Noname manuscript No.
(will be inserted by the editor)

Integrating Business Process Simulation and Information System Simulation for Performance Prediction

Robert Heinrich · Philipp Merkle · Jörg Henss · Barbara Paech

Received: date / Accepted: date

Abstract Business process (BP) designs and enterprise information system (IS) designs are often not well aligned. Missing alignment may result in performance problems at run-time, such as large process execution time or overloaded IS resources. The complex interrelations between BPs and ISs are not adequately understood and considered in development so far. Simulation is a promising approach to predict performance of both, BP and IS designs. Based on prediction results, design alternatives can be compared and verified against requirements. Thus, BP and IS designs can be aligned to improve performance. In current simulation approaches, BP simulation and IS simulation are not adequately integrated. This results in limited prediction accuracy due to neglected interrelations between the BP and the IS in simulation. In this paper, we present the novel approach *Integrated Business IT Impact Simulation* (IntBIIS) to adequately reflect the mutual impact between BPs and ISs in simulation. Three types of mutual impact between BPs and ISs in terms of performance are specified. We discuss several solution alternatives to predict the impact of a BP on the performance of ISs and vice versa. It is argued that an integrated simulation of BPs and ISs is best suited to reflect their interrelations. We propose novel concepts for continuous modeling and integrated simulation. IntBIIS is implemented by extending the Palladio tool chain with BP simulation concepts. In a real-life case study with a BP and IS from practice, we validate the feasibility of IntBIIS and discuss the practicability of the corresponding tool support.

Robert Heinrich · Philipp Merkle · Jörg Henss
Karlsruhe Institute of Technology, Germany
E-mail: {heinrich,merkle,henss}(at)kit.edu
Barbara Paech
University of Heidelberg, Germany
E-mail: paech(at)informatik.uni-heidelberg.de

1 Introduction

Business processes (BPs) and enterprise information systems (ISs) mutually affect each other in non-trivial ways [1]. The complex interrelations between BPs and ISs, however, are not adequately researched so far. Especially interrelations between quality aspects (such as performance, reliability, or maintainability) concerned with business people and those concerned with IS developers are not well understood. Frequently, a direct mapping of metrics is not possible as the representation of a certain quality aspect may differ in the BP and IS domain. For example, reliability in the business context is often understood as fault tolerance and capability of fault handling (e.g., [16]). In the hardware context, reliability is typically represented in the form of mean-time-to-failure [43] of a hardware component (e.g., CPU or hard disk), where reliability in the software context is often described as failure probability of a certain system call in percent (e.g., [7]).

Engineering methods for aligning one domain to the quality objectives of another are missing. One major reason for insufficient quality engineering is that current approaches lack an integrated consideration of quality aspects among several domains. Frequently, BPs and ISs are not well aligned, meaning that BPs are designed without taking IS impact into account and vice versa [2, 9, 54]. Neglecting the mutual impact between BPs and ISs in the joint development leads to serious practical issues. On the one hand, it is not known whether a particular requirement can be satisfied by a proposed IS design, because it is not known how the system is used in the BP scenario, and how this usage affects the IS quality. On the other hand, it is unknown whether a particular requirement can be satisfied by a proposed BP design, because it is unknown whether involved ISs adequately

support the adherence of the requirement. Decisions in IS development are not reliably made since important BP-related information may not be considered. Insufficient consideration of BP properties may decelerate IS development due to rework needed in subsequent development phases. The same applies to neglected IS properties in BP development. A positive effect of the interrelations, however, is that new opportunities for BP evolution may come up due to novel capabilities provided by ISs [55,9].

Simulation is a powerful approach to predict the impact of a certain BP design on the quality of ISs and vice versa. BP design and IS design can be aligned by making adjustments based on the predicted quality impact. Performance is one of the quality aspects most addressed by current approaches. There are simulation approaches aiming at IS performance prediction (e.g., [4]) as well as approaches targeting BP performance prediction (e.g., [25]). However, current approaches do not adequately integrate both in simulation. Only few approaches in literature (cf. [36,13,5,23]) address the alignment of BPs and ISs. These approaches exchange information between isolated BP simulation and IS simulation. Simulating BPs and ISs in isolation is not an adequate approach as this neglects the mutual impact on workload burstiness [18]. Workload burstiness has "paramount importance for queuing prediction" [33] because it reflects whether load is dispersed equally or in bursts in a BP scenario. Thus, prediction accuracy of approaches using isolated simulations is limited.

In this paper, we present the novel approach *Integrated Business IT Impact Simulation* (IntBIIS) that adequately represents workload burstiness in performance simulation. IntBIIS enables the integrated simulation of BPs and ISs. Palladio [4] is an established approach to predict quality of software properties from software architecture models. The Palladio Component Model (PCM) [4] provides domain-specific modeling concepts and thus better supports modelers compared to traditional quality prediction formalisms, such as Petri nets and queuing networks. However, Palladio does not consider the business context of a software system. This work is an extension of the PCM and the event-based simulator EventSim [32] by BP-specific properties proposed in [19]. Parts of the content presented in this paper have been developed in the context of the first author's dissertation, which is published in [18].

Knowing that quality of BPs and ISs is a multi-dimensional concept (cf. [22]), we focus on performance in this paper due to the following reasons. Performance is one of the most demanded quality aspects across several domains, including BP [9] and IS [46]. In contrast to other quality aspects, foundations are available in form of established prediction methods and formalisms (e.g., [29,38,3]) that can be built upon. The performance of an existing BP and IS can be measured relatively easy (e.g., using monitoring techniques [50,11]), where other quality aspects, such as reliability, may require remarkably high number of observations, e.g., to gather events that happen very rarely [6]. Consequently, performance prediction methods can be validated quickly and easily, by comparing prediction results with measurements.

Van der Aalst et al. [49] distinguish operational decision making and design decision making. Operational decision making is conducted to solve a concrete problem at hand, e.g. by mobilizing additional resources [49]. In contrast, design decision making addresses fundamental and long-term modifications of structure or behavior, e.g. by introducing a new IS. This terminology is comparable to that used in software engineering where one distinguishes evolution and adaptation [17]. In the context of this paper, we focus on design decision making.

IntBIIS contributes to (a) the alignment of BP designs and IS designs while considering the mutual impact in between, (b) a more accurate performance prediction compared to isolated simulation, especially in cases of high workload burstiness, (c) the comparison of design alternatives and the verification of a design against requirements based on the predicted impact. Considering this, IntBIIS aims at reducing time and costs caused by rework in subsequent development phases.

IntBIIS supports several roles in the joint development of BPs and ISs: (i.) Requirement engineers can verify in the design phase whether an IS performance requirement can be satisfied by a proposed IS design for a given BP design. (ii.) IS designers can compare the performance of proposed design alternatives of ISs invoked in a given BP without implementing IS prototypes. (iii.) Hardware administrators can check the utilization of hardware resources such as a CPU or a hard disk drive for a proposed IS design or BP design. (iv.) Business analysts can verify in the design phase whether a BP performance requirement can be satisfied by a proposed BP design and a given IS design. (v.) Process designers can compare BP design alternatives without executing a BP in practice while the IS impact is included in the comparison.

The paper is structured as follows: In Sec. 2, we introduce definitions required to understand the paper. Sec. 3 introduces the order picking process. Three types of mutual impact between BPs and ISs in terms of performance are described in Sec. 4. In Sec. 5, we discuss related work. Several solution alternatives to represent the mutual impact in simulation are discussed in Sec. 6. In Sec. 7, we describe IntBIIS by introducing BP-specific

modeling constructs and simulation concepts. In Sec. 8, the feasibility of IntBIIS is validated and the practicability of our tool support is discussed.

2 Definitions

A *business process* is a "set of one or more linked activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships" [57]. Each *activity* within the BP is composed of a set of one or more linked steps. Steps are either performed completely by a human actor – called *actor steps* – or performed completely by an IS – called *system steps*. The representation of a BP or an IS in a model is called design hereafter. IS design refers to the meta-models in [4] while a BP design is described using the meta-models proposed in Sec. 7.1 through Sec. 7.3. System steps within a BP design refer to interfaces of software components in the IS design that implement services specified by the interfaces. In other words, system steps are system entry calls invoked by a human actor. They make the transition between BP and IS modeling by referring to IS-intern behavior [4].

Adapting the definition of *instance* in [57], we define a BP instance as the "representation of a single enactment" of a BP design. Similar to workloads in queuing networks [29], the BP workload specifies the intensity of process execution by determining the amount of BP instances that traverse the BP. Often workload is measured in BP instances per time unit. BP instances traverse all the actor steps and system steps on a certain path through the BP from the process starting point to a process end point. If the workload does not change over time, it is called *time-invariant*, otherwise it is called *time-variant*.

Several performance measures are applied in this paper. A short form of the definitions is given in the following, where a complete definition based on queuing network terminology is given in [18]. The total time required by a BP instance to traverse a system step is called *response time*. The total time required by a BP instance to traverse an actor step is named *execution time*. The time needed to traverse an activity within the BP or an entire BP is also termed execution time. These performance measures are predicted in simulation as described later in the paper. The performance measures are also applied to specify performance requirements on a BP design or an IS design. The requirements may be specified among others in the form of mean values, thresholds or intervals. Comparing the predicted performance measures to the requirements allows for determining whether a certain requirement is satisfied by a design or not.

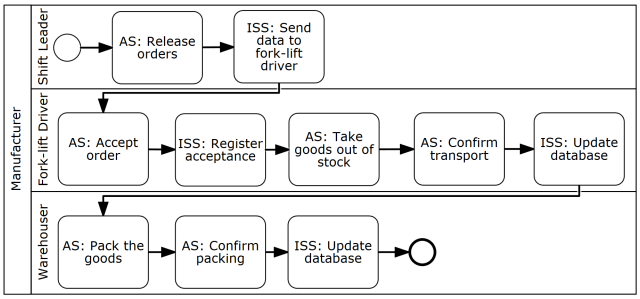

Fig. 1 The order picking process (simplified overview)

A set P of BP designs and a set S of IS designs are *aligned* in terms of performance, if

- each system step in a $P_i \in P$ refers to an interface of a $S_j \in S$
- each system step that refers to an interface of a S_j is contained in an element of P
- $\forall\ S_j \in S$: S_j meets its requirements
- $\forall\ P_i \in P$: P_i meets its requirements

At a particular point in time, each BP instance has its own *position* relative to the BP design, which represents its progress towards completion. A position is a model element in the BP design such as an actor step. The difference in time in which two BP instances reach a certain position is called *distance*. The distance in which two subsequent BP instances come into the process start position is named *inter-arrival time* (cf. [29]). *Workload burstiness* refers to the distance in which subsequent BP instances come into a specific position in the BP design. For example, three BP instances come into a specific position within a minute. The BP instances can come into the position at a constant distance (30 seconds) to each other, or they can occur in bursts. In both cases, the workload is three BP instances in one minute however the burstiness differs. A formal characterization of workload burstiness is given by the index of dispersion. Mi et al. [33] tailored the index of dispersion originally used in network analysis to IS requests. It is applicable to human actors alike as the ratio of the variance of the number of completed steps to the mean service rate.

3 The Order Picking Process

In this section, we introduce the order picking process and involved IS as a case of application from practice which is used for demonstration and validation purposes in this paper.

