Business Process Simulation Survival Guide

Wil M.P. van der Aalst

Abstract Simulation provides a flexible approach to analyzing business processes. Through simulation experiments various "what if" questions can be answered and redesign alternatives can be compared with respect to key performance indicators. This chapter introduces simulation as an analysis tool for business process management. After describing the characteristics of business simulation models, the phases of a simulation project, the generation of random variables, and the analysis of simulation results, we discuss 15 risks, i.e., potential pitfalls jeopardizing the correctness and value of business process simulation. For example, the behavior of resources is often modeled in a rather naïve manner resulting in unreliable simulation models. Whereas traditional simulation approaches rely on hand-made models, we advocate the use of process mining techniques for creating more reliable simulation models based on real event data. Moreover, simulation can be turned into a powerful tool for operational decision making by using real-time process data.

1 Introduction

Simulation was one of the first applications of computers. The term "Monte Carlo simulation" was first coined in the Manhattan Project during World War II, because of the similarity of statistical simulation to games of chance played in the Monte

W.M.P. van der Aalst (✉)
Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O.Box 513, Eindhoven, MB NL-5600, The Netherlands

Business Process Management Discipline, Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia

International Laboratory of Process-Aware Information Systems, National Research University Higher School of Economics, 33 Kirpichnaya Street, Moscow, Russia
e-mail: w.m.p.v.d.aalst@tue.nl

Carlo Casino. This illustrates that that already in the 1940s people were using computers to simulate processes (in this case to investigate the effects of nuclear explosions). Later Monte Carlo methods were used in all kinds of other domains ranging from finance and telecommunications to logistics and workflow management. For example, note that the influential and well-known programming language Simula (Dahl and Nygaard 1966), developed in the 1960s, was designed for simulation. Simulation has become one of the standard analysis techniques used in the context of operations research and operations management. Simulation is particularly attractive since it is versatile, imposes few constraints, and produces results that are relatively easy to interpret. Analytical techniques have other advantages but typically impose additional constraints and are not as easy to use (Buzacott 1996). Therefore, it is no surprise that in the context of *Business Process Management* (BPM), simulation is one of the most established analysis techniques supported by a vast array of tools (van der Aalst 2013; Rosemann and vom Brocke 2014).

Consider for example a large car rental agency (like Hertz or Avis) having thousands of offices in different countries sharing a centralized information system where customers can book cars online. One can make simulation models of individual offices and the centralized information system to answer question such as:

- What are the average waiting times of customers when booking a car online?
- What is the variability of waiting times when picking up a car at a particular location?
- What is the utilization of staff at a particular location?
- Will waiting times be reduced substantially if extra staff is deployed?
- How many customers are lost due to excessive waiting times?
- What is the effect of allocating staff based on the number of bookings?
- What is the effect of changing the opening hours at a particular location?

To answer these and many other questions, a *simulation model* can be used. A proper simulation model is a simplified representation of reality and thus can be used to simulate that reality using a computer. Obvious reasons for using a simulation model are (van der Aalst and Stahl 2011; van der Aalst and Voorhoeve 2000):

- Gaining *insight* in an existing or proposed future situation. By charting a business process, it becomes apparent what is important and what is not.
- A real experiment may be *too expensive*. Simulation is a cost-effective way to analyze several alternatives. Decisions such as hiring extra staff or adding new servers many too expensive to simply try out in reality. One would like to know in advance whether a certain measure will have the desired effect.
- A real experiment may be *too dangerous* and may not be *repeatable*. Some experiments cannot be carried out in reality due to legal, ethical, or safety reasons. Moreover, it is often impossible to reliably compare alternatives due to changing conditions (performance may change due to external factors).

There is an abundance of mathematical models that can be used to analyze abstractions of business processes. Such models are often referred to as *analytical* models. These models can be analyzed without simulation. Examples are queueing models (Kleinrock 1975), queueing networks (Baskett et al. 1975), Markov chains, and stochastic Petri nets (Haas 2002; Ajmone Marsan et al. 1995). If a simple analytical model can do the job, one should not use simulation. In comparison to a simulation model, an analytical model is typically less detailed and requires fewer parameter settings. Widely acknowledged *advantages* of simulation are:

- Simulation is *flexible*. Any situation, no matter how complex, can be investigated through simulation.
- Simulation can be used to *answer a wide range of questions*. It is possible to assess waiting times, utilization rates and fault percentages using one and the same model.
- Simulation stimulates *creativity*. Simulation triggers "process thinking" without restricting the solution space upfront.
- Simulation is *easy to understand*. In essence, it is nothing but replaying a modeled situation. In contrast to many analytical models, little specialist knowledge is necessary to understand the analysis technique used. Hence, simulation can be used to communicate ideas effectively.

Unfortunately, simulation also has some *disadvantages*.

- A simulation study can be *time consuming*. Sometimes, very long simulation runs are necessary to obtain reliable results.
- One has to be very careful when *interpreting* simulation results. Determining the reliability of results can be very treacherous indeed.
- Simulation *does not provide any proof*. Things that can happen in reality may not be witnessed during some simulation experiment.

Today's simulation tools can be used to rapidly construct simulation models using drag-and-drop functionality. However, faulty simulation models or incorrectly interpreted results may lead to bad decisions. Therefore, this chapter will focus on the validation of simulation models and the correct derivation and interpretation of simulation results. We will highlight potential pitfalls of traditional simulation approaches. Therefore, this chapter can be viewed as a *"survival guide"* for people new to the topic. Moreover, we also aim to broaden the view for people familiar with traditional business process simulation approaches. The availability of detailed event data and possible connections between simulation tools and information systems enables new forms of simulation. For example, *short-term simulation* provides users and managers with a "fast forward button" to explore what will happen in the near future under different scenarios.

The remainder of this chapter is organized as follows. Section 2 introduces traditional business process simulation by describing the simulation-specific elements of process models and by discussing the different phases in a typical simulation project. Section 3 discusses the role of pseudo-random numbers in simulation. Section 4 explains how to set up a simulation experiment and how to compute confidence intervals. Pitfalls that need to be avoided are discussed in Sect. 5. Section 6 discusses more advanced forms of simulation that exploit the availability of event data and modern IT infrastructures. Section 7 concludes the chapter with suggestions for further reading.

2 Traditional Approach to Business Process Simulation

The correctness, effectiveness, and efficiency of an organization's business processes are vital for survival in today's competitive world. A poorly designed business process may lead to long response times, low service levels, unbalanced resource utilization, angry customers, back-log, damage claims, and loss of goodwill. This is why it is important to analyze processes before they are put into production (to find design flaws), but also while they are running (for diagnosis and decision support). In this section, we focus on the role of simulation when analyzing business processes at design time.

