresponds to late-binding/pockets of flexibility, item 4 corresponds to case-handling, and item 5 corresponds to adaptive processes.

Another similar differentiation can be found in [46], in which dimensions of change for workflows are defined. Note, that the terminology in the following

approach is a little contradictory to the terminology used in the first approach. Some terms like "flexibility" and "change" have now a slightly different meaning. The classification of change dimensions for workflows is [46]:

1. *Flexibility* is the ability of the workflow process to execute on the basis of an incomplete specified model, where the full specification of the model is made at runtime. This dimension of change is the equivalent of *flexibility by underspecification* of the previous taxonomy.
2. *Adaptability* is the ability of the workflow processes to react to exceptional circumstances. These exceptional circumstances generally effect one or a few instances. This dimension of change is comparable to *flexibility of momentary change* or *flexibility of deviation* of the previous taxonomy dependent on if the process definition is momentarily adapted or not.
3. *Dynamism* is the ability of the workflow process to change when the business process evolves. This evolution may be slight as for process improvements, or drastic as for process innovation or process reengineering. Compared to the previous taxonomy this dimension is equivalent to *flexibility of permanent change*.

5.2 Identifying Run-Time Characteristics for BPM@RT

Both approaches capture dimensions of change that are either defined at design-time or at run-time. Of importance for the dimensions of change with regards to BPM@RT is, however, the associated abstraction level of the change, i.e. the granularity of a change. There are two abstraction levels of change that can be identified in both: (1) The change of the execution path of a process instance, in the remainder called *Variability*, and (2) the change of a complete business process definition, in the remainder called *Dynamism*. Due to the focus of both approaches on process change, one abstraction level of change has not been regarded, yet: the fine-granular state change in a process instance, in the remainder called *Reflectivity*. We argue that a language for BPM@RT needs to be able to support these three dimensions of change (see Figure 5) in order support any business process related purpose from business process monitoring to dynamic process optimisation of business processes.

The first conclusion to be drawn after identifying these three different dimensions of change is that some business process models already have the properties of adaptive models at run-time: Business process models that are executable and monitor the state of the system, e.g. certain workflow models like ADEPT [40], are adaptive models at run-time for *process instances*. That means in particular, that the prescriptive part specifies the possible states and transitions of one process instance, the descriptive part describes the current state in the process instance, and the valid modifications of the prescriptive parts are the shifts of execution paths for the process instance dependent on the circumstances. This is shown in Figure 6.

This is a good example to show how important the abstraction level of change is, i.e. in terms of dynamic adaptation: on what level do we capture change

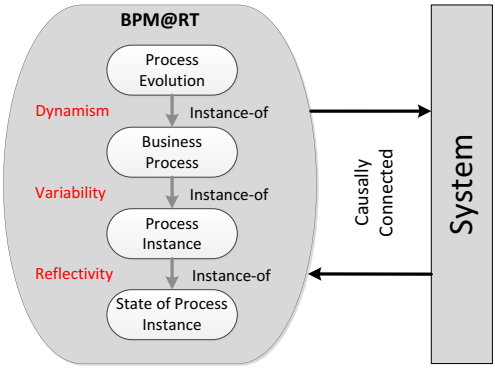

Fig. 5. Different Levels of State and State Change in BPM@RT

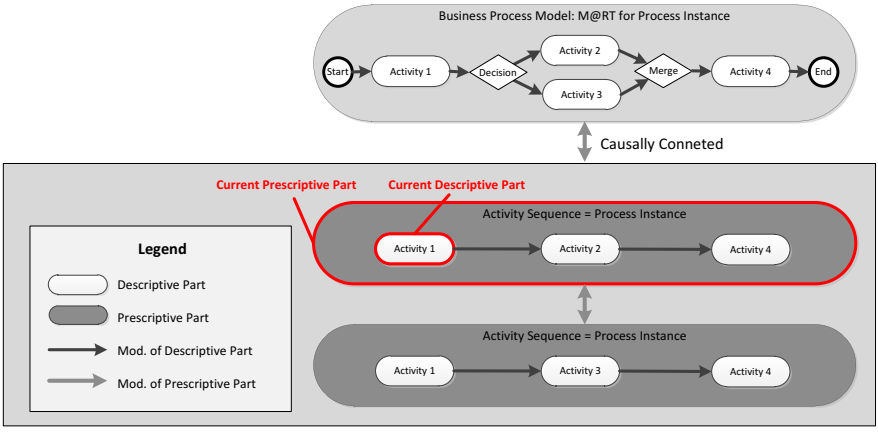

Fig. 6. Business Process Model as Process Instance Model at Run-time

of the system and on what level do we want to deploy change to the system. With regards to general BPM@RT, the requirements are to be able to capture and propagate change on the levels of reflectivity, flexibility, and dynamism. In addition to the desired change dimension of a BPM@RT the model also has to support standard business process modelling capabilities which is why the requirement of expressibility is essential, as well. A language to model BPM@RT has to support the requirements listed in Table 1.