Fig. 1 shows a simplified representation of the order picking process at Thor GmbH, a multinational manufacturer and distributor of specialty chemicals. Parts of

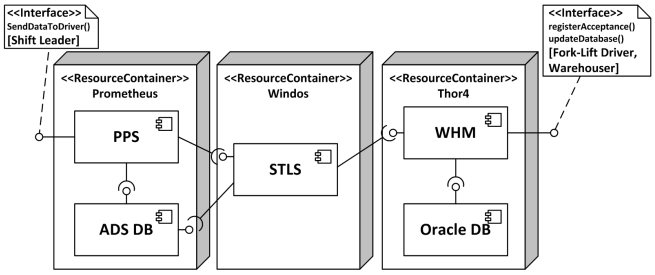

Fig. 2 IS on software component level (simplified overview)

the process description have been presented in early versions in [19,23]. A more detailed model is contained in the appendix of the first author's dissertation [18]. The simplified overview consists of a starting point and an end point, both represented by circles, and a sequence of steps in between. For a better understanding, activities that hierarchically nest the steps, loops, and path branches included in the process are not depicted in the figure. Steps are visualized by rectangles with rounded corners. "AS:" denotes actor steps. "ISS:" denotes system steps. Lanes represent roles of human actors. In the order picking process, goods requested by an order are taken out of the stock of the organization and are packed for transportation by truck. First, the shift leader releases orders for packing. The IS inserts the order data into a database (cf. Oracle database in the next paragraph) and transfers the order data from the database to a mobile client of the fork–lift driver. Then, the fork–lift driver accepts the order, which is registered in the database by the IS. The fork–lift driver takes the goods out of the stock and puts them on a location where they are packed for transport. Then, the fork–lift driver confirms the transport. The IS updates the database and informs the warehouser. The warehouser packs the goods for transport, takes them to a location where they are collected by a truck later, and confirms the transport. Finally, the IS updates the database.

Fig. 2 shows a simplified representation of the IS involved in the BP on software component level. Further details are given in [18]. Logical software components are represented as rectangles marked with a component symbol. The software components are deployed on several hardware nodes which are depicted as cuboid. Interface symbols indicate the source and target of call-dependencies between the components. PPS is a German abbreviation for the production planning and controlling component. ADS stands for Advantage Database Server. WHM is an abbreviation for Ware House Manager and STLS is a German abbreviation for the fork-lift control component.

The PPS component is used to present all order-related information to the shift leader. It is also used to trigger the release of new orders. PPS uses the ADS database component. The ADS database contains all information of an order such as ordered goods, quantities, and pricing. The WHM component communicates with the mobile clients located in each fork-lift. All movements of packing units are sent to the WHM component. The WHM component uses the Oracle database component to store all information related to the processing of the released orders such as storage places, movements of packing units, and the status of each order. When an order is released, the STLS component is responsible for the data exchange between the ADS database and the WHM. The STLS reads data from the ADS database and transfers it to the WHM component. Finally, the WHM inserts the order data into the Oracle database.

4 Mutual Performance Impact between BPs and ISs

In this section, we discuss the mutual impact between BPs and ISs regarding performance which represents the requirements on simulation approaches discussed in Sec. 5.

4.1 BP Impact on IS Performance

IS performance is affected by the BP design as well as by the BP workload. The BP design represents the usage profile of the IS at an abstract level. It determines which and when a specific system step is invoked, and which system steps are invoked concurrently. As the BP workload determines the number of BP instances traversing the actor steps and system steps in the BP, it also determines the workload of the IS. BP workload is performance-relevant since the performance of the IS may differ depending on its usage intensity [29].

4.2 IS Impact on BP Performance

Two kinds of IS impact on BP performance have been identified. First, permanently overloaded ISs impede BP execution. If one or more resources of an IS are permanent overloaded (e.g., by too many actor requests), the IS may no longer be available for actors in the BP. Second, the response time of ISs can significantly increase the process execution time. Frequently, IS response time is in a millisecond range. However, large database requests, complex calculations or data transmission to mobile systems may result in response times of several minutes. If the response time of the IS is in its extent comparable to the execution times of actor steps within the BP, IS response time may significantly affect BP performance. For example, in the order picking process, the transfer of order data to the mobile client of the

fork–lift driver lasts up to 40 minutes and more, which heavily impairs the process execution time as it extends accordingly.

4.3 Joint Impact on Workload Burstiness

The way human actors process jobs (i.e. actor steps) can be reflected by queuing network theory. Regarding this, the job processing by human actors is similar to those by hardware resources. However, there are specific differences (e.g., [49,56,34]) which are initially ignored in this section to focus on similarities, instead they are further discussed in Sec. 7.5. Human actors as well as hardware resources have a kind of waiting queue – called worklist for human actors [57]. Both resource types offer a certain service to their environment; jobs carry an amount of work to be done (the demand) and line up in a waiting queue when the resource is occupied. They process jobs from their waiting queue in a certain order, e.g. in FIFO or in a priority-based order. Human actors as well as hardware resources affect workload burstiness. Suppose the FCFS (first-come, first-served) scheduling principle. If an actor is already busy when an actor step has to be performed by this actor, the execution of the actor step is blocked until the actor is ready to perform the actor step. If a hardware resource used in a system step is already busy when it is invoked by an actor request, the request must wait until the resource is ready to process the request.

Moreover, also passive resources can cause waiting times. Passive resources in a BP are non–Information Technology devices or machines, such as a fork–lift. Passive resources in ISs, for example, are threads in a thread pool or database connections. They are available in a limited capacity and shared among all BP instances. If more passive resources are requested in the steps than currently available, the requesting BP instances have to wait until passive resources are released again.

Waiting times hinder the BP instances in traversing the BP design. For each step in the BP design, waiting times of BP instances may differ from one another depending on the waiting queue length of the corresponding resources. In a BP, it is common that several instances are processed concurrently by several actors of the same role whose waiting queues may differ. Frequently, hardware resources are also available in multiple replications, each of them having a different waiting queue length. Consequently, the distances between the BP instances in the BP design may vary during process execution. High workload burstiness often leads to increasing mean execution/response times (cf. [33]). Further discussion and demonstration on this is given in [23] and [18].

5 Related Work

Van der Aalst et al. [49] analyzed existing BP simulation tools and criticized the rather naive representation of the behavior of human actors. They identified limited support for modeling the organizational environment. This means that process models need to be complemented by information about resources [49] (e.g., human actors and their particular role). Thus, van der Aalst et al. concluded that existing BP simulation tools are not very useful for performance prediction. We share this criticism and address it by the modeling constructs and simulation strategies proposed in this paper.

The appearance of the BPMN [35] brought a variety of associated modeling and simulation tools with it [8]. We analyzed BPMN tools for quality modeling and proposed extensions in previous work [20,21]. The BPMN is appropriate to model human tasks and software service calls, and how they are embedded in BPs. The BPMN per specification, however, lacks the representation of service internals and their execution context, such as the software architecture and hardware resources. Hence, BPMN-associated simulation approaches are inadequate to reflect the mutual performance impact between BPs and ISs. Furthermore, what is called simulation in BPMN-associated tools is mostly limited to a sequence of steps in which one or more tokens pass through the process elements [8], however, does not take into account the utilization of resources, for example. Few simulation tools, such as ADONIS[1], come along with BPMN extensions by modeling and simulation concepts, i.a. to reflect human actors. The upcoming BPSim standard [53] is an extension of the BPMN that addresses limited simulation capabilities by offering diverse parameters (e.g., time parameters) that can be applied to predict BP performance. Nevertheless, also BPSim neglects IS internals and thus cannot adequately reflect interrelations between BPs and ISs. In contrast, we target a much more in-depth simulation of BPs and ISs building upon queuing network theory.

Furthermore, van der Aalst et al. [49] propose making use of event log analysis to gather data needed to create simulation models. More and more simulation tools are interconnected to event analysis techniques (e.g., via process mining [50] and system monitoring [11]). ProM [51] is an established process monitoring framework which is applied to provide inputs to BP simulation [39]. Kieker [10] is an IS monitoring framework that is currently extended [24,17] to provide inputs to the Palladio simulator [4]. Some techniques are applied to analyze the behavior of resources within the BP based on event logs recorded by ISs [47,34]. Event log analysis

[1] http://www.adonis-community.com/

to gather simulation inputs related to the IS and the BP is also a part of the simulation study conducted in the context of this paper. However, event logs are solely one possible data source. In practice, many information is not ascertainable from event logs (e.g., non-visible activities [56]) and must be gathered by other techniques, such as interviews. This is further described in Sec. 8.1.

There are few approaches in literature that address the mutual performance impact between BPs and ISs in simulation. Painter et al. [36] use BP simulation and computer network simulation in isolation in order to predict BP and IS performance. Giaglis et al. [13] present an approach to support concurrent engineering of BPs and ISs and to facilitate investment evaluation. BP simulation is used to predict BP performance. Computer network simulation is used to depict several alternative network architectures and topologies. Betz et al. [5] sketch a framework to integrate the lifecycles of BPs and business software for requirements coordination and impact analysis. Still, they use BP simulation and component-based software architecture simulation in isolation. In a prior publication [23], we present an approach to define interfaces between isolated BP and IS simulations for information exchange in order to predict the mutual impact. This is described in more detail in Sec. 6.1.

Although some of the approaches are not described in detail, it can be seen that all the approaches: (a) consider the BP impact on IS performance, as described in Sec. 4.1, and (b) IS performance is considered as a factor of BP performance, as described in Sec. 4.2.

The approaches by Serrano & den Hengst [44] as well as Tan & Takakuwa [48] only predict the impact of ISs on BP performance but do not consider the impact of BPs on IS performance.

All the approaches we found use BP simulation and IS simulation in isolation. Isolated simulations do not adequately reflect workload burstiness within the BP as described in Sec. 4.3. As workload burstiness may impact the performance significantly, the prediction accuracy of approaches using isolated simulations is limited. Mi et al. [33] showed how big the deviations can be. In an experiment, they observed bursts in workload of an IS. They compared the response time of a system step in the case of a random workload burstiness to the response time of the system step in the case that all the requests are compressed into a single large burst. In case of the burst, they observed that the mean response time is approximately 40 times longer than in random burstiness. The 95th percentile of the response times is nearly 80 times longer in bursts. Since human actors process jobs in a similar manner as hardware resources (cf. description in Sec. 4.3), we expect similar results for actor steps in the case of bursts. This leads to the conclusion that in case of bursts, one cannot expect accurate simulation results using existing approaches.

6 Discussion of Solution Alternatives

The coupling of multi-domain models for usage in simulation is a common problem found in many engineering disciplines. This coupling usually is done to analyze (mutual) interactions between modeled domains. A general overview on available techniques and frameworks is given in [37]. Simulation developers are challenged to find a meaningful mapping of entities and interactions to create an interoperable simulation composition. In our example, the system steps in the BP design have to match system entry calls of the IS design.

In the following, four solution alternatives for simulating the mutual impact between BPs and ISs are presented (see Fig. 3). These solutions can be classified by the heterogeneity (or homogeneity) on simulation model and infrastructure level. Moreover, we applied three criteria for comparing the solution alternatives:

C1 Workload burstiness: refers to the capability of the solution alternative to reflect the joint impact on workload burstiness.
C2 Modeling concepts: refers to relevant modeling concepts, such as time-variant workloads and suspendable resources, that have to be supported by the domain modeling languages.
C3 Realizability: refers to prerequisites that must be met to realize the solution alternative.