2.1 Simulation Models

For the construction of a simulation model and to conduct experiments, we need a simulation tool. Originally, there were two typical kinds of simulation tools:

- A *simulation language* is a programming language with special provisions for simulation. Classical examples of simulation languages are Simula, GPSS, Simscript, Simpas, MUST and GASP.
- A *simulation package* is a tool with building blocks for a certain application area, which allow the rapid creation of a simulation model, mostly graphically. Classical examples of simulation packages for production processes are: Sim-Factory, Witness and Taylor. Examples of simulation packages specifically designed for workflow analysis are Protos, COSA, WoPeD, and Yasper. In fact, most of today's BPM systems provide such a simulation facility.

The advantage of a simulation language is that almost every situation can be modeled. The disadvantage is that one is forced to chart the situation in terms of a programming language. Modeling thus becomes time-consuming and the simulation program itself provides no insights. A simulation package allows to rapidly build an intuitive model. Because the model must be built from ready-made building blocks, the area of application is limited. As soon as one transgresses the limits of the specific area of application, e.g., by changing the control structure, modeling becomes cumbersome or even impossible.

Fortunately, many tools have been introduced with characteristics of both a simulation language and a simulation package. These tools combine a graphical design environment and a programming language while also offering graphical analysis capabilities and animation. Examples of such tools are Petri-net-based simulators such as ExSpect and CPN Tools (van der Aalst and Stahl 2011). These allow for hierarchical models that can be constructed graphically while parts can be parameterized and reused. The ARENA simulation tool developed by Rockwell Automation also combines elements of both a simulation language (flexibility and expensiveness) and simulation package (easy to use, graphical, and offering predefined building blocks). ARENA emerged from the block-oriented simulation language SIMAN. The use of proprietary building blocks in tools such as ARENA makes it hard to interchange simulation models between packages. Simulation tools based on more widely used languages such Petri nets or BPMN are more open and can exchange process models with BPM systems and other analysis tools (e.g., process mining software).

In the remainder of this chapter we remain tool-independent and focus on the essential characteristics of simulation.

To explain the typical ingredients of a model used for business process simulation, we first focus on the control-flow of a business process. Figure 1 shows the same control-flow using three widely used notations. Figure 1a shows a *Petri net*; a *WF-net* (WorkFlow net) to be precise (van der Aalst and Stahl 2011; ter Hofstede et al. 2010; Weske 2007). Activities are modeled by labeled transitions and the ordering of these activities is controlled by places (represented by circles). A transition (represented by a square) is enabled if each of its input places contains a token. An enabled transition may occur thereby consuming a token from each input place and producing a token for each output place. Initially, source place *in* contains a token. Hence, transition a is enabled in the initial state. After registering a request (modeled by transition a), extra insurance can be added (b) or not (modeled by the silent transition). Then the check-in is initiated (c). Subsequently, the selection of the car (d), the checking of the license (e), and the charging of the credit card (f) are executed (any ordering is allowed, including the concurrent execution of d, e, and f). Finally, the car is provided (g). The process instance terminates when place out is marked. Figure 1b shows an event log describing some example traces.

BPMN, EPCs, UML ADs, and many other business process modeling notations have in common that they all use token-based semantics. Therefore, there are many techniques and tools to convert Petri nets to BPMN, BPEL, EPCs and UML ADs, and vice versa. As a result, the core concepts of Petri nets are often used indirectly, e.g., to enable analysis, to enact models, and to clarify semantics. For example, Fig. 1c shows the same control-flow modeled using the *Business Process Modeling Notation* (BPMN). BPMN uses activities, events, and gateways to model the control-flow. In Fig. 1c two types of gateways are used: exclusive gateways are used to model XOR-splits and joins and parallel gateways are used to model AND-splits and joins. BPMN also supports other types of gateways corresponding to inclusive OR-splits and joins, deferred choices, etc. (Dumas et al. 2013; ter Hofstede et al. 2010; Weske 2007). *Event-driven Process Chains* (EPCs) use functions, events, and connectors to model the controlflow (cf. Fig. 1d). Connectors in EPCs are similar to gateways in BPMN. There are OR, XOR, and AND

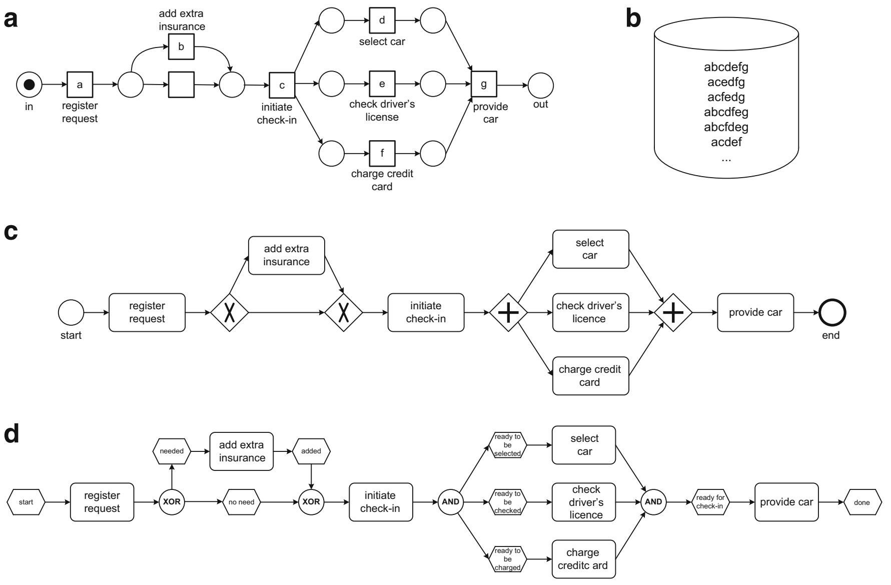


Fig. 1 Three types of models describing the same control-flow: (**a**) Petri net, (**c**) BPMN, and (**d**) EPC. The event log (**b**) shows possible traces of this model using the short activity names provided by the Petri net

connectors. Events in EPCs are similar to places in Petri nets. Just like places and transitions in a Petri net, events and functions need to alternate along any path in an EPC. However, events cannot have multiple successor nodes, thus making it impossible to model deferred choices (ter Hofstede et al. 2010). *UML Activity Diagrams* (UML ADs) – not shown in Fig. 1 – are similar to BPMN and EPCs when it comes to the basic control-flow constructs.