Note, that some types of change cannot be assigned to one single level of change, e.g. the specification of language X might already allow to model a "Resource Change" but the specification of language Y does not support that notion. In that case, "Resource Change" is part of the variability dimension for X but part of the dynamism dimension for Y.

Table 1. Requirements for business process models at run-time

Requirement	Description
Expressibility	The expressive power of a process modelling language is governed by its ability to express specific process requirements reflecting the purpose of process modelling and execution. A process model is required to contain structure, data, execution, temporal, and transactional information of the business process [22][43].
Reflectivity	Reflectivity is the ability of the business process model to represent change in the system on the process instance level, i.e. the model should be able to reflect every fine-granular state the system can be in, e.g. state of the activity. This dimension is almost exclusively only triggered by the SUS and hence belongs to the descriptive part in most BPM@RTs.
Variability	Variability is the ability of a business process model to handle change on the business process level, i.e. it has the capabilities to model adaptations for process instances according to the desired behaviour, e.g. via a decision element, or according to exceptional but tolerated behaviour e.g. via exception handling. Depending on the purpose, changes of the variability level belong either to the descriptive or to the prescriptive part of the BPM@RT or to both.
Dynamism	Dynamism describes the ability of a business process model to be adapted at run-time according to changed circumstances. This business process evolution entails special challenges for the transition of process instances that have been initiated with the old business process generation but have not yet terminated. A Strategy has to be defined how these instances are migrated into the new process schema [50], which is discussed in Section 6.2. This dimension is almost exclusively used to change the currently executing business process model which in turn modifies the system in use and hence belongs to the prescriptive part of the BPM@RT.

As a next step towards BPM@RT we propose to check existing business process modelling languages like BPMN, BPEL, EPC, ADEPT, and YAWL against these requirements. Whereas most of them support the variability requirement to some extent, the other two dimensions of change, dynamism and reflectivity, are expected to be less supported. In case none of the existing solutions prove expressive enough an extension of the closest match or a new BPM@RT has to be specified. A formal validation of the resulting modelling language can be carried out based on general business process patterns [55] and business process change patterns [62].

6 Research Challenge: Causal Connections

In Section 3.3 we have distinguished between two different kinds of causal connections: (1) *model update* which alters the descriptive part of a model, and (2) system modification which has to be performed if the prescriptive part of

the model has been updated. In this section we take both causal connections under examination with regards to business process models and survey existing methodologies, respectively.

6.1 Model Update

The focus of this section lies on descriptive parts of a BPM@RT, i.e. the propagation of system changes to the model. This action is called model update. The basic task of a model update is to make sure that the current state from a problem space perspective of the SUS is reflected in the corresponding BPM@RT at any point in time. The assumption is that every single change in the SUS is represented as an event e_n which triggers a transition of an old $BPM@RT_{n-1}$ into an updated $BPM@RT_n$. This means a BPM@RT is built incrementally as conceptually shown in Op_1.

$$(Op_1) \quad e_n + BPM@RT_{n-1} \overset{ModelUpdate}{\rightarrow} BPM@RT_n$$

However, the common approach of extracting a-posteriori business process model information is called process mining and operates in a different way: The input is a complete event set $e_1, e_2, ..., e_n$ from which the business process model $BPModel_n$ is determined as shown in Op_2.

$$(Op_2) \quad (e_1, e_2, ..., e_n) \overset{ProcessMining}{\rightarrow} BPModel_n$$

The traditional and static process mining approach of Op_2 stands in contrast to the process model update approach and is not appropriately supporting the run-time characteristic of M@RT. This is why the process model update operation Op_1 has to be addressed by investigating suitable, incrementally operating algorithms for dynamic process mining.

In general a dynamic descriptive M@RT in the business process domain, e.g. process performance model, is causally connected with the BPMS through an event stream. Events indicating a change in the system are processed, aggregated and eventually trigger an update of the descriptive BPM@RT. This approach is called Business Activity Monitoring (BAM) (see Section 3.1) and is achieved through the application of Complex Event Processing (CEP) technologies. Existing BAM solutions mostly focus on monitoring key performance indicators on the business process level, e.g. [16,39,10]. As identified in the previous section, this is, however, only one abstraction level on which dynamic model updates can be triggered. With respect to the classification of abstraction levels of change, three different update types exist which are depicted in Figure 7 and described in the following list:

- **Dynamic Process Mining** is the discipline of updating model information on the business process level at run-time, i.e. detecting changes in the variability dimension. Many BAM solutions operate on that dimension of change, i.e. extract the performance of a business process model at run-time. It corresponds to the traditional a-posteriori process mining discipline