6.1 Isolated Simulations

The simplest solution to the interoperability problem is to run simulations in isolation and only exchange simulation results. In [23] we describe an approach where an IS usage profile describing the workload of the IS is derived from a BP model. The results of the IS simulation are written back as stochastic values to the BP model and used for consequent BP simulations.

Both simulations are conducted in isolation and information is exchanged ex-post. Thus, we have heterogeneity on both simulation model and infrastructure level.

6.2 Online Co-Simulations

Another approach is the usage of online co-simulation, where models remain in their specialized simulators

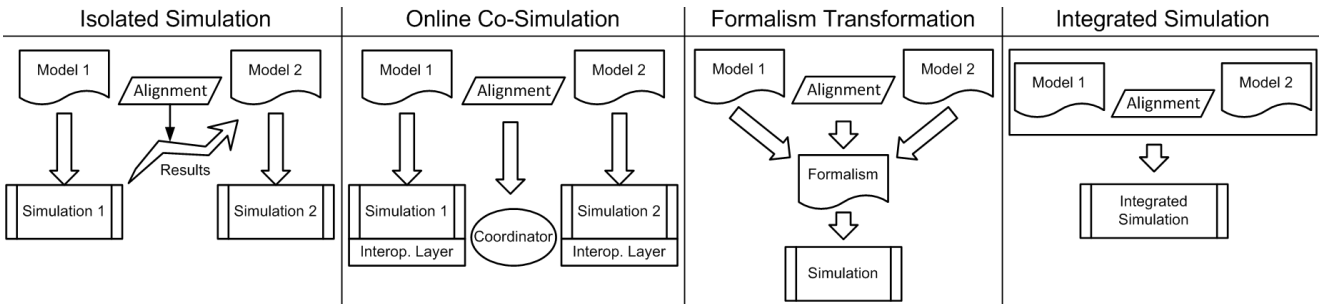

Fig. 3 Comparison of Solution Alternatives

which, however, are interlinked. This co-simulation commonly require additional efforts, e.g. a *Coordinator* for time management, model synchronization and connectivity in order to coherently integrate the simulations. With this kind of approach simulation models can remain heterogeneous, while at infrastructure level simulators have to assure technical interoperability.

The High-Level Architecture (HLA) [45] is a standard for federated co-simulations. It uses separately defined object models to describe shared objects along with possible interactions. These descriptions are domain specific and have to be created in coordination with the simulations. The HLA requires a central component (Run-Time-Infrastructure, RTI) which controls the simulation execution and manages the communication between simulations. Moreover, simulators commonly have to be adapted to interface and interact with the RTI and the shared object model.

6.3 Formalism Transformation

By formalism transformation, we understand the usage of model transformations for creating a homogeneous simulation model. A characteristic of this approach is that a single-formalism model is used as input to the simulation. Commonly, general purpose simulation formalisms like Petri nets or queuing networks are used as target formalisms [52]. The mapping of domain elements to the formalized simulation execution semantics is expressed in the used model transformations. Thus, the problems of technical interoperability can be neglected as only a single simulation infrastructure is used.

To employ this approach, simulation developers have to find a common formalism suitable for all simulated domains. Petri nets and queuing networks have a long tradition as formalisms to describe ISs as well as BPs. Moreover, there are transformations for translating IS models to the Layered Queueing Networks (LQNs) and Queueing Petri Nets (QPNs) [38,3] formalisms. Transformations and corresponding formalisms have been successfully applied, e.g., in [26,27]. There are also approaches to translate BP models to the LQN or QPN formalism, e.g., in [15].

Current transformation approaches, however, have strong restrictions on supported modeling concepts and have a lower prediction accuracy, compared to dedicated simulation tools [31].

6.4 Integrated Simulation

By integration, we understand that models of the distinct domains are combined to form a comprehensive model and are evaluated in a single simulator. In such a unified simulation framework the heterogeneity on simulation infrastructure level is eliminated. Thus, there is no need for external communication or additional time synchronization. In contrast to the priorly mentioned approach, heterogeneity on model level is preserved and no unified execution semantics has to be found. Still, the comprehensive simulation model has to include an alignment of used domains to enable interoperability.

An integrated simulation of IS and BP domain seems to be a promising approach, as there are several analogies: Both kinds of simulations usually built upon queuing theory concepts, use a specification of a sequence of actions to be processed by resources, and use hierarchical compositions of actions.

6.5 Comparison

When using the simulation in isolation approach, the impact of one simulation on the workload burstiness represented in the other is neglected. Consequently, this approach cannot satisfy criterion C1, resulting in reduced prediction accuracy. All other approaches support direct interaction on workload level and thus satisfy C1.

Considering criterion C2, both, LQNs and QPNs do not have support for modeling concepts, such as suspendable resources, and complex scheduling strategies related to them. To our knowledge, no support for time-variant workload is integrated in widespread QPN

and LQN simulators (e.g., [12,26]). Consequently, criterion C2 is not fulfilled, by the formalism transformation approach.

We encountered problems for the realizability of the formalism transformation and the co-simulation approach: No formalism transformation for BPs covering all required concepts and no co-simulation enabled simulators were readily available. Moreover, the existing BP simulation could not be retrofitted due to the closed source nature and, to our knowledge, available open source BP simulations do not cover required concept. Therefore, criterion C3 cannot be fulfilled by both approaches without major modifications to transformation and simulation approaches.

In the integrated simulation approach, a lot of the existing simulation infrastructure for the IS model can be reused or easily be adapted for the new BP elements, since actors and hardware resources often behave similarly, while processing jobs. We therefore consider this approach as realizable. One potential drawback of an integrated solution, however, is the lack of modularity and reusability of the simulation parts. Thus, although integrated, the simulations must be structured in a modular way, such that both the IS simulation and the BP simulation parts can be reused in other settings. That means that both simulations share a common basis for running and a set of glue elements for combining the domains (cf. Sec. 7). This allows us to use a shared single future event list for both simulations. A more loosely coupled variant, e.g. using remote interface communication, introduces the aforementioned drawbacks of requiring additional efforts for time synchronization and other communication overheads.

The discussion is summarized in Tab. 1. The cells represent whether a criterion is expected to be fulfilled or not (yes/no). The table clearly shows that an integrated simulation is the best alternative, since only the integrated simulation fulfills all three criteria.

Criterion/Alternative	Isolated Simulations	Co-Simulation	Transformation	Integrated Simulation
Workload burstiness	no	yes	yes	yes
Modeling concept	no	yes	no	yes
Realizability	yes	no	no	yes

Table 1 Comparison of Solution Alternatives

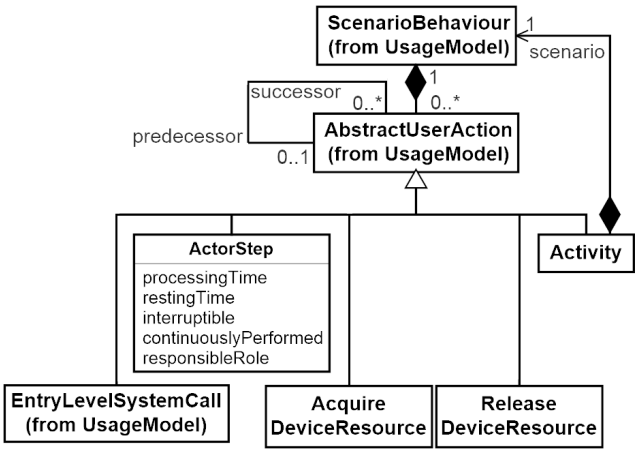

Fig. 4 Business Process Meta-Model (excerpt)

7 Integrated Simulation

In this section, we present the integrated simulation approach IntBIIS. We build upon the Palladio approach for realizing the integrated simulation of BPs and ISs. While Palladio already provides adequate means for modeling and simulation of IS designs, our work extends Palladio by modeling and simulation of BP designs. Extensions to the PCM meta-model are described in Sec. 7.1 through Sec. 7.3. An overview on how IntBIIS deals with BP properties in simulation is given in Sec. 7.4 through Sec. 7.6.

7.1 Business Process Model

The *business process model* represents a set of BPs, each basically described by a sequence of actor steps and system steps, as defined in Sec. 2. For each BP, the business process model also contains a workload specification. In this section, we focus on the behavior specification of BPs, while a discussion of workloads is deferred to Sec. 7.3.

An excerpt of the meta-model for business process models is depicted in Fig. 4. It extends the existing PCM usage model by BP-specific model elements. The proposed meta-model is compatible to established process modeling notations, such as the BPMN, in terms of actions and the control flow in between. However, we initially focus on the elements essential to describe the interrelations between BP and IS as our research targets the analysis of their mutual impact, not a complete support for exiting notations. Supporting existing BP notations is a topic of future work.

If meta-classes are taken from the original PCM, this is mentioned in parentheses in the figure. A business process model consists of one or more ScenarioBehaviours,

each representing a BP in our context. The behavior of a BP is specified by a sequence of AbstractUserActions, which are interconnected by a predecessor-successor relationship. All types of actions allowed in a regular PCM usage model can also be used within a business process model, including control flow elements such as loops and branches. For business process modeling, we introduced four additional actions: ActorStep, Activity, AcquireDeviceResource and ReleaseDeviceResource.

An ActorStep denotes a process step to be performed by a human actor in a specified role (responsibleRole). Each actor step requires a certain time to be processed by the assigned actor (processingTime) and can be followed by a timespan in which the corresponding BP instance rests (restingTime). For example, after mixing chemicals a certain waiting time might be needed before further processing. Otherwise, there is a risk of explosion. The interruptible attribute indicates whether the actor step may be interrupted. Interrupting an actor step is desirable in the following situations: (i) the assigned actor stops working, e.g., due to lunch break or to prevent working overtime, or (ii) the corresponding actor gets assigned an actor step with higher priority. Non-interruptible actor steps are given preferential treatment in that they always "overtake" interruptible steps queued for processing by the same resource and are not interrupted due to an imminent break. The corresponding scheduling policy is presented in more detail in Sec. 7.4. The continuouslyPerformed attribute indicates that a sequence of actor steps for which the attribute is set true, is performed by the same actor, if the same responsibleRole is allocated for all the actor steps.

A system step in the business process model is represented by an EntryLevelSystemCall which denotes a step to be performed by an IS, as described in [4].

An Activity serves as container for AbstractUserActions to allow for modeling hierarchically nested processes. The actions AcquireDeviceResource and ReleaseDeviceResource are used together to define a sequence of actions, that a particular device or machine is required.

7.2 Organization Environment Model

The *organization environment model* represents the organizational context of BPs in terms of resources involved in the BPs. Resources encompass human actors and their equipment – devices or machines, e.g., a fork–lift used by a warehouser. In this sense, an organization environment model is the counterpart of a PCM hardware environment model, which specifies available hardware resources, such as CPUs.