The control-flow oriented models shown in Fig. 1 provide necessary but not sufficient information for business process simulation. Figure 2 sketches the minimal additional information that needs to be provided to conduct meaningful simulation experiments. First of all, a *simulation environment* needs to be provided that generates new cases according to some predefined *arrival process* and that collects statistics based on the *Key Performance Indicators* (KPIs) of interest. Often a so-called Poisson arrival process is used (the time in-between two arrivals is sampled from a negative-exponential distribution). Typical KPIs are average flow time, service level, mean utilization, etc. Choices modeled in the process need to be resolved when executing a simulation model. Therefore, *priorities* and *probabilities* can be used. For example, in Fig. 2 one could specify that on average 80 % of cases skip the extra insurance (i.e., *b* is executed in 20 % of cases). One also needs to model the *duration of activities*. In most business processes, the average flow time of a case is much longer than the average service time (i.e., the time actually worked

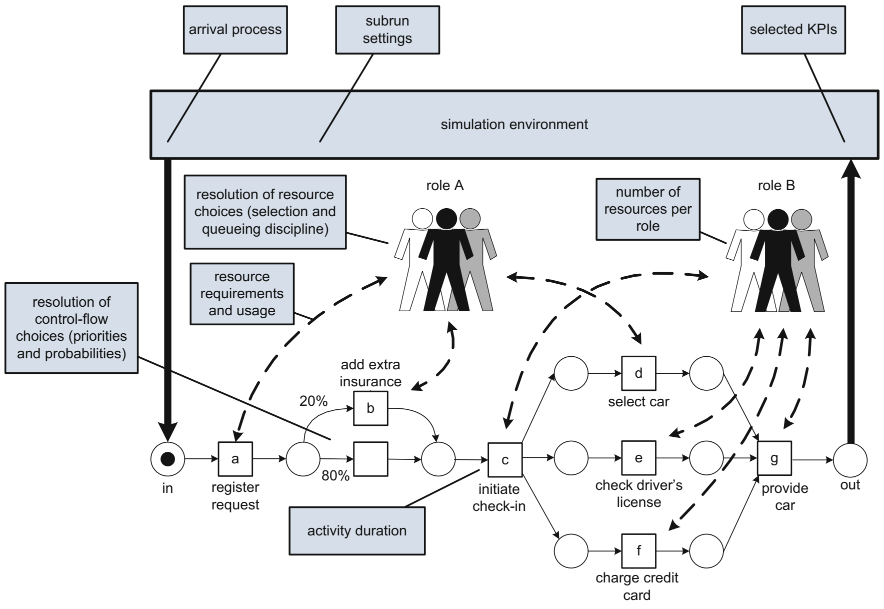


Fig. 2 Information required for business process simulation. This information is not needed for enactment (using for example a BPM/WFM system), but needs to be added for simulation

on the case). This is due to queueing for unavailable or busy resources. Often activities require a particular type of resource, commonly referred to as a *role*. Several resources may have the same role and several activities may require a particular role. The simulation model needs to specify *resource requirements and usage*. Also the *number of resources* per role, the *selection of resources* and the *ordering of pending activities* need to be specified. For example, a round-robin mechanism can be used to select available resources and a First-Come First-Served (FCFS) queueing discipline can be used to order pending activities. Other queueing disciplines are Last-Come First-Served (LCFS), Random Order (RO), Rush Orders First (ROF), and Shortest Processing Time First (SPTF).

To conduct experiments, one also needs to determine the *number of subruns*, *subrun length*, and *warm-up period*. As explained in Sect. 4, these subrun settings are needed to be able to compute *confidence intervals*.

Interestingly, one does not need to supply the additional information shown in Fig. 2 when configuring a *Business Process Management* (BPM) or *Workflow Management* (WFM) *system* (van der Aalst 2013; Dumas et al. 2013; ter Hofstede et al. 2010; Weske 2007). For example, activity durations and routing probabilities emerge over time based on the real characteristics of cases and resources.

2.2 Life-Cycle of BPM and Simulation Projects

To explain the role of simulation as an analysis tool, we start by discussing the *BPM life-cycle* (van der Aalst 2013; van der Aalst and Stahl 2011) shown in Fig. 3. In the *(re)design phase*, a process model is designed. This model is transformed into a running system in the *implementation/configuration phase*. If the model is already in executable form and a WFM or BPM system is already running, this phase may be very short. However, if the model is informal and needs to be hard-coded using some conventional programming language, this phase may take substantial time. After the system supports the designed processes, the *run & adjust phase* starts. In this phase, the processes are enacted and adjusted when needed. In the run & adjust phase, the process is not redesigned and no new software is created; only predefined controls are used to adapt or reconfigure the process. Figure 3 shows two types of analysis: *model-based analysis* and *databased analysis*. While the system is running, event data are collected. These data can be used to analyze running processes, e.g., discover bottlenecks, waste, and deviations. This is input for the redesign phase. During this phase process models can be used for analysis. For example, simulation is used for "what if" analysis or the correctness of a new design is verified using model checking.

Traditionally, simulation is positioned on the left-hand side of Fig. 3, i.e., business process simulation is a form of model-based analysis conducted during the (re)design phase. Figure 4 shows the phases of a typical simulation project. These phases should be seen as a further refinement of the (re)design phase in Fig. 3.

The simulation process starts with a *problem definition*, describing the goals and fixing the scope of the simulation study. The scope tells what will and what will not be a part of the simulation model. The problem definition should also state the questions to be answered. Preferably, these questions should be quantifiable. Instead of asking "Are the customers satisfied?", one should ask "How long do customers have to wait on average?"

After defining the problem, the next phase is *modeling*. In this phase the *conceptual model* is created. The conceptual model defines classes of *objects* and the *relations* between these objects. In the case of a car rental organization example objects to be distinguished are cars, customers, staff members, parking spaces, etc. The relevant characteristics (properties) of these objects need to be determined. The construction of the conceptual model will most likely unveil incomplete and contradictory aspects in the problem definition. Also, the modeling process may bring forth new questions for the simulation study to answer. In either case, the problem definition should be adjusted.

After the conceptual modeling phase, the *realization phase* starts. Here, the conceptual model is mapped onto an *executable model*. The executable model can be directly simulated on the computer. How to create this model depends strongly on the simulation tool used. Simulation languages require a genuine design and implementation phase. Simulation packages that fit the problem domain merely

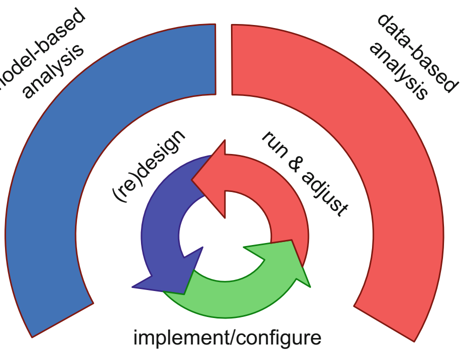


Fig. 3 BPM life-cycle consisting of three phases: (re)design, implement/configure, and run & adjust. Traditional simulation approaches can be seen as a form of model-based analysis mostly used during the (re)design phase

require a correct parameterization. The objects of the conceptual model are mapped to building blocks from the package and their quantitative characteristics (e.g. speed) are translated to parameter values of these building blocks.