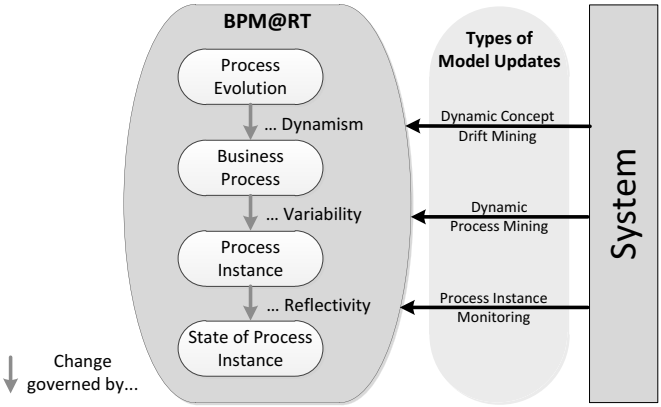

Fig. 7. Different types of model updates dependent on the level of change they are capturing

which is concerned with the extraction of knowledge about a business process based on its event logs [56]. Process mining approaches provide insight into a number of different perspectives: control-flow (called process discovered), performance, data, and organisation. Whereas BAM approaches address the performance perspective at run-time, development of solutions for dynamically mining knowledge about the other perspectives at run-time is, to the authors' best knowledge, still an open research challenge.

- **Process Instance Monitoring** is the discipline of updating model information on the process instance level at run-time, i.e. detecting changes in the reflectivity dimension. This represents capturing fine-granular atomic changes on the execution level, e.g. that an activity has been completed, and based on that updating the model to the current state of the instance. Some workflow and business process languages and their corresponding WfMS's and BPMS's already support the capturing and representation of that dimension of change at run-time, e.g. [40].
- **Dynamic Concept Drift Mining** is the discipline of updating model information on the process evolution level at run-time, i.e. detecting changes in the dynamism dimension. If the process gradually or suddenly evolves into another schema this has to be updated in the model. This corresponds to the traditional a-posteriori concept drift mining in processes [3], which to authors' best knowledge has not yet been approached in a dynamic way at run-time.

All three of these update types are event-based and should operate based on dynamic algorithms in the fashion of Op_1 as opposed to their static a-posteriori counter parts, i.e. concept drift mining and process mining. The common basis for model updates at run-time in the domain of BPM is through processing the

event stream which is a standard interface of modern BPMS, e.g. SAP Netweaver BPM [64]. However, this stream contains change events of the lowest possible dimension: the fine-granular state changes of the system. As there are no other generalised hooks available for changes of both other dimensions, dynamism and variability, have to be detected based on these low-level events.

6.2 System Modification

Since business process designers are not capable of anticipating all possible cases, exceptions, and events beforehand, the run-time system may not have sufficient knowledge to handle these situations and an adapted business process model might have to be redeployed. State of the art for business process adaptation with regards to existing approaches (build-time flexibility vs. run-time flexibility) has already been presented in Section 3.2. System modification is the action that has to be performed if the prescriptive part of the deployed business process model has been adapted at run-time.

Generally, business process models are abstract workflows where the abstraction level correlates to the type of causal connections between business process model and SUS, i.e. the higher the abstraction level of the business process model the more manual effort is potentially needed to execute an adaptation. In terms of the application of a system modification this means that with a high abstraction level it becomes more difficult to perform a system modification on the basis of the prescriptive part of the BPM@RT in an automated way. Common practice is that a graphical standard (e.g. BPMN) is used to design the business process model [change], then an interchange standard (e.g. BPDM) is utilised to transform that into an execution standard (e.g. BPEL) which is then executed and monitored. The actual modification of the system based on model adaptations is in the prominent BPMS not supported. Even though, there have been approaches to deal with adaptations for business processes as presented in Section 3.2, e.g. by build-time flexibility [25,60,35,53,46] or run-time flexibility [40,58], the actual system modification in an automated way remains to be generally very difficult to execute due to the high abstraction level of business processes.