The meta-model for organization environment models can be seen in Fig. 5. ActorResources represent human

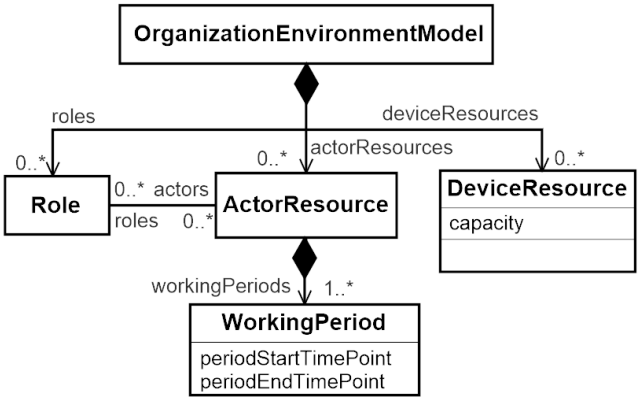

Fig. 5 Organization Environment Meta-Model

actors (hereafter also referred to as actors), each of which is assigned to one or more roles. A Role is an abstraction of concrete actors. It comprises several actors that have the same properties. In BP modeling, it is common that steps do not refer to actors directly but point to roles instead. Actors adhere to working hours determined by one or more WorkingPeriods. A working period is specified by a periodStartTimePoint and a periodEndTimePoint. For example, a workday split by a lunch break would be represented by two successive working periods – one before lunch, and one after. DeviceResource is a device or machine which is required to perform an actor step of the process but does not actively process the step. Thus, it is called a passive resource. The attribute capacity indicates the number of resource instances available to be shared among the BP instances.

7.3 Business Process Workload

The *business process workload* specifies the inter-arrival time of BP instances as a function of simulation time. It is part of the business process model, i.e., for each BP there is a corresponding workload specification. Workload specifications are also part of regular PCM usage models. These workloads, however, do not change over simulation time. This reflects Palladio's orientation towards steady-state analyses, where the focus is on predicting certain quality measures as simulation time approaches infinity. A simulation run is said to be in steady state with regard to a certain performance measure, if the performance measure becomes stable in the long run, i.e., if the underlying probability distribution does not change any longer when simulation time approaches infinity [28]. Changing workloads over simulation time could easily affect the stability of performance measures, which is one of the reasons why the original PCM kept workloads fixed.

Workload variations, however, are a fundamental property of BPs. Especially when studying how performance measures evolve over simulation time, it is vital to consider workloads as a function of time. Therefore, we extend the PCM usage model by time-variant workloads which may change over time according to their specification. Thereby, changed intensities of BP execution over the course of a day, a month, or even a year and more can be reflected. Fig. 6 exemplifies a varying workload over the course of a day and its effects on the queue length of an active resource. It can be seen how the workload variation leads to temporary overload conditions at peak load, which might be interesting to examine using simulation. As a side-effect from time-variant workloads, steady-state analyses of simulation results are often not viable as has been discussed above. Instead, analysis techniques known from terminating simulations have to be used (cf. [28]) to gather performance measures with sufficient statistical confidence. If the workload is specified in terms of repeating periods (e.g., when assuming that weeks do not differ in their workload pattern), a special case of steady-state analysis can be conducted, which involves so-called steady-state cycle parameters [28].

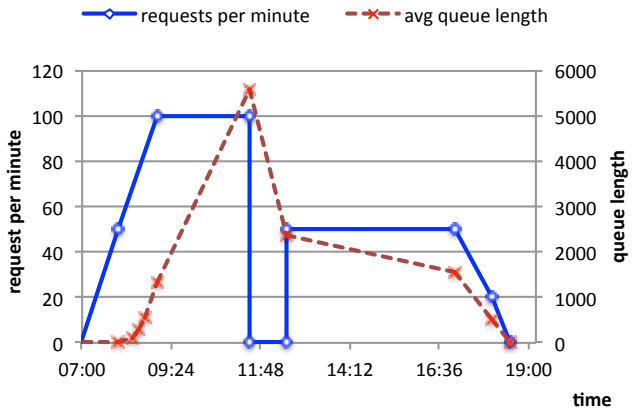

Fig. 6 Example: Time-Variant Workload on a Resource and Resulting Queue Length

The meta-classes for workload specifications are depicted in Fig. 7. The ProcessWorkload is an open workload whose intensity changes over simulation time, i.e., the inter-arrival time is specified as a function of time. For this, each ProcessWorkload may be decomposed into a sequence of non-overlapping, but not necessarily contiguous time periods (ProcessTriggerPeriod). Each period is specified by a start and an end point along with the inter-arrival time valid in between.

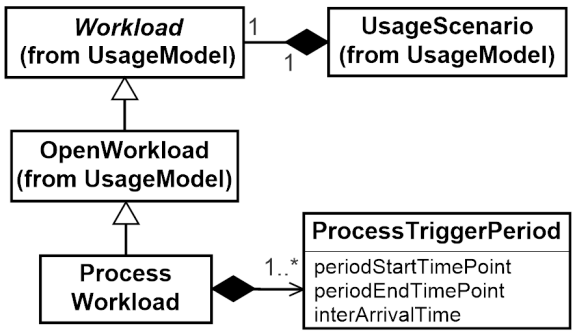

Fig. 7 Business Process Workload Meta-classes

7.4 Scheduling Policy for Human Actors

In simulation, it is feasible in particular situations to treat human actors as processing resources like a CPU, for example. However, there are specific differences, e.g. in working hours [56,49], priority [49] and interruptibility [56] of steps, effects of workload on processing speed [34,49], etc., that must be considered when reflecting human behavior in simulation. Hence, the scheduling policies commonly used with simulation of hardware resources do not meet our requirements for BP simulation. Specifically, first-come, first-served (FCFS) or processor sharing (PS) assume continuous operation without interruptions (due to holidays, lunch breaks, or non-working hours) and do not take into account preferential treatment of steps before others. For considering this in simulation, IntBIIS provides the opportunity to implement scheduling policies specialized for human actors. An example of a human actor scheduling policy is given in the following. Implementing further or extended scheduling policies to represent additional specifics of human actors is easily possible. The proposed scheduling policy supports suspension of resources and provides preferential treatment for non-interruptible actor steps. The latter involves two waiting queues per actor resource, one for interruptible actor steps (low priority) and one for non-interruptible steps (high priority). This allows for separated treatment of both kinds of steps instead of rearranging a single queue each time a higher prioritized step is allocated to the actor resource. In order to manage the suspension and resuming of an actor resource, it is described as a finite-state automaton by different states and transitions in between. The scheduling policy is defined by the following rules:

1. Non-interruptible (NI) actor steps have priority over interruptible (I) actors steps, meaning that no I-step is processed when there is an NI-step waiting to be processed.
2. Actor steps are processed in FIFO (first-in, first-out) order, as long as the abovementioned priority condition is not violated.

3. Whenever the actor is about to stop working (e.g., due to an imminent break), an I-step is immediately interrupted while an NI-step is still completed, even if this means working overtime.
4. If an NI-step is being enqueued while the actor is about to stop working, the newly arrived actor step is processed before the actor actually stops. I-steps, in contrast, do not delay the time until the actor stops.

7.5 Discussion on Human Actor Behavior

Now that we proposed modeling constructs and a scheduling policy for human actors, we discuss how they address open issues in resource modeling listed in literature. Van der Aalst et al. [49] identified an open issue regarding modeling the actors' availability when they are involved in multiple processes. They state that current simulators often focus on a single BP. We address this issue by the business process model (cf. Sec. 7.1), which allows for specifying several BPs, and by the scheduling policy for human actors, which allows for distributing an actor's workforce over various BPs based on priorities and workload, as demanded in [49]. *Non-visible activities* as mentioned by Wombacher & Iacob [56] refer to the same issue.

Another issue is that the performance of humans is affected by the workload they face [34,49]. Observations indicate that the relationship between performance and workload follows an inverse U-shaped curve. As our simulation determines the workload intensity per actor (by the length of waiting queues), all information is available to implement this observation in the scheduling policy and thus reflect it in simulation. Nevertheless, we neglect modeling variations in the processing rate of actors but assume a static rate instead. This is because in most countries, it is not allowed to determine the processing rate of a particular actor, due to regulations of law or the works councils (e.g., §87 BetrVG in Germany). Besides workload intensity, the processing speed of human actors in terms of performed activities is affected by a variety of influence factors [49,14], such as motivation, physical condition (e.g., the actor is tired or ill), or batch processing. An outlook is given in the future work section.

As demanded in literature [49,56], our organization environment model (cf. Sec. 7.2) allows for modeling working hours, part-time work, holidays, and lunch breaks of human actors and our scheduling policy allows considering this in simulation.

Van der Aalst et al. [49] point out the necessity of modeling priorities of tasks which is comparable to "preemption of activities" mentioned by Wombacher & Iacob [56]. Both is addressed by the proposed scheduling policy in simulation.

The last issue identified by van der Aalst et al. [49] is that BPs as well as the organization may change over time. This means that the process configurations and resource allocations may change flexibly depending on the context. As we focus on design decision making in this paper, supporting operational decision making is a topic of future work. For this purpose models must reflect the current state of process and organization. First ideas for reflecting the current IS state based on the Palladio approach have been proposed in [24]. These ideas may be extended for BPs and the organization building upon IntBIIS. However, design decisions have to be understood first. Then, the corresponding concepts can be applied to operational decision making.

7.6 Simulator Extensions

In order to simulate the mutual impact between BPs and ISs, we decided to extend the event-driven PCM simulator EventSim [32]. EventSim has been specifically developed for extensibility, which is especially reflected by the concept of traversal strategies. A traversal strategy encapsulates the simulation behavior for a specific type of action, an ActorStep for example. In this way the existing simulation semantics for PCM models can be easily modified or extended by registering an adapted or newly created traversal strategy with the simulator. Furthermore, EventSim has been shown to be faster and more resource-efficient in several scenarios compared to Palladio's reference simulator SimuCom [32]. Simulation speed is important in the BP context since long periods of time are simulated, which often span months or years.

Inspired by SimuCom, EventSim simulates the operation of a software system at different layers. This is illustrated in Fig. 8, where elements with a stickman indicate layers and elements introduced as a result of our work. The remaining layers and elements can also be found in Palladio's reference simulator. A run of the integrated simulation starts at the topmost layer with simulating time-variant business process workloads. For each workload specification, a workload generator spawns a new BP instance whenever the inter-arrival time has been passed. Each BP instance is then simulated individually by traversing the corresponding action chain specified in the BP model. When the traversal procedure arrives at an action, basically two cases can be distinguished: (i) the simulation encounters an actor step, or (ii) it encounters a system step (i.e., an EntryLevelSystemCall).

In case (i), a suitable human actor is requested (layer 5, left). When multiple actors are available, the actor

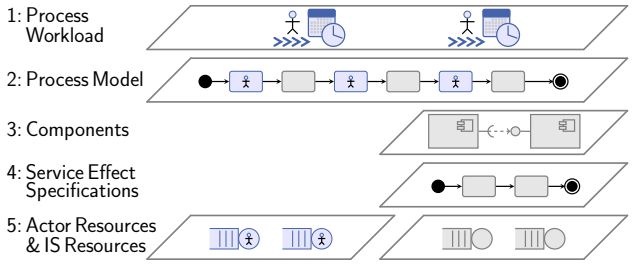

Fig. 8 Extended Simulation Layers

resource with the shortest waiting queue in terms of pending demand is selected. If the selected actor is already busy with another actor step or is temporarily suspended (e.g., due to a lunch break), the actor step is enqueued. This induces a waiting period not only for the actor step, but in particular also for the enclosing BP instance.

In case (ii), resource demands are not issued directly by the BP instance, but emerge as the system request propagates through components (layer 3), their service effect specifications (layer 4), down to hardware resources (layer 5, right). Like with actor resources, hardware resources may block a request, leading to a block of the enclosing BP instance.