An executable model is not necessarily correct, so it has to be *verified*. Verification of the model is necessary to examine whether the model contains qualitative or quantitative errors, like programming errors or wrong parameter settings. For verification purposes, small trial runs can be simulated step-by-step, or a stress test can be applied to the model. In the stress test the model is subjected to extreme situations, like having more customers arrive than can be attended to. In such a case,

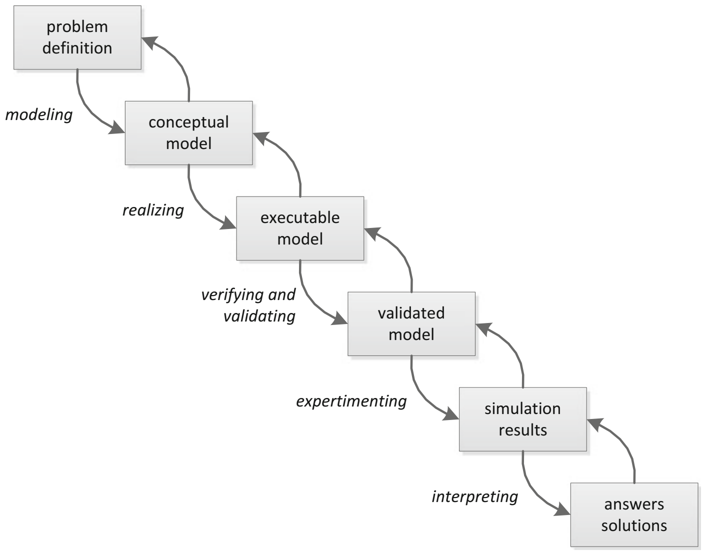


Fig. 4 Phases of a traditional simulation study

waiting times measured should increase dramatically in the course of time. Some tools support more advanced forms of verification (van der Aalst 2013; van der Aalst and Stahl 2011). Apart from verification, *validation* of the model is also required. During validation we compare the simulation model with reality. When simulating an existing situation, the results of a simulation run can be compared to observations from historical data. Verification and validation may lead to adjustments of the simulation model. New insights may even lead to adjusting the problem definition and/or the conceptual model. A simulation model found to be correct after validation is called a *validated model*.

Starting from the validated model, *experiments* can be carried out. These experiments have to be conducted in such a way that reliable results are obtained as efficiently as possible. In this stage decisions will be made concerning the number of simulation runs and the length of each run (cf. Sect. 4).

The simulation results need to be *interpreted* to allow feedback to the problem definition. Confidence intervals will have to be calculated for the various KPIs based on low-level measurements gathered during simulation. Also, the results will have to be interpreted to answer the questions in the problem definition. For each such answer, the corresponding reliability should be stated. All these matters are summarized in a final report with answers to questions from the problem definition and proposals for solutions.

Figure 4 shows that feedback is possible between phases. In practice, many phases do overlap. Specifically, experimentation and interpretation will often go hand in hand.

Figure 4 may be misleading as it refers to a single simulation model. Usually, several *alternative situations* are compared to one another. In that case, several simulation models are created and experimented with and the results are compared. Often, several possible improvements of an existing situation have to be compared through simulation. We call this “what if” analysis. Simulation is well-suited for “what if” analysis as it is easy to vary parameters and compare alternatives based on selected KPIs.

3 Sampling from Distributions

Figure 2 illustrates that random variables need to be added to resolve choices, to sample durations from some probability distribution, and to generate the arrival of new cases. This section shows how to introduce “randomness” selectively.

3.1 Pseudo-Random Numbers

A simulation experiment is little more than replaying a modeled situation. To replay this situation in computer, we have to make assumptions not only for the modeled business process itself but also for its *environment* (cf. Fig. 2). As we cannot or will not model these matters in detail we turn to “Monte Carlo”. We do not know when and how many customers will enter a car rental office, but we do know the mean and variation of customer arrivals. So, we have the computer take seemingly random samples from a probability distribution. The computer is by nature a deterministic machine, so we need to smartly generate so-called *pseudo-random numbers*.

A *random generator* is a piece of software for producing pseudo-random numbers. The computer does in fact use a deterministic algorithm to generate them, which is why they are called “pseudo random”. Most random generators generate pseudo-random numbers between 0 and 1. Each value between 0 and 1 being equally probable, these values are said to be distributed *uniformly* over the interval between 0 and 1.

Most random generators generate a series of pseudo-random numbers $\frac{X_i}{m}$ according to the formula:

$$X_n = (aX_{n-1} + b) \text{ modulo } m$$

For each i, X_i is a number from the set $\{0, 1, 2, \ldots, m - 1\}$ and $\frac{X_i}{m}$ matches a sample from a uniform distribution between 0 and 1. The numbers a, b and m are chosen in such a way that the sequence can hardly or not at all be distinguished from “truly random” numbers. This means that the sequence X_i must visit, on average, each of the numbers $0, 1, 2, \ldots, m - 1$ equally often. Also, m is chosen as closely as possible to the largest integer that can be manipulated directly by the computer. There are several tests to check the quality of a random generator [cf. (Bratley et al. 1983; Law and Kelton 1982; Pidd 1989; Shannon 1975)]: frequency test, correlation test, run test, gap test and poker test.

A reasonable random generator for a 32-bit computer is:

$$X_n = 16807X_{n-1} \text{ modulo } \left(2^{31} - 1\right)$$

That is: $a = 16807$, $b = 0$ and $m = 2^{31}-1$. For a 64-bit machine:

$$Xn = (6364136223846793005X_{n-1} + 1) \text{ modulo } 2^{64}$$

is a good choice.

The first number in the sequence (X_0) is called the *seed*. The seed completely determines the sequence of random numbers. In a good random generator, different seeds produce different sequences. Sometimes the computer selects the seed itself (e.g., based on a system’s clock). However, preferably the user should consciously select a seed himself, allowing the *reproduction* of the simulation experiment later. Reproducing a simulation experiment is important whenever an unexpected phenomenon occurs that needs further examination.

Today's simulation tools provide adequate random generators. This generator can be seen as a black box: a device that produces (pseudo) random numbers upon request. However, beware: pseudo-random numbers are not truly random! (A deterministic algorithm is used to generate them.) Do not use more than one generator and take care when selecting the seed.

To illustrate the dangers in using random generators we mention two well-known pitfalls.