One challenge that needs addressing to enable automation of system modification is the *validation* of the change that is to be applied to the system. In [40] a conceptual and operational framework is proposed that can reason about the correctness of a requested change to handle dynamic structural adaptations of workflows. At the core of this framework is a conceptual graphical workflow model (ADEPT) based upon which a complete and minimal set of change operations (ADEPT_{flex}) is defined, e.g. dynamic insertions/deletion of activities, or changing activity sequence. These operations allow for modifying the structure while preserving correctness and consistency of the system. With the help of formal constraints for state, flow of data, and flow of control, changes can be rejected if they can potentially lead to an invalid state of the system. This solution provides only a minimal set of changes with strict constraints to ensure that no invalid state can be reached. A more coarse-grained view on these changes

can help to reduce or relax these constraints, i.e. grouping changes instead of regarding every change as an atomic modification action. For instance, assuming two previously sequential activities are to become parallel, the constraints for this coarse-grained modification would be less strong than the constraints of the sub-modifications, deletion and parallel insertion, regarded individually. Weber et al. [62] identified 18 change patterns based on 157 real-life business processes from the domains of health care and automotive. 14 of these are adaptation patterns of different granularity, e.g. insert process fragment, delete process fragment, swap process fragments, parallelise activities, and embed process fragment in loop. The identified changes only consider the control-flow perspective and would have to be extended by patterns for the other perspectives, e.g. reallocation of resources.

However, if more complex changes, e.g. to split or parallelise activities, are requested *modification policies* have to be in place to ensure that the run-time system continues to operate in the expected manner. Modification policies specify how the transition from one business process to another is carried out [44]. These policies are important with respect to the still active process instances of the outdated business process and describe how to deal with them. Example policies are *Flush*, which allows all current instances to complete according to the old process model, *Abort*, which aborts all active process instances, and *Migrate*, which maps the state of active process instances to the new model. The last option is only applicable if additional migration constraints can be met, i.e. the migration into a valid instance is possible. Modification policies are discussed in more detail by Sadiq [44] and Schonenberg et al. [50].

In conclusion, due to the usually high abstraction level of business processes both causal connections, model update and system modification, pose difficult challenges. In the case of model updates for BPM@RT especially the dynamic update algorithms for the higher levels of change, dynamism and variability, are highlighted challenges for the future. In the case of system modification, determining patterns of change for different perspectives of business process models, e.g. resource and organisation perspective, will be a challenging task in the future.

7 Research Challenge: Reasoning on Run-Time Business Process Models

Reasoning is the action of drawing conclusions from available facts or statements. We understand reasoning as a discipline not only based on logic but also achieved by, for instance, statistical reasoning techniques, e.g. computation of key performance indicators. With regards to the actual problem in consideration appropriate reasoning methodologies can strongly vary in terms of input, applied techniques, and resulting output. In the remainder we summarise existing work in the domains of BPM and models@run.time with regards to these aspects and relate it to the concept presented in Section 4.

7.1 Input Information for Reasoning on BPM@RT

As opposed to traditional BPM reasoning methodologies like process mining and BAM, the reasoning in the proposed setup (see Figure 4) will not be based on *state change events* but on run-time business process models which capture the *current state* and *historical states* of the system. In the following listing both information types are described in further detail:

- *Current state information* comprises the descriptive parts of the BPM@RT representing the state of the BPMS on all three identified dimensions of change: reflectivity (i.e., the current state of an active process instance), variability (i.e., the current state of the business process, comprising all states of the active process instances), and dynamism (i.e., the current state of the process evolution, representing the current business process schema in use). Usually, at most two of the change dimensions are captured in current BPM@RT as they serve a specific purpose, e.g. BAM solutions capturing performance information on the reflectivity and/or variability level [16,38]. However, we propose to separate the concerns of capturing and reasoning: capturing the general state of the SUS on all three dimensions and apply the purpose-oriented reasoning based on this information.
- *Historic state information* comprises all past states the SUS has been in and their associated time spans. State changes happen with different frequency, ranging from a high frequency on the reflectivity dimension to a rather low frequency at the dynamism dimension. However, for elaborate reasoning techniques, e.g. simulation, it is a requirement to take the past states into consideration to achieve meaningful results. Hence, a general-purpose BPM@RT captures not only the current state on all three levels it also has a record of all the past states on these levels.

With these two types simple reasoning can already be applied, e.g. performance analysis, trend analysis, or path prediction. In terms of more elaborate reasoning additional *adaption information*, i.e. constraints, rules, and variants [9], which is usually defined at design time are necessary. These aspects would then belong to the prescriptive part of the BPM@RT and are dependent on which level of change the reasoning is considering. In our proposed setup this adaptation information is associated with the highest level of change abstraction: dynamism, i.e. changing the deployed business process models at run-time.

Note, that in literature for some analysis techniques (e.g., business process simulation [42,61]) an additional input data type is required: *design information*, which contains business process design information, e.g. control- and data-flow [42]. This type of information is in our point of view already captured in the current state information as all three change dimensions are to be captured, including the current business process schema (as a state).

7.2 Analysis Types for Reasoning on BPM@RT

In the following we discuss three analysis types that we consider important in terms of reasoning on BPM@RT: decision support, adaptation, and optimisation.