The integrated simulation method IntBIIS[2] is capable of reflecting workload burstiness as has been requested in Sec. 4.3. Namely, the simulation considers the BP impact on IS waiting queues (of hardware resources) as well as the IS impact on BP waiting queues (of actor resources). In dependence upon the individual hardware resource utilization, each call to the IS causes the distance between concurrent BP instances to grow or to shrink. Likewise, each actor step affects the distance between concurrent BP instances depending on the individual actor resource utilization. Moreover, breaks of actor resources influence the distance. Exhausted passive resources have the same effect since they temporarily block BP instances or system requests until the requested amount of resource instances becomes available. As a consequence, occasional bursts may arise over simulation time leading to temporary overload conditions of both hardware resources as well as actor resources. Furthermore, occasional bursts may lead to situations in which passive resources are temporarily exhausted. Both, – overloaded processing resources and exhausted passive resources – (a) affect the performance of BPs and ISs, since they cause waiting times, due to blocked instances and increased queue lengths, and (b) again affect workload burstiness. This can be reflected by IntBIIS, due to the integrated simulation.

[2] IntBIIS is available online `http://sdqweb.ipd.kit.edu/wiki/IntBIIS`

8 Validation

Böhme & Reussner [6] introduce three types of validations for prediction methods. Type I (Feasibility) studies validate the accuracy of a prediction method by comparing prediction results to measurements from reality or results of another method. Subsequent to the description of data elicitation and simulation model construction in Sec. 8.1, we describe a feasibility study in Sec. 8.2 to validate whether IntBIIS yields accurate results under the assumption that its inputs were accurate. Type II (Practicability) studies validate the practicability of a method, when it is applied by the target users instead of the method developers. We discuss this in Sec. 8.3. Type III (Effort-Benefit) studies analyze the effort-benefit ratio of a prediction method by comparing the effort for conducting the same project at least twice. Once without using the prediction method, which may cause higher effort for rework, and once with using the prediction method, which may cause higher up-front effort. Type III validations are very seldom conducted due to (a) the high efforts required and (b) it is unlikely to convince an organization to conduct a project many times [6]. For this reasons, we did not conduct a Type III study.

The order picking process (see Sec. 3) is a comprehensive process from practice. Since an IS is involved in the process, it satisfies our requirements to analyze the mutual impact between BP and IS applying IntBIIS. On the one hand, simulation results can be compared to measurements (Type I). On the other hand, the maturity of the tool support and the interpretability of the results can be examined on a real-life example (Type II). For these reasons, the real-life validation is based on the order picking process, which was examined using IntBIIS and the approach in [23] for comparison. Using IntBIIS, we apply our Palladio extension which explicitly reflects the interrelations between the BP and the IS. For conducting [23] we chose the simulation tool ADONIS [25] and the original Palladio tooling. ADONIS enables the simulation of BPs and organizational resources, such as human actors, for business evaluation and process optimization. The ADONIS model takes into account only how the IS affects the BP, but not vice versa. Prior to the ADONIS simulation, the IS performance has been determined using the original Palladio tooling and annotated to the ADONIS model. The models are parameterized with data gathered in reality.

8.1 Data Elicitation and Model Construction

Before discussing the case study results, it is important to know how the simulation inputs and reference outputs were elicited in reality and how the simulation

models were created. For elicitation, we follow the guidelines of empirical research in software engineering by Runeson et al. [42]. Where possible, we use multiple sources for the same data (i.e. data source triangulation [42]) to increase precision and validity of the simulation inputs. Thereby, we closely involve process owners (i.e. BP experts) and technical staff (i.e. IS experts) of the organization in the data elicitation by conducting interviews. The experts were motivated to participate in the study as they recognized the problems with the status quo of the process in their everyday work.

The process model depicted as a simplified overview in Fig. 1 results from several interview sessions with BP and IS experts. Moreover, observations in reality and event log analysis have been conducted to elicit information required for model construction, e.g. path probabilities or number of loop iterations. In order to construct the IS model (depicted as a simplified overview in Fig. 2), we conducted reverse engineering of software components and interviews with IS experts. Performance-relevant data from reality, such as resource demands, frequencies and workloads, have been measured using event logs recorded by the IS. In order to check our measurements, the execution of several BP instances were observed and further interviews with BP and IS experts have been conducted. From the interviews and observations we gathered further performance-relevant data not available in the recorded event logs, for example the processing time of the single actor steps. The execution time distribution depicted in Fig. 9 has been measured over a period of more than six weeks by event log analysis which results in logs containing more than 1 200 observed BP instances. The distribution represents per BP instance the time required from the process start position to the process end position. Besides process start and end, we recorded various other events to gather aforementioned information, such as path probabilities or interaction with and workloads of the IS, which results in logs containing up to about 74 000 events.

After the entities had been elicited, we created the simulation models as described hereafter. It is important to note that the parameters of IntBIIS meta-models can be found in a comparable form in ADONIS meta-models, e.g. inter-arrival rate, step durations, etc. For corresponding parameters in the meta-models, we made the same settings. For specifying the IS, we used the original Palladio meta-models in both simulation studies.

Workload burstiness at the process start position must be mapped to the model. The BP instances start executing the BP in several bursts. Between the bursts, there are long time frames in which no BP instances start executing the process. Owing to the long time frames between the bursts, a mean distance between all the BP instance start time points over the whole day would not adequately reflect the workload burstiness. Therefore, we recorded an exemplary day. We decomposed the day into several process trigger periods in which the distance between the BP instance start time points is about the same. Each period indicates a burst. Initially, we determined the periods' inter-arrival times by the mean distance between the BP instance start times points in the corresponding period. Per period, the start time point, the end time point, and the inter-arrival time is specified in the process model. As mentioned above, we verified the simulation inputs whenever possible. For verifying the period specification in the models, we conducted a preliminary simulation run and compared the number of BP instances in the run to the number of BP instances observed in the same time span in reality. We identified that the number was too low in the preliminary run. Therefore, we adapted the period specifications (in both models, ADONIS and IntBIIS) to approach the workload in simulation to the workload observed in reality. We varied the period properties (start, end, interarrival) to ensured adequate simulation inputs.

Hardware resources as well as human actors execute several BPs concurrent to the order picking process, such as loading and shipping. They are demanded by other BPs which causes waiting time in the order picking process. However, we were not able to model all concurrent BPs and include them in simulation. There would be high additional effort required to include the concurrent BPs in the study. Since this effort would exceed the scope agreed with the organization, we decided to create default resource utilizations which estimate the load induced by concurrent BPs. This is a common procedure in performance prediction. In order to model the IS default utilization, we measured the utilization of hardware resources used in the order picking process. We added hardware resource demands to a concurrent BP model until the simulated resource utilization approached the measured resource utilization. For human actors we added several actor steps to a concurrent BP model. Actor steps were added until the minimum and maximum value of the execution time distribution of the order picking process in simulation approached the minimum and maximum value of the distribution measured in reality. We proceed this way because we could not measure the default actor resource utilization.

Each working day is divided into three shifts (early shift, late shift and night shift), but only one shift – the late shift – was recorded in detail. The late shift was chosen because mainly the orders are packed in this shift.

This was the result of interviews with experts of the organization and a six-week observation of the process in reality. In consequence, the models and corresponding simulation runs also consider only the late shift. BP instances whose steps cannot be completely processed in a single shift, are carried over to the next shift. In the next shift, processing is continued. Since the early shift has not been modeled, there is no way for the simulation to know the amount of work carried over to the late shift. To mitigate this problem, we approximate the carryover to the late shift with the carryover from late shift to night shift of the previous day. This is a worst-case approximation of the carryover since most orders are packed in the late shift. The carryover from one shift to another, however, is quite small. Even in the late shift, the carryover is about one of more than thirty BP instance handled per late shift on average. Thus, the carry over does not significantly affect the simulation results.

In reality, we did not observe significant differences in execution time among recorded workdays. The simulation results reflect this observation, which is why simulation of longer periods can be considered as a sequence of replicated simulation runs. With each further simulated day, we receive an additional replication leading to an improved confidence in the simulation results. With this technique, we ensured a sound basis for statistical comparisons.

8.2 Feasibility

The mutual impact between BPs and ISs affect hardware resources as well as actor resources. We assume that Palladio correctly simulates the behavior of hardware resources as demonstrated in several case studies (such as [4,30]). The mutual impact is correctly represented by the integrated simulation, if also the actor resources' behavior is correctly reflected.

8.2.1 Real-life Validation

In order to validate whether the actor resources' behavior is correctly simulated, we compare results of IntBIIS to values measured in reality. Mapping human actor behavior to a model is hard. Thus, we decided to additionally compare the results of IntBIIS to the results of another model-based simulation approach to exclude influencing factors caused by modeling. The approach in [23] was applied to specify interfaces between existing simulation tools to provide an alternative way of predicting the mutual impact between BP and IS. The simulations are conducted in isolation and information is exchanged via the interfaces ex-post. In the following, IntBIIS denotes the results of the integrated simulation whereas ADONIS denotes the results of the isolated simulation [23].

IntBIIS predicted a mean process execution time of 5 409 sec. (about 1h 30min). The ADONIS simulation predicted a mean value of 4 954 sec. (about 1h 23min). In reality we measured a mean process execution time of 5 326 sec. (about 1h 29min). The distributions of the predicted and measured process execution time are depicted in Fig. 9. The left side shows the estimated probability density, where the right side depicts the corresponding cumulative density.

It is seen in the figure that the curves follow a similar trend. Compared to the measured curve, the simulated curves show higher peaks but less variance in execution time, which is reflected by the width of the curves. Deviation between the curves can be explained by the following three reasons. (i) It is hard to map the behavior of human actors to a model and reflect it in simulation because, in reality, actors do not always behave in exactly the same manner. Actors do not always start processing the next step directly after the former has been finished. Sometimes they have to take a break within a working period (e.g., to go to the toilet) or have a conversation. This causes variability in BP performance. (ii) The processing rate of actors may differ from one actor to the other in reality. Even the processing rate of a certain actor may vary, as discussed in Sec. 7.5. (iii) Simulation inputs might have limited accuracy. The order picking process is a real-life example. Thus, it is hard to gather accurate data. For example, data gathered in an interview or an observation typically has a certain deviation to reality. This is why we used measurements from event logs recorded by the IS where possible. Fig. 9 also shows some deviations between the ADONIS curve and the IntBIIS curve. Deviations may be caused by differences in the simulation strategies of both methods and the mutual impact on workload burstiness, which is considered in IntBIIS but not in ADONIS.