The first mistake is using the so-called 'lower order bits' of a random sequence. For example, if a random generator produces the number 0.1321734234, the higher order digits 0.13217 are 'more random' than the lower order digits 34234. In general the lower order digits show a clear cyclical behavior.

Another frequent mistake is the double use of a random number. Suppose that the same random number is used twice for generating a sample from a probability distribution. This introduces a dependency into the model that does not exist in reality, which may lead to extremely deceptive results.

3.2 Example Probability Distributions

Only rarely do we need random numbers uniformly distributed between 0 and 1. Depending on the situation, we need samples from different *probability distributions*. A probability distribution specifies which values are possible and how probable each of those values is.

To simplify the discussion of random distributions and samples from probability distributions, we introduce the term *random variable*. A random variable X is a variable with a certain probability of taking on certain values. For example, we can model the throwing of a dice by means of a variable X that can take on the values 1, 2, 3, 4, 5 and 6. The probability of obtaining any value a from this set is $\frac{1}{6}$. We can write this as follows:

$$\mathbb{P}[X = a] = \begin{cases} \dfrac{1}{6} & \text{if } a \in \{1,2,3,4,5,6\} \\ 0 & \text{else} \end{cases}$$

Given a random variable X we can define its *expectation* and *variance*. The expectation of X, denoted by $\mathbb{E}[X]$, is the average to be expected from a large number of samples from X. We also say the *mean* of X. The variance, denoted as Var[X], is a measure for the average deviation of the mean (expectation) of X. If X has a high variance, many samples will be distant from the mean. Conversely, a low variance means that, in general, samples will be close to the mean. The expectation of a random variable X is often denoted with the letter μ, the variance (Var[X]) is denoted as σ^2. The relation between expectation and variance is defined by the following equality:

Table 1 Discrete random distributions

Distribution	Domain	$\mathbb{P}[X = k]$	$\mathbb{E}[X]$	Var[X]
Bernoulli $0 \leq p \leq 1$	$k \in \{0,1\}$	$\begin{cases} 1-p & k=0 \\ p & k=1 \end{cases}$	p	$p(1-p)$
Homogeneous $a < b$	$k \in \{a,\ldots,b\}$	$\frac{1}{(b-a)+1}$	$\frac{a+b}{2}$	$\frac{(b-a)((b-a)+2)}{12}$
Binomial $0 \leq p \leq 1$ $n \in \{1,2,\ldots\}$	$k \in \{0,1,\ldots,n\}$	$\binom{n}{k} p^k (1-p)^{n-k}$	$n\,p$	$n\,p(1-p)$
Geometric $0 \leq p \leq 1$	$k \in \{1,2,\ldots\}$	$(1-p)^{k-1}\,p$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Poisson $\lambda > 0$	$k \in \{0,1,\ldots\}$	$\frac{\lambda^k!}{k!} e^{-\lambda}$	λ	λ

$$\mathrm{Var}[X] = \mathbb{E}\left[(X-\mu)^2\right] = \mathbb{E}\left[X^2\right] - \mu^2$$

As Var[X] is the expectation of the *square* of the deviation from the mean, the square root of Var[X] is a better measure for the deviation from the mean. We call $\sigma = \sqrt{\mathrm{Var}[X]}$ the *standard deviation* of X.

Table 1, lists some well-known *discrete* probability distributions. For example, a random variable X having a Bernoulli distribution with parameter p has two possible values: 0 (no success) and 1 (success). Parameter p models the probability of success. Hence, $\mathbb{P}[X = 1] = p.\mathbb{E}[X] = p$ and $\mathrm{Var}[X] = p(1-p)$.

Table 2 lists some *continuous* distributions. Unlike discrete distributions, the probability of a specific value is zero, i.e., $\mathbb{P}[X = k] = 0$ for any k. Therefore, the probability density function f_X (k) is used to describe the likelihood of different values. Consider for example a random variable X *uniformly distributed* on the interval $[a,b]$. $f_X(k) = \frac{1}{b-a}$, i.e., all values on the interval have the same likelihood. $\mathbb{E}[X] = \frac{a+b}{2}$ and $\mathrm{Var}[X] = \frac{(b-a)^2}{12}$.

Arrival processes are often modeled using the *negative-exponential distribution*. Parameter λ is called the *intensity* of the arrival process, i.e., λ is the expected number of new arrivals per time unit. Negative-exponentially distributed random variable X models the time in-between two subsequent arrivals. $\mathbb{E}[X] = \frac{1}{\lambda}$ is the expected average time between two such arrivals. If there is a large population of potential cases (e.g., customers) that behave independently, then, by definition, the inter-arrival times are distributed negative exponentially. This is referred to as a *Poisson arrival process*.

Durations are often modeled using the *normal* or *beta* distribution. The well-known normal distribution has two parameters: μ (mean value) and σ (standard deviation). If we use a normally distributed random variable for modeling time durations, like processing times, response times or transport times, we must be aware that this random variable can also take on *negative* values. In general negative durations are impossible; this may even cause a failure of the simulation software. To circumvent this problem, we might take a new sample whenever the

Table 2 Continuous random distributions

Distribution	Domain	$f_X(x)$	$\mathbb{E}[X]$	Var$[X]$
Uniform $a < b$	$a \leq x \leq b$	$\frac{1}{b-a}$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
Exponential $\lambda > 0$	$x \geq 0$	$\lambda e^{-\lambda x}$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
Normal $\mu \in \mathbb{R}$ $\sigma > 0$	$x \in \mathbb{R}$	$\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$	μ	σ^2
Gamma $r, \lambda > 0$	$x > 0$	$\frac{\lambda(\lambda x)^{r-1} e^{-\lambda x}}{\Gamma(r)}$	$\frac{r}{\lambda}$	$\frac{r}{\lambda^2}$
Erlang $\lambda > 0$ $r \in \{1,2,\ldots\}$	$x > 0$	$\frac{\lambda(\lambda x)^{r-1} e^{-\lambda x}}{(r-1)!}$	$\frac{r}{\lambda}$	$\frac{r}{\lambda^2}$
$\chi 2$ $v \in \{1,2,\ldots\}$	$x > 0$	See Gamma $r = \frac{v}{2}$ and $\lambda = \frac{1}{2}$	v	$2v$
Beta $a < b$ $r, s > 0$	$a \leq x \leq b$	$\frac{1}{b-a} \frac{\Gamma(r+s)}{\Gamma(r)\Gamma(s)} \left(\frac{x-a}{b-a}\right)^{r-1} \left(\frac{b-x}{b-a}\right)^{s-1}$	$a + (b-a)\frac{r}{b-a}$	$\frac{rs(b-a)^2}{(r+s)^2(r+s+1)}$

given sample produces a negative value. Note that this will affect the mean and the variance. Therefore, this solution is recommended only if the probability of a negative value is very small. We use the following rule of thumb: if $\mu - 2\sigma < 0$, the normal distribution should not be used to model durations. The normal distribution with parameters $\mu = 0$ and $\sigma = 1$ is called the *standard normal distribution*.