Decision Support. are analyses supporting the business analyst in making decisions about the business process through providing him with additional computed information of diverse nature, e.g. performance of the business process or involved resources. This information enables the analyst to obtain more insight into the process execution and its environment and react if an adaptation or exceptional interference becomes necessary, reallocation of resources. Examples of these analyses are *what-if analysis* [12], *performance monitoring* [16], *performance prediction* [38], *path prediction* [6], *sensitivity analysis* [11], and *bottleneck detection* [41]. Traditionally, in the BPM domain two basic types of analysis techniques are utilised to extract additional information from low-level data, i.e. event logs:

- *Analytical techniques* are based on mathematical methods and models to directly obtain information from the given data, e.g. FMC-QE [37]. Generally speaking, the biggest advantage is that instant results can be computed, which is why analytical techniques are preferably used in high-level analyses like optimisation were thousands of different cases have to be analysed as fast as possible. Disadvantages are that they typically are only simplified approximations (e.g., conditional loop behaviour hard to be represented by a formula [37]), impose additional constraints and are difficult to use [5].
- *Simulation* "... attempts to mimic real-life or hypothetical behaviour" [61]. It is considered to be versatile, impose only a few constraints, and produce results that similarly interpreted as the ones of the simulated system [61]. This is why simulation is one of the most established techniques in the domain of BPM supported by many tools. Most of these tools, however, focus on analysing rather abstract *steady-state* situations which are simplified models and less suitable run-time decision support [42]. To achieve more accurate results a *transient* analysis, where the current state is the starting point for an analysis is preferred [42]. This notion is fully supported by the BPM@RT approach. The biggest disadvantage of simulations is that they are time consuming and not very scalable: size of the business process, time to simulate, and average instance occurrence similarly have a linear influence on the execution time of the simulation. Additionally, as heuristic approach simulations even have to be executed several times to gain a certain confidence about the results.

Adaptation Reasoning. The challenge of reasoning is the connection between the descriptive and the prescriptive part of a M@RT and triggers possible system adaptations caused by an environment change. According to Fleurey et al. adaptation reasoning requires the following types of input [9]:

- *Context* which abstractly captures all the descriptive information, including current state and historical states. Traditional approaches however, only consider the current state to be important for an ad-hoc adaptation. Computed high-level information in the sense of the previously discussed decision support, e.g. performance information, can be part of the context and help determining the adaptation.

- *Variants* describe the flexibility of the run-time model or system, i.e. what adaptations are possible. Variants are of a prescriptive nature and belong to the adaptation information, introduced earlier. In the domain of BPM variants are, for instance, inserting a new activity, and reallocation of resources.
- *Constraints* specify restrictions on the variants and hence reduce the problem space. Constraints are extending the prescriptive part of a BPM@RT and also belong to the adaptation information. Examples for adaptation constraints can be state dependent, e.g. an activity can only be duplicated if it is not active at the moment, or state independent, e.g. an activity can only be allocated to a resource which can fulfill that role.
- *Rules* define how model and system should adapt to the change in the environment. These rules are in practice relations between the current state and the possible variants [9]. They extend the prescriptive part and belong to the adaptation information. One example is $\forall r \in Resources$: If $utilisation(r) > 0.8$ Then $multiplicity(r) \leftarrow multiplicity(r) + 1$.

The reasoning framework processes makes a decision based on the current context, variants, constraints, and rules at run-time. The output of the reasoning framework is an adaptation that matches the rules based on variants as well as context and satisfies the dependency constraints.

Optimisation. The reasoning based on rules and logic as proposed by [9] and introduced in the previous paragraph requires very good knowledge about the business process and about its possible adaptations. A more flexible approach is optimisation, an analysis which is driven by a fitness function. With the help of this function variants within the constraints can be rated and the one with the highest rating is considered to be the optimum. An optimisation is about finding the best solution for a given environment, i.e. technically it is not a subset of adaptation, but can be utilised to replace the adaptation reasoning via rules/logic. Alternatively, an optimisation function could be part of the rules but then all variants would have to be analysed. This is not suitable for a large number of variants. Well known optimisation techniques can be found in the areas of artificial intelligence, e.g. evolutionary/genetic algorithms, and mathematics, e.g. numerical algorithms. Note, that if an heuristic approach is utilised, a continuous swapping between localoptima is possible. This is a very undesired effect.

In BPM only one initial approach for optimisation is known by the authors which was discussed in Section 3.2. Here the future performance of every business process variant, which was computed via using simulation, represents the fitness function for the optimisation [49].

In conclusion, traditional reasoning in BPM is mostly based on the analysis of state transition events, especially in the very prominent area of decision support. With introducing the concepts of models@run.time a shift towards reasoning on current and historic states is motivated and has to be further investigated.

Adaptation reasoning is already well researched, its major limitation being the lack of applicability of current solutions in the industry, i.e. a challenge is how an adaptation can be modelled in an easier way. Additionally, in this section we did not distinguish between online or offline reasoning, i.e. static reasoning solutions have to be transformed if they are to be used at run-time.