At a first glance, both simulation methods yield reasonable accuracy, where the mean value predicted by IntBIIS better fits measurements than the one predicted by ADONIS. It is hard to judge visually which of the simulation methods performs better in the validation scenario. This is why we applied a distance measure to make the mutual differences tangible. A comparison of distribution-based similarity metrics in [40] confirms the earth mover's distance (EMD) [41] as the most appropriate one. Moreover, EMD operates in an intuitive manner and takes into account differences both in shape and location of probability distributions. For these reasons, we choose EMD as a distance measure. Given two

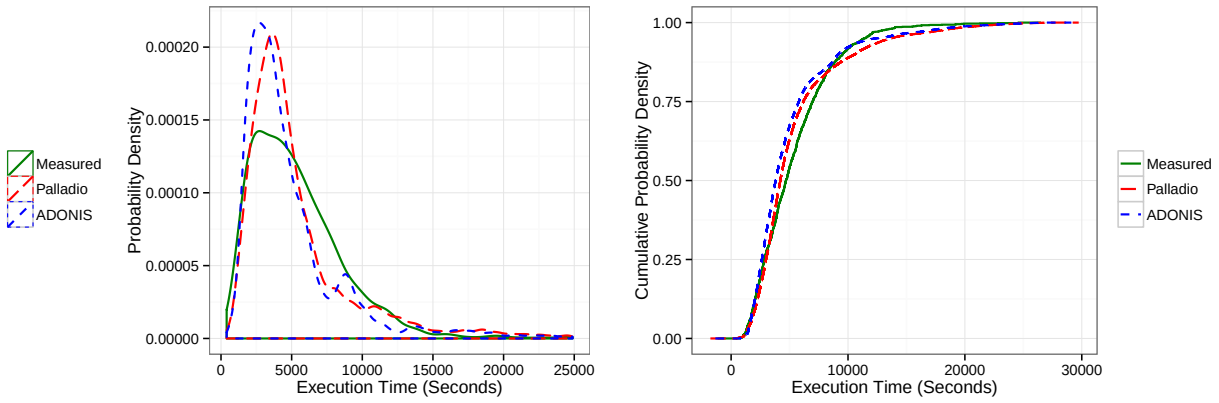

Fig. 9 Comparison: IntBIIS, ADONIS and Reality

probability distributions, the EMD algorithm calculates their distance in terms of the effort that must be undertaken to transform one distribution into the other. Figuratively speaking, a distribution's probability mass is moved to the distribution under comparison until they are aligned [41] – the amount and distance of mass transported yields the EMD metric.

IntBIIS vs. Measured:	559.43
ADONIS vs. Measured:	696.66
IntBIIS vs. ADONIS:	460.05

Table 2 Earth mover's distance between predictions and measurements

Execution Time (Seconds)	Probability (IntBIIS)	Probability (ADONIS)	Probability (Measured)
0–5 000	0.63	0.67	0.54
5 000–10 000	0.26	0.25	0.38
10 000–15 000	0.07	0.04	0.07
15 000–20 000	0.03	0.02	0.01
20 000–25 000	0.01	0.01	0.00
25 000–30 000	0.00	0.00	0.00

Table 3 Execution time (probabilities rounded)

Tab. 2 presents the mutual differences. Although reported values are dimensionless, they meet the definition of a metric and thus are suited to give an impression on relative distances. With a distance of 559 and 697 for IntBIIS and ADONIS, respectively, IntBIIS resembles the measured distribution better than ADONIS. The distance of 460 between IntBIIS and ADONIS is comparatively small, which reflects the aforementioned observation that both curves follow a similar trend. A comparison of the probabilities for various ranges of execution time leads to a similar conclusion (cf. Tab. 3). Except for the range 15000 to 20000, compared to ADONIS, IntBIIS yields for each range a probability closer or equal to the probability calculated from real-world measurements. This observation is also supported by the three quartiles Q1, Q2, and Q3 shown in Tab. 4. For all quartiles, the predictions with IntBIIS are closer to the measurements than ADONIS. In summary, these findings confirm the feasibility of IntBIIS. Compared to ADONIS, IntBIIS seems to have a higher prediction accuracy as the mean values, distances, probabilities, and quartiles consistently indicate.

Moreover, IntBIIS reduces effort compared to isolated simulation of BPs and ISs as described in [23]. Using isolated simulations, (a) an IS usage profile has to be derived from the BP model for IS simulation and (b) the BP model must be extended by IS simulation results for BP simulation. Further reading on the effort needed for both tasks in the order picking study is given in [18]. Both tasks are not necessary using the integrated simulation since all the information is either contained in a single model or results from the integrated simulation.

Simulator	Min	Q1	Q2	Q3	Max
Measured	754	2 862	4 688	6 964	24 610
IntBIIS	400	2 988	4 208	6 184	27 560
ADONIS	754	2 630	3 814	5 798	27 010

Table 4 Quartiles for the probability distribution from Fig. 9 (decimal places truncated)

8.2.2 Experiment on Workload Burstiness

In the order picking process, too much influence factors are involved that may hamper the observation of workload burstiness. Thus, after examining the real-life example, an experiment based on a minimum process example taken from [23] is conducted to investigate the representation of workload burstiness in simulation.

The burstiness experiment is constructed so that targeted, occasional bursts of BP instances emerge, which results in temporary overload situations. The minimum example consists of a sequence of two steps, i.e., an actor step (AS1) succeeded by a system step (ISS1). There are two actors A1 and A2 available to perform AS1. ISS1 is executed by an IS that contains one hardware resource, the CPU. Owing to the experiment design (see [23] for details), pairs of BP instances reach the system step ISS1 at the same time, although they had a certain distance before the actor step AS1. This is because the BP instances are processed concurrently by the two actors. Reaching the system step, one BP instance receives service from the CPU immediately, which results in a response time of one time unit, because there is no waiting time. Another BP instance has to wait until its predecessor has been processed, before receiving service, which results in a response time of two time units. The expected mean response time of ISS1 is 1.5 time units which is predicted by two different simulation approaches hereafter.

Using the isolated simulation in [23], the actor step AS1 was modeled in ADONIS and the system step ISS1 was modeled in the original Palladio tooling. IS simulation and BP simulation were conducted in isolation which results in a mean response time for ISS1 of 1.0 time units. The BP instances arrive at ISS1 in their initial distance before the actor step. The distance between the BP instances does not shrink at AS1 due to isolated simulations. Using IntBIIS, both AS1 and ISS1 were modeled in the Palladio extension described in Sec. 7. The integrated simulation comprises the IS and the BP. The predicted mean response time of ISS1 is 1.5 time units. Both BP instances arrive at ISS1 at the same time. The temporary overload situations is correctly reflected in the integrated simulation. The experiment shows the impact of workload burstiness on performance prediction. IntBIIS correctly reflects workload burstiness in simulation. Therefore, the IntBIIS result matches the expected mean response time of ISS1. In contrast, applying isolated simulations [23] a high deviation of about 33 percent has been shown, as workload burstiness is not correctly represented.

8.2.3 Scalability Analysis

For an integrated BP and IS simulation, the ability to cope with long simulation runs that resemble one or more years of operation is just as important as its ability to handle large input models, i.e. complex business processes and information systems, gracefully. Therefore we examine and discuss the scalability of IntBIIS and show its ability to handle long and complex simulation runs. The starting point of our scalability analysis is an artificial *base model* that contains just enough information to constitute a valid model. Then, we gradually increase the model's complexity or the length of the simulation, respectively, and observe how this influences simulation performance. In our scalability study, simulation performance is the wall-clock time required to simulate a certain model until the simulated time reaches a specified upper limit, the stopping condition. In the following, we denote the required wall-clock time by *simulation execution time*. The resulting performance measurements give an impression of IntBIIS's scalability. Note that we focus on the BP extensions introduced with IntBIIS – the scalability of its underlying simulator EventSim has been shown in an earlier publication [32] and applies to the IS parts of IntBIIS.

The scalability analysis addresses the following questions. First (Scalability Experiment 1), to what extent does modeling granularity affect simulation performance? Virtually every human tasks can be decomposed into multiple smaller tasks, and, conversely, multiple small tasks can be composed to a single larger task. It is up to the modeler to choose the appropriate level of detail. More detailed models however tend to consume more simulation resources thereby posing a potential scalability issue. Second (Scalability Experiment 2), with increasing model complexity, does the simulation execution time increases linearly? By model complexity we mean the number of elements used in a model, e.g. the number of business processes or human actors. If we observe a superlinear increase, the simulation's efficiency will fall as complexity rises. In this case, complex models could not be simulated in acceptable time. And third (Scalability Experiment 3), with increasing simulation length, does the simulation execution time increases linearly? When we double the simulation length, the simulation execution time should increase no more than by the factor two. Otherwise, the simulation's efficiency will fall as the simulation length rises.

For each question, we now briefly present the experimental design and discuss the analysis results. Each experiments starts with a *base model*. As described earlier, the base model contains just enough information to constitute a valid model that can be simulated in IntBIIS. Namely, the base model comprises a single business process with a single actor step demanding 1000 work units served by a single actor resource. The inter-arrival time of the business process is set to 1000 so that the actor resource is neither underloaded nor overloaded: at the moment an actor step's demand is processed completely, a new BP instance arrives. Starting from the base model, each experiment gradually adds complexity as decribed below.

Scalability Experiment 1 examines how modeling granularity affects simulation performance. First, we slightly increase the base model's complexity by changing the number of business processes and actor resources to 10 each. This reduces the relative influence of measurement errors. Then we gradually increase the modeling granularity by splitting the actor step of each business process into i parts while holding their total demand fixed at 1000. Hence the total demand issued by each business process instance remains unchanged; only the modeling granularity changes. The variable i has been varied in the ranges $10, 20, ..., 100$ and $100, 200, ..., 1000$. The simulation execution time of each variation can be seen in Fig. 10 (left). The execution times rises slowly until reaching a fairly fined-grained modeling granularity of 100 actor steps per business process. For even finer granularities, we observe a linear increase in execution time. Note, linearity is given since the graph has two exponential axes. Although modeling granulary affects simulation performance, the linear increase prevents the granularity level from becoming a scalability issue. In consequence the modeler is free to choose the desired level of granularity.

Scalability Experiment 2 examines how model complexity affects simulation performance. Again, we start from the base model. This time, we gradually increase two variables i and j in dependence upon each other, with i being the number of concurrent business processes and j being the number of actor resources available to serve their demands. We consider two cases. In the first case, the actor resources are fully utilized, neither underloaded nor overloaded. For this to hold we set $i = j$. The simulated systems is then in a *steady state*. In the second case, we drive the simulated system into overload by demanding more capacity from human actors than they are capable of serving. For this we set $j = i - 10$, i.e. we have always 10 more business processes than actor resources. The variable i has been varied in the range $10, 20, ..., 100$. From Fig. 10 (middle) we can see that the simulation scales linearly, even for the second case where queues of actor resources are overloaded. This is interesting because in the latter case the queues become more and more crowded without any chance of recovering. The missing effect on simulation execution time can be explained by the scheduling policy employed. We use a variant of a FCFS policy which allows all relevant operations to be performed in constant time. Of course, memory usage can still become an issue and we can actually observe that in the next experiment.

Scalability Experiment 3 examines how expensive it is to simulate an additional time unit, i.e. how much processing time – measured in wall-clock time – is needed for that additional step in simulated time. The ratio between simulated time and wall-clock time must not increase over the course of a simulation. Otherwise, the longer we simulate the less efficient becomes the simulation. We consider the same two cases introduced before and compare the simulation performance for a steady state system and for a system with overloaded actor resources. In the first case, we modify the base model to comprise 10 business processes, each consisting of 10 actor steps, and 10 actor resources. Opposed to the scalability experiments discussed before, we do not apply any further gradual modifications to this model. Instead, we vary the length of the simulation, expressed in simulated time units, in the range $10^5, 10^6, ..., 10^8$. In the second case, we add 10 more business processes, so that 10 actor resources face 20 business processes. The results are depicted in Fig. 10 (right). For the steady state system, we see a curve that is even sublinear, meaning the abovementioned ratio even improves. This could be attributed to the ongoing optimisation effort by the JVM. For the overloaded system, however, the ratio tends to deteriorate over simulation time. Again, we see the cause in the JVM infrastructure. As the queues of simulated actor resources grow, the management overhead for the queued jobs grows too. This includes especially garbage collection activities. The last simulation run (10^8) even fails under the given configuration with an `OutOfMemoryError`. This shows clearly a scalability limit in regard to the available memory. The second scenario of permanently overloaded human actors is, however, artificial and would not occur in practice – at least not to that extent. For this reason we consider this issue rather a theoretical scalability issue than a practical one.