Like the uniform distribution, the *beta distribution* is distributed over a finite interval. We use it for random variables having a clear upper and lower bound. The beta distribution has four parameters a, b, r and s. The parameters a and b represent the upper and lower bounds of the distribution. The parameters r ($r > 0$) and s ($s > 0$) determine the shape of the distribution. Very different shapes of the probability density function are possible, see (van der Aalst and Voorhoeve 2000) for examples.

It is impossible to describe all frequently used probability distributions here. Probability distributions often used for simulation are described in detail in (van der Aalst and Voorhoeve 2000). Also consult standard textbooks on probability theory and simulation (Altiok and Melamed 2007; Kleijnen and van Groenendaal 1992; Law and Kelton 1982; Pidd 1989; Ripley 2006; Ross 1990). These references also explain how particular random variables can be constructed from pseudo-random numbers. For example, if X_i is a pseudo random number from the set $\{0, 1, \ldots, m-1\}$, then $-ln\left(\frac{X_i}{m}\right)/\lambda$ is a sample from a negative-exponential distribution with parameter λ.

4 Processing the Results

In Sect. 2.1 we described the typical ingredients of a simulation model. Simulation models abstract from details that cannot be fully modeled (e.g., perfectly modeling human decision making and customer behavior) or that are too specify (e.g., data entered into a form). Such abstractions may necessitate the introduction of stochastic elements in the model. For example, a path is selected with a certain probability and the duration of an activity is sampled from some continuous probability distribution. In Sect. 3 we showed that pseudo random numbers can be used to introduce such stochastic elements. This section focuses on the interpretation of the raw simulation results. In particular, we will show that subruns are needed to compute confidence intervals for KPIs.

During simulation there are repeated *observations* of quantities, such as waiting times, flow times, processing times, or stock levels. These observations provide information on KPIs (cf. Sect. 2.1). Suppose we have k consecutive observations x_1, $x_2,\ldots,x_k$ also referred to as *random sample*. The mean of a number of observations is the *sample mean*. We represent the sample mean of observations $x_1,x_2,\ldots,x_k$ by $\bar{x}$. We can calculate the sample mean $\bar{x}$ by adding the observations and dividing the sum by k:

$$\bar{x}=\frac{\sum_{i=1}^{k} x_i}{k}$$

The sample mean is merely an estimate of the true mean. However, it is a so-called unbiased estimator (i.e., the difference between this estimator's expected value and the true value is zero). The variance of a number of observations is the *sample variance*. This variance is a measure for the deviation from the mean. The smaller the variance, the closer the observations will be to the mean. We can calculate the sample variance s^2 by using the following formula:

$$s^2=\frac{\sum_{i=1}^{k} (x_i-\bar{x})^2}{k-1}.$$

This is the unbiased estimator of the population variance, meaning that its expected value is equal to the true variance of the sampled random variable.

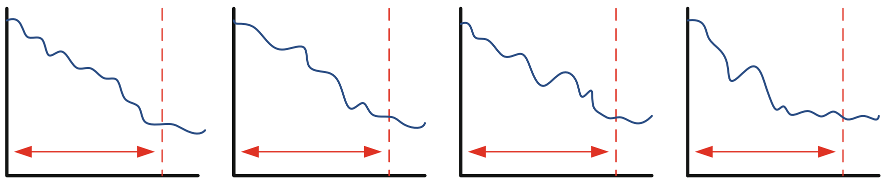
In a simulation experiment, we can determine the sample mean and the sample variance of a certain quantity. We can use the sample mean as an estimate for the real expected value of this quantity (e.g., waiting time), but *we cannot determine how reliable this estimate is*. The sample variance is not a good indicator for the reliability for the results. Consider for example the sample $\bar{x}_a$ and sample variance s_a^2 obtained from a long simulation run. We want to use $\bar{x}_a$ as a predictor for some performance indicator (e.g., waiting time). If we make the simulation experiment ten times as long, we will obtain new values for the sample mean and the sample variance, say, $\bar{x}_b$ and s_b^2, but these values do not need to be significantly different from the previous values. Although it is reasonable to assume that $\bar{x}_b$ is a more reliable predictor than $\bar{x}_a$, the sample variance will not show this. Actually, s_b^2 may be greater than s_a^2. This is the reason to introduce *subruns*.

If we have *n independent subruns*, then we can estimate the reliability of estimated performance indicators. There are two approaches to create independent subruns. The first approach is to take one long simulation run and cut this run into smaller subruns. This means that subrun $i + 1$ starts in the state left by subrun i. As the subruns need to be independent, the initial state of a subrun should not strongly correlate with the final state passed on to the next subrun. An advantage is that startup effects only play a role in the first run. Hence, by inserting a single start run at the beginning (also referred to as "warm-up period"), we can avoid incorrect conclusions due to start-up effects. The second approach is to simply restart the simulation experiment n times. As a result, the subruns are by definition independent. A drawback is that start-up effects can play a role in every individual subrun. Hence, one may need to remove the warm-up period in all subruns.

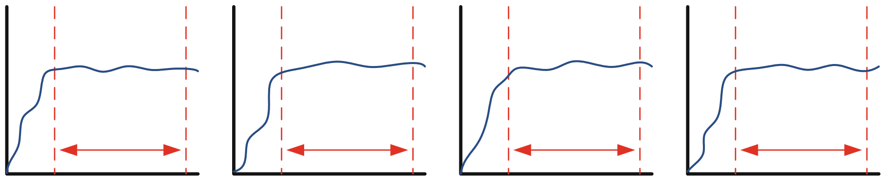
There are two types of behavior that are considered when conducting simulation experiments: *steady-state* behavior and *transient* behavior. When analyzing the steady-state behavior, we are interested in long-term effects. For example, we may consider two process designs and analyze the differences with respect to average flow times and costs in the next 5 years. When analyzing the transient behavior, we are interested in short-term effects. For example, if there are currently many backorders, we may want to know how many additional resources we need to temporarily deploy to handle these orders. When analyzing transient behavior, we are not interested in long-time averages given some stable situation but in the short-term effects. If we investigate steady-state behavior, the simulation runs need to be long and we may want to discard the initial part of the simulation. When analyzing transient behavior, the simulation runs are short and the initial part is most relevant. Figure 5 illustrates the difference between steady-state and transient analysis. Moreover, Fig. 5c shows that one simulation run can be partitioned into subruns (provided that the state at the beginning of subrun $i + 1$ does not depend on the state at the beginning of subrun i). In the remainder of this section, we concentrate on the steady-state behavior and assume that warm-up periods have been removed. Note that for each of the three situations sketched in Fig. 5, we obtain a set of independent subruns (in this case four subruns) with corresponding measurements.