8 Conclusion

Adapting to changing demands and shortening the business process lifecycle are prominent challenges in the domain of business process management. This paper motivates that a more dynamic handling of business processes is desirable, moving from design-time business process models to run-time business process models. We argue that a promising approach to address these challenges is provided by the community of models@run.time, in which causally connected models reflect the system's current state at any point in time and allow immediate reasoning and adaption mechanisms. This paper is a first attempt to raise the abstraction level of models@run.time to the domain of business processes, i.e. leveraging principles and concepts of the M@RT discipline to address the challenges of business adaptation and automation. With that it aims to unify BPM solutions towards a general models@run.time paradigm, i.e. having a model express the current state and its history which is the basis of reasoning algorithms that can in turn change the model and eventually the system. In order to generalise future research challenges three topics were highlighted that need further addressing:

1. Run-time characteristics of BPM@RT
2. Causal connections between BPM@RT and the associated system
3. Reasoning on BPM@RT

Each of these topics have been discussed in more detail individually, including review of related work, first findings, and proposed next steps. A number of resulting and more specific research challenges that need to be addressed have been identified and discussed: dimensions of change for BPM@RT, model update methodologies, modification types, modification policies, and business process optimisation. In the case of dimensions of change for BPM@RT, a first step has been taken by specifying the three different levels of business processes in which change can happen: dynamism, variability, and reflectivity.

Concluding, raising the abstraction level to the domain of BPM will provide contributions to the area of models@run.time generally, and for other M@RT at a similarly high abstraction level in particular. Furthermore, work in the area of BPM@RT will provide a valid use-case for M@RT and help to address the general challenges of business adaptation and automation.

References

1. von Ammon, R., Ertlmaier, T., Etzion, O., Kofman, A., Paulus, T.: Integrating Complex Events for Collaborating and Dynamically Changing Business Processes. In: Dan, A., Gittler, F., Toumani, F. (eds.) ICSOC/ServiceWave 2009. LNCS, vol. 6275, pp. 370–384. Springer, Heidelberg (2010)
2. Blair, G., Bencomo, N., France, R.B.: Models@run.time. Computer 42(10), 22–27 (2009)
3. Bose, R.P.J.C., van der Aalst, W.M.P., Žliobaitė, I.e., Pechenizkiy, M.: Handling concept drift in process mining. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 391–405. Springer, Heidelberg (2011)
4. Breton, B., Bézivin, J.: Towards an understanding of model executability. In: Proc. of the International Conference on Formal Ontology in Information Systems (2001)
5. Buzacott, J.: Commonalities in Reengineerd Business Processes: Models and Issues. Management Science 2(5), 768–782 (1996)
6. Cardoso, J., Lenic, M.: Web process and workflow path mining using the Multimethod approach. IJBIDM 2006 1(3), 304–328 (2006)
7. Cheng, B.H.C., et al.: Software Engineering for Self-Adaptive Systems: A Research Roadmap. In: Cheng, B.H.C., de Lemos, R., Giese, H., Inverardi, P., Magee, J. (eds.) Self-Adaptive Systems. LNCS, vol. 5525, pp. 1–26. Springer, Heidelberg (2009)
8. Dehnert, J., Van Der Aalst, W.: Bridging The Gap Between Business Models and Workflow Specifications. Int. J. Cooperative Inf. Syst. 13, 289–332 (2004)
9. Fleurey, F., Dehlen, V., Bencomo, N., Morin, B., Jézéquel, J.-M.: Modeling and validating dynamic adaptation. In: 3rd Int. Workshop on Models@run.time (2008)
10. Friedenstab, J.-P., Janiesch, C., Matzner, M., Müller, O.: Extending BPMN for Business Activity Monitoring. In: Proceedings of 45th Hawaii International International Conference on Systems Science, pp. 4158–4167. IEEE (2012)
11. Fritzsche, M.: PhD Thesis - Performance related Decision Support for Process Modelling. School of Electronics, Electrical Engineering and Computer Science, Queens University Belfast (2010)
12. Fritzsche, M., Johannes, J., Cech, S., Gilani, W.: MDPE Workbench - A Solution for Performance Related Decision Support. In: Proceedings of the Business Process Management Demonstration Track, vol. 489 (2009)
13. Ghalimi and D. McGoveran, D.: Standards and BPM. bpm.com (2005)
14. Hill, J.B., Pezzini, M., Natis, Y.V.: Findings: Confusion remains regarding BPM terminologies. ID no. G00155817. Gartner Research (2008)
15. Intalio. BPMS designer, `http://www.intalio.com/products/designer/` (accessed October 13, 2012)
16. Janiesch, C., Matzner, M., Müller, M., Vollmer, R., Becker, J.: Slipstream: architecture options for real-time process analytics. In: Chu, W., Wong, W., Palakal, M., Hung, C. (eds.) Proceedings of the 2011 ACM Symposium on Applied Computing (SAC). ACM (2011)
17. Ko, R.K.L.: A computer scientist's introductory guide to business process management (BPM). Crossroads Journal (2009)
18. Ko, R.K.L., Lee, S.S.G., Lee, E.W.: Business process management (BPM) standards: A survey. Business Process Management Journal 15(5), 744–791 (2009)
19. Lawrence, P.: Workflow Handbook 1997, Workflow Management Coalition. John Wiley and Sons, NY (1997)