We conclude that IntBIIS is able to cope with complex input models, where the modeling granularity has no major impact on the simulation performance. From the variety of modeling elements introduced, we focused on the most important ones in our scalability study. Our observation that IntBIIS scales well is therefore to be understood as a general tendency that should be underpinned by additional scalability experiments in future. Each measurement shown in the results has been repeated 10 times to alleviate measurement errors, i.e. we performed 10 independent simulation runs. From these 10 results, we visualized the median in the result plots. For all experiments, we used DESMO-J 2.3.3 running in Oracle's Java 1.7 HotSpot^{TM} 64-bit server virtual machine (Version 24.65-b04). The *-xmx* VM argument was set to 2048M to increase the available memory to 2 GB. The simulation was executed on Windows 8.1 running on an AMD FX-8350 octa-core processor clocked at 4 GHz per core and equipped with a Samsung 840 Pro SSD.

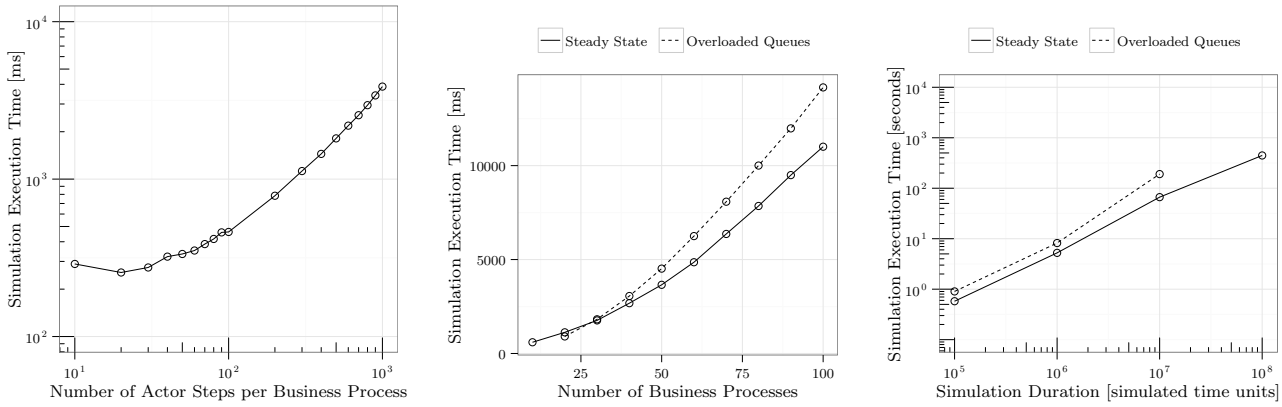

Fig. 10 Scalability of IntBIIS: Increasing Modeling Granularity (left), Increasing Model Complexity (middle), Increasing Simulation Length (right)

8.3 Practicability

Since there are several analogies between BP simulation and IS simulation, there are also several analogies in the application of the corresponding tool supports. Similar models have to be created. As described in Sec. 7, the business process model is constructed based on the PCM usage model and the organization environment model is related to the PCM hardware environment model. The simulations are configured and executed in a similar way. Metrics such as mean response/execution time or resource utilization determined by the tool support and presented to the user are closely related.

The practicability of the Palladio approach and the related tool support was validated by Martens [30]. The results confirm the practicability of Palladio for third-party users. We are currently validating the practicability of IntBIIS from the practitioners' point of view in a case study. In this paper, we discuss the practicability and maturity of our prototypical tool support by applying it to the real-life example. We extended the Palladio tool chain by model elements and simulation behavior, as described in Sec. 7. Thus, the practicability of our tool support is widely determined by the practicability of the Palladio tool chain. From the user's perspective the Palladio tool chain comprises the graphical editors for modeling PCM instances, the configuration and execution of the simulation, and the presentation of the simulation results [4]. Martens [30] validated these features as an influence factor of practicability. In the following, we discuss our extensions to these features.

Editors: Currently, we are extending the Palladio tool chain by a new graphical editor for organization environment models, as specified in Fig. 5. Moreover, we are extending the usage model editor by the new meta–model elements presented in Fig. 4 to provide comprehensive process modeling capability.

Simulation Execution: The Palladio tool chain provides automated model checking for violations of constraints. Simulation settings can be edited in a configuration dialog and automated execution of the simulation can be triggered. We reused the model checking and simulation setting functionality without making changes.

Presentation of simulation results: The Palladio tool chain provides a sensor framework to gather performance-relevant data during simulation and to calculate various performance metrics. The tooling also comprises a charting framework to visualize simulation results in several forms, such as mean values or histograms. We added sensors related to the new model elements, such as execution time of actor steps or utilization of actor resources, and reused the charting framework for the new metrics.

None of the functionality, basic behavior, or visualization related to the features were changed, but applied to the integrated tooling. Thus, indicators for the practicability from third-party users' view can be transferred from the results by Martens. We are confident that the integrated tool support is practicable for third-party users, however, this needs to be further investigated in the future.

The integrated tool support could be used to (a) model a BP and IS from practice, (b) execute the simulation of the BP and the involved IS and (c) obtain interpretable simulation results (as shown in Fig. 9). Thus, we consider our prototypical tool support as matured enough to handle real-life BPs.

8.4 Threats to Validity

In case study research, four aspects of validity are distinguished [42] – internal validity, external validity, construct validity, and conclusion validity (i.e., reliability).

Internal validity: in the order picking study, simulation results were compared to measurements from reality. Reality was mapped to a model that was input to the simulation. Consequently, the mapping from reality to the model may influence the simulation results. Therefore, the results of IntBIIS were compared to the results of another simulation method to exclude influence factors caused by modeling. However, both simulation methods use different simulation strategies, which again may influence the simulation results although the models used in the simulations are similar.

External validity: according to Runeson et al. [42], in case study research, the representativeness of a sample case may be sacrificed to achieve a deeper understanding and better realism of the phenomena under study. Consequently, the results achieved in the order picking study might not be transferable to an arbitrary other case, due to the individual properties of each case. However, the case study gives important insights and provides indicators for cases having similar properties.

Construct validity: in the particular case of the order picking process, BP workload did not suffice to drive the IS into an overload situation. Also the mutual impact of BP and IS on workload burstiness does not significantly affect the simulation results, as shown in the figures. The order picking process was not selected with the focus on this criterion, but rather as it represents a real-life example where IS response times significantly affect the BP performance. Nevertheless, the burstiness experiment conducted in Sec. 8.2.2 demonstrated the mutual impact on workload burstiness which significantly affects performance.

Conclusion validity: while analyzing simulation results, the effects of interpretation by a specific researcher must be eliminated. In order to analyze the response time distributions of the order picking process, we apply statistical tests which give a reasonable evidence and reduce the need for interpretation. In the burstiness experiment, IntBIIS matched the expected mean value for adequate workload burstiness, whereas the approach in [23] exhibits a high deviation. Consequently, due to the experiment design, there is hardly an interpretation that may lead a researcher to another conclusion.

8.5 Assumptions and Limitations

The contributions proposed in this paper rely on assumptions that may bear some limitations. As we build upon the Palladio approach, its assumptions and limitations (cf. [4]) also apply to IntBIIS. Besides the static processing rate of human actors (justified in Sec. 7.5), the modeled BP and organizational environment are assumed to be static in IntBIIS. This means that dynamic selection of suppliers or temporary adaptations in the control flow, for example, is not supported.

Another possible limitation of IntBIIS is that the different granularities of events in terms of their duration may limit the feasibility of the integrated simulation. In cases where many fine-grained IS events happen during a short time frame (e.g., a second) simulating a week or even a year may take a long time. This is because fine-grained simulation (which takes long per simulated second) is required but also a long simulated time frame is needed. However, this seems to be a hypothetic limitation as we focus on IS response times that have a discernible impact on BP performance. We do not need to consider cases in detail where the IS simulation has a large number of fine-grained events per second. Because fine-grained IS events neither affect the BP performance directly nor significantly influence the workload burstiness in a BP scenario. Only if IS events get a granularity that is comparable to BP events, they obtain importance from a business perspective. Rough estimates of fine-grained IS events are sufficient for BP performance prediction. The order picking process example includes IS events in a millisecond range where it is simulated over a long time frame (a year). As we were able to apply IntBIIS to the example, we could demonstrate the feasibility of the integrated simulation, even if fine-grained IS events are included in the simulation study.

9 Conclusion and Future Work

In this paper, we presented the novel approach IntBIIS for the integrated analysis of BPs and ISs using simulation. We proposed a holistic simulation that combines performance prediction on software architecture level and business process level. IntBIIS predicts the impact of an IS design on BP performance and vice versa. In contrast to existing approaches, workload burstiness is reflected adequately in simulation, since both, the BP impact as well as the IS impact on workload burstiness, is considered. This results in increased prediction accuracy compared to isolated simulation approaches. We proposed a scheduling policy to specifiy the behavior of human actors in simulation. IntBIIS supports the comparison of design alternatives and the verification of a certain design against requirements. In this way, the alignment of BP and IS design can be supported.

IntBIIS builds upon the Palladio tool chain to implement the integrated simulation of BPs and ISs. Based on a BP example from practice, we examined the feasibility and practicability of the approach and tooling. Compared to values measured in reality and prediction results of another BP simulation tool, IntBIIS yields

accurate simulation results. We argued for the practicability of IntBIIS and the maturity of the tool support.

In the future, we plan to improve the usability and expandability of our prototypical tool support. We plan to include further quality aspects, such as reliability and maintainability, in modeling and analysis. A better modularization of the PCM and related simulators contributes to this intention. We also plan to completely support an established BP notation, e.g. BPMN, and adapt or extend it by the modeling constructs and simulation strategies proposed in this paper.

Another topic of future work is improving the reflection of human behavior in simulation. Beyond the proposed scheduling policy, there are various aspects that might influence the behavior and performance of human actors in terms of the activities to perform, for example the humans' stress level, whether they are tired or ill, whether they like the activity to perform, or whether they have experience with the activity or similar activities, and so on. Modeling the behavior of human actors is one of the major challenges in business process simulation. There is a lot of future work to do, which must be conducted in a close interaction with social sciences, to adequately reflect the characteristics of humans. It is an open issue to find a trade off between the representation of realistic human behavior and modeling effort.

Moreover, we want to apply IntBIIS to further industrial cases and to conduct further investigation on the practicability of the tool support, for example using controlled experiments. Further investigations may also examine the usefullness of IntBIIS for operational decision making.

Acknowledgements The authors thank Thor GmbH for giving us the opportunity to apply IntBIIS in a real-life case study. We also give thanks to the anonymous reviewers and to Ralf Reussner for review and valuable comments. This work is partially supported by the DFG (German Research Foundation) in the Priority Programme SPP 1593: Design For Future – Managed Software Evolution.