Suppose we have executed n subruns and measured a result y_i for each subrun i. Hence, each result y_i serves as an estimate for a performance indicator. We assume that there exists a "true" value μ that each result y_i approximates. We want to derive assertions about μ from the values y_i. For example, y_i is the mean waiting time measured in subrun i and μ the "true" mean waiting time that we would find by conducting a hypothetical simulation experiment of infinite length. Also KPIs other than the mean waiting time could be considered, e.g., y_i could be an estimate for the mean variance of the waiting time, the mean occupation rate of a server, or the mean length of a queue. However, we must be certain that the values y_i are mutually independent for all subruns. This can be ensured by choosing a long enough subrun length or by using independent subruns. Given the results $y_1, y_2, \ldots, y_n$, we derive the sample mean:

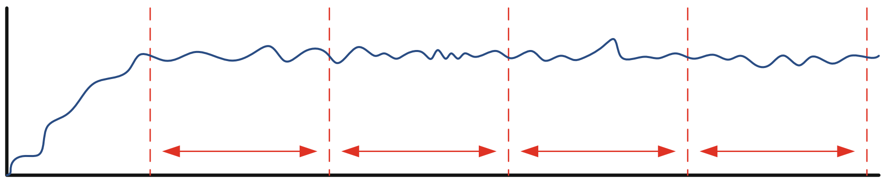
$$\overline{y} = \frac{\sum_{i=1}^{n} y_i}{n}$$



(a) transient analysis (no warm-up period, initial state matters, bounded time frame)



(b) steady-state analysis (separate runs each with warm-up period)



(c) steady-statean alysis (long run with one warm-up period split into smaller subruns)

Fig. 5 For transient analysis, the initial state and the first part of the simulation are relevant. For steady-state analysis, the initial state and warm-up period are irrelevant and only the behavior after the warm-up period matters. Each graph shows one simulation run. The X-axis denotes time whereas the Y-axis represents the state of the process. For steady-state analysis one can take separate simulation runs (each with a warm-up period) or one large simulation run cut into smaller subruns

and the sample variance:

$$s_y^2 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{n-1}.$$

The sample standard deviation is $s_y = \sqrt{s_y^2}$. The sample mean and the sample variance for the results of the subruns should not be confused with the mean and the variance of a number of measures *within* one subrun. We can consider the sample $\bar{y}$ as an estimate of the true value μ. Value y can be seen as a sample from a random variable $\overline{Y} = (X_1 + X_2 + \ldots + X_n)/n$, the *estimator*. Now $\frac{s_y}{\sqrt{n}}$ is an indication of the reliability of the estimate $\bar{y}$. If $\frac{s_y}{\sqrt{n}}$ is small, it is a good estimate.

If there is a large number of subruns, we can consider the estimator $\overline{Y}$ as normally distributed. Here we use the well-known *central limit theorem*. For a set X_1,X_2,. . .,

Table 3 $\mathbb{P}[Z > z(x)] = x$ where Z is standard normally distributed

x	$z(x)$
0.001	3.090
0.005	2.576
0.010	2.326
0.050	1.645
0.100	1.282

X_n of independent uniformly distributed random variables with expectation μ and variance σ^2, the random variable

$$\frac{(X_1 + X_2 + \ldots + X_n) - n\mu}{\sigma\sqrt{n}}$$

converges for $n \to \infty$ to a standard normal distribution. Thus, the sum or average of a large number of independent random variables is approximately normally distributed. If the subrun results are indeed independent and there are plenty of such results, we can assume that the estimator $\overline{Y}$ is normally distributed. Therefore, we treat the situation with over 30 subruns as a special case.

Given a large number of independent subruns (say, $n \geq 30$), we can easily determine a *confidence interval* for the quantity to be studied. Because the sample mean $\overline{y}$ is the average of a large number of independent measures, we can assume that $\overline{y}$ is approximately normally distributed. From this fact, we deduce the probability that the true value μ lies within a confidence interval. Given the sample mean $\overline{y}$ and the sample standard deviation s_y, the true value μ conforms with confidence $(1-\alpha)$ to the following equation:

$$\overline{y} - \frac{s_y}{\sqrt{n}} z\left(\frac{\alpha}{2}\right) < \mu < \overline{y} + \frac{s_y}{\sqrt{n}} z\left(\frac{\alpha}{2}\right)$$

where $z\left(\frac{\alpha}{2}\right)$ is defined as follows: If Z is a standard normally distributed random variable, then the probability that random variable Z is greater than $z(x)$ is x. Table 3 shows for five values of x the value $z(x)$. The value α represents the unreliability; that is, the probability that μ does not conform to the equation. Typical values for α range from 0.001 to 0.100. The interval

$$\left[\overline{y} - \frac{s_y}{\sqrt{n}} z\left(\frac{\alpha}{2}\right), \overline{y} + \frac{s_y}{\sqrt{n}} z\left(\frac{\alpha}{2}\right)\right]$$

is known as the $(1-\alpha)$-*confidence interval* for the estimated value μ.

Given a smaller number of independent subruns (say, $n \leq 30$), we need to make more assumptions about the distribution of the individual subrun results. A common assumption is that the individual subrun results are normally distributed. This is a realistic assumption when the subrun result itself is calculated by taking the average over a large set of independent measurements (see the central limit

Table 4 The critical values for a student's t-distribution with v degrees of freedom

	$x =$			
$t_v(x)$	0.100	0.050	0.010	0.001
$v = 1$	3.08	6.31	31.82	318.31
2	1.89	2.92	6.96	22.33
3	1.64	2.35	4.54	10.21
4	1.53	2.13	3.75	7.17
5	1.48	2.02	3.37	5.89
6	1.44	1.94	3.14	5.21
7	1.41	1.89	3.00	4.79
8	1.40	1.86	2.90	4.50
9	1.38	1.83	2.82	4.30
10	1.37	1.81	2.76	4.14
15	1.34	1.75	2.60	3.73
20	1.33	1.72	2.53	3.55
25	1.32	1.71	2.49	3.45
50	1.30	1.68	2.40	3.26
100	1.29	1.66	2.35	3.17
∞	1.28	1.64	2.33	3.09

theorem, which states that as the sample size increases the distribution of the sample average of these random variables approaches the normal distribution irrespective of the shape of the common distribution of the individual terms). By using this assumption, we can deduce—given n subruns with a sample mean $\overline{y}$, sample deviation s_y, and reliability $(1-\alpha)$—the following confidence interval:

$$\left[\overline{y} - \frac{s_y}{\sqrt{n}} t_{n-1}\left(\frac{\alpha}{2}\right), \overline{y} + \frac{s_y}{\sqrt{n}} t_{n-1}\left(\frac{\alpha}{2}\right)\right]$$

where $t_v(x)$ is the critical value of a *Student's t-distribution* with v degrees of freedom. Table 4 shows for several values of v and x the critical value $t_v(x)$.