20. Lehmann, G., Blumendorf, M., Trollmann, F., Albayrak, S.: Meta-modeling Runtime Models. In: MODELS Workshops 2010, pp. 209–223 (2010)
21. Liu, R., Nigam, A., Jeng, J., Shieh, C.-R.: Integrated Modeling of Performance Monitoring with Business Artifacts. In: Proceedings of 7th IEEE International Conference on e-Business Engineering (ICEBE), pp. 64–71 (2010)
22. Lu, R., Shazia, S.: A survey of comparative business process modeling approaches. In: Abramowicz, W. (ed.) BIS 2007. LNCS, vol. 4439, pp. 82–94. Springer, Heidelberg (2007)
23. Mendling, J., Neumann, G.: A Comparison of XML Interchange Formats for Business Process Modelling. In: Workflow Handbook (2005)
24. Milner, R.: Communicating and Mobile Systems: The Pi Calculus. Cambridge University Press (1999)
25. Modafferi, S., Mussi, E., Pernici, B.: SH-BPEL: A self-healing plug-in for Ws-BPEL engines. In: MW4SOC 2006, vol. 184, pp. 48–53 (2006)
26. Momm, C., Malec, R., Abeck, S.: Towards a Model-driven development of monitored processes. In: Proceedings of the 8th Internationale Tagung Wirtschaftsinformatik, pp. 319–336 (2007)
27. Muller, P., Fleurey, F., J'ez'equel, J.: Weaving executability into objectoriented meta-languages. In: Proc. of the 8th International Conference on Model-Driven Engineering Languages and Systems (2005)
28. OASIS: Web Services Business Process Execution Language Version 2.0 (2007), http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf
29. Object Management Group Inc: BPMN and Business Process Management - Introduction to the New Business Process Modeling Standard (2003), http://www.omg.org/bpmn/Documents/6AD5D16960.BPMN_and_BPM.pdf
30. Object Management Group Inc: The OMG Business Process Related Standards (2007), http://bpmfocus.pbworks.com/f/BPM+Standards+At+The+OMG+-+July+07.pdf
31. Object Management Group Inc: Business Process Model and Notation (BPMN) Specification 2.0 (2011), http://www.omg.org/spec/BPMN/2.0/PDF.formal/2011-01-03
32. Object Management Group Inc: Business Process Definition MetaModel - Volume I: Common Infrastructure (2008), http://www.omg.org/spec/BPDM/1.0./formal/2008-11-03
33. Object Management Group Inc: Unified Modeling Language 2.0: Superstructure (2005), http://www.omg.org/spec/UML/2.0/Superstructure/PDF.formal/05-07-04,
34. del-Río-Ortega, A., Resinas, M., Ruiz-Cortés, A.: Defining Process Performance Indicators: An Ontological Approach. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2010, Part I. LNCS, vol. 6426, pp. 555–572. Springer, Heidelberg (2010)
35. Pesic, M., Schonenberg, M.H., Sidorova, N., van der Aalst, W.M.P.: Constraint-Based Workflow Models: Change Made Easy. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 77–94. Springer, Heidelberg (2007)
36. Petri, C.A.: Kommunikation mit Automaten. PhD thesis. Rheinisch-Westfälisches Institut f. Instrumentelle Mathematik (1962)
37. Porzucek, T., Kluth, S., Fritzsche, M., Redlich, D.: Combination of a Discrete Event Simulation and an Analytical Performance Analysis through Model-Transformations. In: IEEE ECBS 2010, pp. 183–192 (2010)