References

1. A. T. M. Aerts, J. B. M. Goossenaerts, D. K. Hammer, and J. C. Wortmann. Architectures in context: on the evolution of business, application software, and ICT platform architectures. *Information and Management*, 41(6):781–794, 2004.
2. J. Barjis. The importance of business process modeling in software systems design. *Science of Computer Programming*, 71(1):73–87, 2008.
3. F. Bause. Queueing Petri Nets - A formalism for the combined qualitative and quantitative analysis of systems. In *Proceedings of the 5th International Workshop on Petri Nets and Performance Models*, pages 14–23, 1993.
4. S. Becker, H. Koziolek, and R. Reussner. The Palladio component model for model-driven performance prediction. *Journal of Systems and Software*, 82:3–22, 2009.
5. S. Betz, E. Burger, A. Eckert, A. Oberweis, R. Reussner, and R. Trunko. An approach for integrated lifecycle management for business processes and business software. In I. Mistrík, A. Tang, R. Bahsoon, and J. A. Stafford, editors, *Aligning Enterprise, System, and Software Architectures*. IGI Global, 2012.
6. R. Böhme and R. Reussner. Validation of predictions with measurements. In I. Eusgeld, F. C. Freiling, and R. Reussner, editors, *Dependability Metrics*. Springer-Verlag, 2008.
7. F. Brosch, H. Koziolek, B. Buhnova, and R. Reussner. Architecture-based reliability prediction with the Palladio component model. *Software Engineering, IEEE Transactions on*, 38(6):1319–1339, 2012.
8. M. Chinosi and A. Trombetta. BPMN: An introduction to the standard. *Computer Standards & Interfaces*, 34(1):124–134, 2012.
9. T. Davenport. *Process Innovation: Reengineering Work through Information Technology*. Harvard Business School Press. 1993.
10. J. Ehlers and W. Hasselbring. A self-adaptive monitoring framework for component-based software systems. In I. Crnkovic, V. Gruhn, and M. Book, editors, *ECSA*, volume 6903 of *LNCS*, pages 278–286. Springer, 2011.
11. H. Eichelberger and K. Schmid. Flexible resource monitoring of java programs. *Journal of Software Systems*, 93:163–186, 2014.
12. G. Franks. Simulating layered queueing networks with passive resources. In *Proceedings of the 2011 Symposium on Theory of Modeling & Simulation*, pages 8–15. Society for Computer Simulation International, 2011.
13. G. M. Giaglis, R. J. Paul, and R. M. O'Keefe. Research note: Integrating business and network simulation models for IT investment evaluation. *Logistics Information Management*, 12:108–117, 1999.
14. B. Gladwin and K. Tumay. Modeling business processes with simulation tools. In M. S. Manivannan and J. D. Tew, editors, *Proceedings of the 26th Conference on Winter Simulation*, pages 114–121. Society for Computer Simulation International, 1994.
15. S. Graupner, J. Rolia, and N. Edwards. Deriving IT configurations from business processes. In *Proceedings of the 10th Conference on E-Commerce Technology and the 5th Conference on Enterprise Computing*, pages 317–322. IEEE, 2008.
16. A. Gucegliooglu. *A Pre-Enactment Model for Measuring Process Quality*. PhD Thesis, METU. 2006.
17. W. Hasselbring, R. Heinrich, R. Jung, A. Metzger, K. Pohl, R. Reussner, and E. Schmieders. iObserve: Integrated observation and modeling techniques to support adaptation and evolution of software systems. Research report, Kiel University, Kiel, Germany, October 2013.
18. R. Heinrich. *Aligning Business Process Quality and Information System Quality*. PhD Thesis, Software Engineering Heidelberg. 2013.
19. R. Heinrich, J. Henss, and B. Paech. Extending Palladio by business process simulation concepts. In S. Becker, J. Happe, A. Koziolek, and R. Reussner, editors, *Palladio Days 2012 Proceedings*, pages 19–27. CEUR-WS.org, 2012.
20. R. Heinrich, A. Kappe, and B. Paech. Modeling quality information within business process models. In S. Wagner et al., editor, *Proceedings of the 4th SQMB Workshop*, pages 4–13. TUM-I1104, 2011.

21. R. Heinrich, A. Kappe, and B. Paech. Tool support for the comprehensive modeling of quality information within business process models. In M. Nüttgens, O. Thomas, and B. Weber, editors, *Enterprise Modelling and Information Systems Architecture*, LNI Vol. P-190, pages 213–218. GI, 2011.
22. R. Heinrich and B. Paech. Defining the quality of business processes. In G. Engels, D. Karagiannis, and H. C. Mayr, editors, *Modellierung 2010*, LNI Vol. P-161, pages 133–148. GI, 2010.
23. R. Heinrich and B. Paech. On the prediction of the mutual impact of business processes and enterprise information systems. In S. Kowalewski and B. Rumpe, editors, *Software Engineering 2013*, LNI Vol. P-239, pages 157–170, 2013.
24. R. Heinrich, E. Schmieders, R. Jung, K. Rostami, A. Metzger, W. Hasselbring, R. Reussner, and K. Pohl. Integrating run-time observations and design component models for cloud system analysis. In S. Götz, N. Bencomo, and R. France, editors, *Proceedings of the 9th International Workshop on Models at run.time*, pages 41–46. CEUR-WS.org, 2014.
25. J. Herbst, S. Junginger, and H. Kühn. Simulation in financial services with the business process management system ADONIS. In W. Hahn and A. Lehmann, editors, *Proceedings of the 9th European Simulation Symposium*, pages 491–495. Society for Computer Simulation, 1997.
26. S. Kounev and C. Dutz. QPME - A Performance Modeling Tool Based on Queueing Petri Nets. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):46–51, 2009.
27. H. Koziolek and R. Reussner. A model transformation from the Palladio Component Model to Layered Queueing Networks. In S. Kounev, I. Gorton, and K. Sachs, editors, *Performance Evaluation: Metrics, Models and Benchmarks*, volume 5119 of *LNCS*, pages 58–78. Springer, 2008.
28. A. M. Law and W. D. Kelton. *Simulation modeling and analysis*. McGraw-Hill series in industrial engineering and management science. McGraw-Hill, 2000.
29. E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik. *Quantitative System Performance Computer System Analysis Using Queueing Network Models*. Prentice-Hall, 1984.
30. A. Martens. Empirical Validation of the Model-driven Performance Prediction Approach Palladio. Master's thesis, University of Oldenburg, 2007.
31. P. Meier, S. Kounev, and H. Koziolek. Automated Transformation of Palladio Component Models to Queueing Petri Nets. In *19th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2011.
32. P. Merkle and J. Henss. EventSim – an event-driven Palladio software architecture simulator. In S. Becker, J. Happe, and R. Reussner, editors, *Palladio Days 2011 Proceedings*, pages 15–22, 2011.
33. N. Mi, G. Casale, L. Cherkasova, and E. Smirni. Burstiness in multi-tier applications: symptoms, causes, and new models. In V. Issarny and R. Schantz, editors, *Proceedings of the 9th International Conference on Middleware*, pages 265–286. Springer, 2008.
34. J. Nakatumba and W. Aalst. Analyzing Resource Behavior Using Process Mining. In S. Rinderle-Ma, S. Sadiq, and F. Leymann, editors, *BPM 2009 Workshops, Proceedings of the 5th Workshop on Business Process Intelligence*, volume 43 of *LNBIP*, pages 69–80. Springer, 2010.
35. Object Management Group (OMG). Business Process Model and Notation (BPMN) Version 2.0. Technical report, 2011.
36. M. K. Painter, R. Fernandes, N. Padmanaban, and R. Mayer. A methodology for integrating business process and information infrastructure models. In J. M. Charnes, D. J. Morrice, D. T. Brunner, and J. J. Swain, editors, *Proceedings of the 28th Conference on Winter Simulation*, pages 1305–1312. IEEE, 1996.
37. M. D. Petty, J. Kim, S. E. Barbosa, and J.-J. Pyun. Software frameworks for model composition. *Modelling and Simulation in Engineering*, 2014:18, 2014.
38. J. Rolia and K. Sevcik. The method of layers. *Software Engineering, IEEE Transactions on*, 21(8):689–700, 1995.
39. A. Rozinat, M. Wynn, W. Aalst, A. Hofstede, and C. Fidge. Workflow Simulation for Operational Decision Support using YAWL and ProM. BPM Center Report BPM-08-04, 2008.
40. Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, 84(1):25–43, 2001.
41. Y. Rubner, C. Tomasi, and L. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the 6th International Conference on Computer Vision*, pages 59–66. IEEE, 1998.
42. P. Runeson, M. Host, A. Rainer, and B. Regnell. *Case Study Research in Software Engineering: Guidelines and Examples*. Wiley, 2012.
43. B. Schroeder and G. A. Gibson. Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You? In *Proceedings of the 5th Conference on File and Storage Technologies*, pages 1–16. USENIX Association, 2007.
44. A. Serrano and M. den Hengst. Modelling the integration of BP and IT using business process simulation. *Enterprise Information Management*, 18:740–759, 2005.
45. Simulation Interoperability Standards Committee. IEEE standard for modeling and simulation High Level Architecture (HLA) - Framework and Rules, 2000.
46. C. Smith. *Performance Engineering of Software Systems*. Addison–Wesley, 1990.
47. J. Song, T. Luo, and S. Chen. Behavior pattern mining: Apply process mining technology to common event logs of information systems. In *Proceedings of the International Conference on Networking, Sensing and Control*, pages 1800–1805. IEEE, 2008.
48. Y. Tan and S. Takakuwa. Predicting the impact on business performance of enhanced information system using business process simulation. In H. S. G., B. B., H. M.-H., S. J., T. J. D., and B. R. R., editors, *Proceedings of the 39th Conference on Winter Simulation*, pages 2203–2211. IEEE, 2007.
49. W. van der Aalst, J. Nakatumba, A. Rozinat, and N. Russell. Business process simulation: How to get it right. *BPM Center Report BPM-08-07*, 2008.
50. W. M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 1st edition, 2011.
51. W. M. P. van der Aalst, B. F. van Dongen, C. W. Gunther, R. S. Mans, A. K. A. de Medeiros, A. Rozinat, V. Rubin, M. Song, H. M. W. E. Verbeek, and A. J. M. M. Weijters. Prom 4.0: Comprehensive support for real process analysis. In J. Kleijn and A. Yakovlev, editors, *Petri Nets and Other Models of Concurrency*, volume 4546 of *LNCS*, pages 484–494. Springer, 2007.

52. H. Vangheluwe, J. De Lara, and P. J. Mosterman. An introduction to multi-paradigm modelling and simulation. In B. F. and G. N., editors, *Proceedings of the Conference on Simulation and Planning in High Autonomy Systems*, pages 9–20. AIS, 2002.
53. WfMC. Business Process Simulation Specification (BPSim) Version 1.0. Technical report, 2013.
54. R. Wieringa, H. Blanken, M. Fokkinga, and P. Grefen. Aligning application architecture to the business context. In J. Eder and M. Missikoff, editors, *Advanced Information Systems Engineering*, LNCS 2681, pages 209–225. Springer, 2003.
55. WinterGreen Research. Business Process Management (BPM) Market Shares, Strategies, and Forecasts, Worldwide, 2012 to 2018, 2012.
56. A. Wombacher and M.-E. Iacob. Start time and duration distribution estimation in semi-structured processes. In S. Y. Shin and J. C. Maldonado, editors, *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1403–1409. ACM, 2013.
57. Workflow Management Coalition Specification. *Terminology & Glossary (WFMC-TC-1011)*. Feb. 1999.