38. Redlich, D., Gilani, W.: Event-Driven Process-Centric Performance Prediction via Simulation. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part I. LNBIP, vol. 99, pp. 473–478. Springer, Heidelberg (2012)
39. Redlich, D., Platz, S., Molka, T., Gilani, W., Winkler, U.: MDE in Practice: Process-centric Performance Prediction via Simulation in Real-time. In: Störrle, H., et al. (eds.) Joint Proceedings of co-located Events at the 8th European Conference on Modelling Foundations and Applications, pp. 336–339 (2012)
40. Reichert, M., Dadam, P.: ADEPT flex - Supporting Dynamic Changes of Workflows Without Loosing Control. Journal of Intelligent Information Systems, vol 10, 93–129 (1998)
41. Roser, C., Nakano, M., Tanaka, M.: A practical bottleneck detection method. In: Proceedings of the 33nd Conference on Winter Simulation, pp. 949–953 (2001)
42. Rozinat, A., Wynn, M.T., van der Aalst, W.M.P., ter Hofstede, A.H.M., Fidge, C.J.: Workflow Simulation for Operational Decision Support Using Design, Historic and State Information. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 196–211. Springer, Heidelberg (2008)
43. Sadiq, W., Orlowska, M.: On Capturing Process Requirements of Workflow Based Business Information System. In: Proceedings of 3rd International Conference on Business Information Systems (1999)
44. Sadiq, S.: Handling Dynamic Schema Change in Process Models. In: Proceedings of the Australasian Database Conference. IEEE (2000)
45. Sadiq, S., Orlowska, M., Sadiq, W.: Specification and validation of process constraints for flexible workflows. In Inf. Syst. Journal 30(5), 349–378 (2005)
46. Sadiq, S.K., Sadiq, W., Orlowska, M.E.: Pockets of flexibility in workflow specification. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 513–526. Springer, Heidelberg (2001)
47. Sanchez, M., Barrero, I., Villalobos, J., Deridder, D.: An execution platform for extensible runtime models. In: 3rd Int. Workshop on Models@run.time (2008)
48. Scheer, I.D.S.: ARIS (Architecture of integrated Information Systems) (1992)
49. Solomon, A., Litoiu, M., Lau, A.: Business Process Adaptation on a Tracked Simulation Model. In: ACM IBM Center for Advanced Studies Conference (2010)
50. Schonenberg, H., Mans, R., Russell, N., Mulyar, N., Van Der Aalst, W.: Towards a taxonomy of process flexibility (extended version). BPM Center Report BPM-07-11 (2007)
51. Seidewitz, E.: What models means. IEEE Software 20(5), 26–32 (2003)
52. Simchi-Levi, D., Simchi-Levi, E., Kaminsky, P.: Designing and managing the supply chain: Concepts, strategies, and cases. McGraw-Hill United-States (1999)
53. van der Aalst, W.M.P., Barros, A.P., ter Hofstede, A.H.M., Kiepuszewski, B.: Advanced workflow patterns. In: Scheuermann, P., Etzion, O. (eds.) CoopIS 2000. LNCS, vol. 1901, pp. 18–29. Springer, Heidelberg (2000)
54. van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M.: Business process management: A survey. In: van der Aalst, W.M.P., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 1–12. Springer, Heidelberg (2003)
55. Van Der Aalst, W., Ter Hofstede, A., Kiepuszewski, B., Barros, A.: WorkflowPatterns. Distributed and Parallel Databases 14(1), 5–51 (2003)
56. Van Der Aalst, W., Weijters, A.: Process mining: A research agenda. Comput. Ind. 53(3), 231–244 (2004)
57. Van Der Aalst, W., Ter Hofstede, A.: YAWL: Yet Another Workflow Language (2003)

58. Van Der Aalst, W., Weske, M., Grünbauer, D.: Case Handling: A New Paradigm for Business Process Support. Data and Knowledge Engineering 53(2), 129–162 (2005)
59. Van Der Aalst, W.: Trends in business process analysis - from verification to process mining. In: Cardoso, J., Cordeiro, J., Filipe, J. (eds.) ICEIS 2007 - Proceedings of the Ninth International Conference on Enterprise Information Systems, ICEIS 2007, pp. 5–9 (2007)
60. Van Der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - Research and Development 23(2), 99–113 (2009)
61. van der Aalst, W.M.P.: Business Process Simulation Revisited. In: Barjis, J. (ed.) EOMAS 2010. LNBIP, vol. 63, pp. 1–14. Springer, Heidelberg (2010)
62. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - Enhancing flexibility in process-aware information systems. Data and Knowledge Engineering, vol 66(3), 438–466 (2008)
63. Wetzstein, B., Ma, Z., Leymann, F.: Towards Measuring Key Performance Indicators of Semantic Business Processes. In: Abramowicz, W., Fensel, D. (eds.) BIS 2008. LNBIP, vol. 7(7), pp. 227–238. Springer, Heidelberg (1974)
64. Woods, D., Word, J.: SAP Netweaver for Dummies. Wiley, NJ (2004)
65. Workflow Management Coalition: XML Process Definition Language (XMDL) 2.2, `http://www.xpdl.org/` (accessed October 13, 2012)
66. Zur Muehlen, M.: Tutorial - Business process management standards. In: Proceedings of the 5th International Conference on Business Process Management (2007)