Contrary to the method discussed earlier, we can now also determine the confidence interval if only a limited number of subruns (say, ten) is at our disposal. For small numbers v, we have $t_v(x) > z(x)$. As v increases, the value of $t_v(x)$ decreases and in the limit we obtain $t_v(x) = z(x)$.

When two confidence intervals are overlapping for a KPI, one cannot make any firm statements about the superiority of one the corresponding alternatives. Moreover, one alternative may score better with respect to costs whereas the other alternative may reduce flow times significantly.

Using the above, we can compute confidence intervals for any KPI. If the confidence intervals are too wide, more subruns or longer subruns can be used to obtain tighter confidence intervals. As mentioned before, simulation is an excellent tool for "what if" analysis. Confidence intervals can be computed for different KPIs and different alternatives. Alternatives can be created by varying parameters or by making changes in the design.

5 Pitfalls to Avoid

Simulation is a powerful and flexible tool that can be used to support decision making. If simulation is applied incorrectly (flawed model or poor analysis of the results), then this may result in incorrect decisions that are very costly. Therefore, we point out 15 *typical pitfalls of simulation* that should be avoided. In Sect. 5.1 we present ten general risks that may result in incorrect conclusions and misleading insights. These are linked to the different phases of a simulation study (cf. Fig. 6). Section 5.2 identifies five more specific risks caused by simulation models that do not incorporate essential phenomena such as working speeds depending on workloads, partial availability of resources, and competition among activities in different processes.

5.1 General Risks

In Sect. 2.2 we described the different phases of a traditional simulation study. Figure 6 lists ten risks pointing to typical errors (pitfalls) frequently made when applying simulation. These are described in the remainder.

5.1.1 Risk 1: One-Sided Problem Definition

A simulation study gets off on the wrong foot if the problem definition is drawn up exclusively by either the user or the systems analyst. The user may possess extensive knowledge of the problem area, but lacks the experience needed for defining his problem. The systems analyst on the other hand, fully knows the elements which should be present in a problem definition, but lacks the background

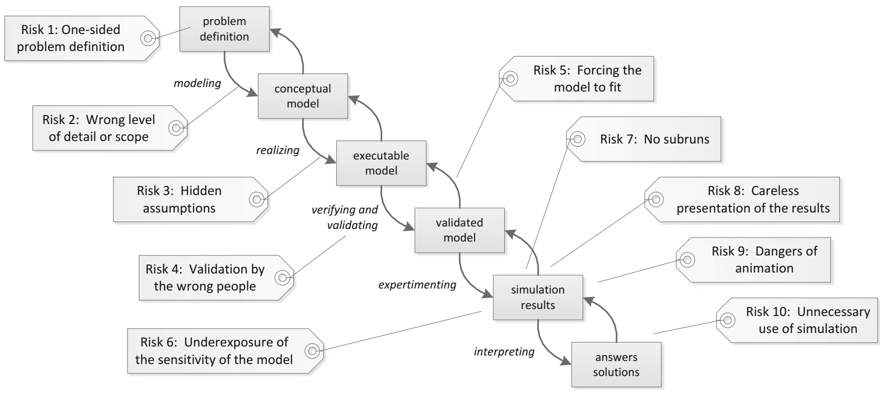

Risk 1: One-sided problem definition
problem definition
modeling
conceptual model
Risk 2: Wrong level of detail or scope
realizing
executable model
Risk 3: Hidden assumptions
verifying and validating
validated model
Risk 4: Validation by the wrong people
experimenting
simulation results
Risk 6: Underexposure of the sensitivity of the model
interpreting
answers solutions
Risk 5: Forcing the model to fit
Risk 7: No subruns
Risk 8: Careless presentation of the results
Risk 9: Dangers of animation
Risk 10: Unnecessary use of simulation

Fig. 6 Various risks associated to the different phases of a simulation study

of the specific problem. The systems analyst is also aware of the possibilities and impossibilities of simulation. The user on the other hand, generally knowing little about simulation, is barely informed on this issue. Therefore, for a simulation study to be successful, it is important that both parties closely cooperate in setting up the problem definition. The problem definition serves as a "contract" between the user and the builder of the model. Hence, the following *rule of thumb* should be used: "Do not start a simulation study until it is clear to both user(s) and analyst(s) which questions need to be answered!".

5.1.2 Risk 2: Wrong Level of Detail or Scope

In making a simulation model, one chooses a certain *level of detail*. In a simulation model for a manufacturing department, a machine may be modeled as an object with a mean service time as its only parameter. Alternatively, it can be modeled in detail, taking into account aspects such as set-up times, faults, tool-loading, maintenance intervals etc. Many simulation studies end prematurely because a wrong level of detail is selected initially. Too much detail causes the model to become unnecessarily complex and introduces extra parameters that need to be assessed (with all the risks involved). Too many abstractions can lead to a simulation model that leaves the essential questions of the problem definition unanswered. The right level of detail is chosen if:

1. Information is present that allows experiments with the model,
2. The important questions from the problem definition are addressed by the model, and
3. The complexity of the model is still manageable for all parties concerned.

If it is impossible to choose a suitable level of detail satisfying these three conditions, the problem definition needs to be adjusted.

Related to the level of detail is the *scope* of the model. When analyzing a process handled within a department, one can also model the other processes within the same department competing for the same resources and the other departments interacting with the process. One can think of the scope as the "breadth" of the model whereas the level of detail is the model's "depth". Broadening the scope or increasing the level of detail may lead to more accurate models. However, more detail or a broader scope may result in increased modeling and data gathering efforts. In fact, sometimes there is no data to support a more refined model. This is why probability distributions are used.

The well-known "80/20-rule" also applies to simulation models: 80 % of the model's accuracy is obtained from 20 % of the model's detail. Hence, a small increase in accuracy may require the addition of lots of details. Hence, the following *rule of thumb* should be used: "Minimize the breadth and depth of a model given a set of predefined questions and required level of accuracy".

5.1.3 Risk 3: Hidden Assumptions

During modeling and while realizing an executable simulation model, many assumptions must be made. Assumptions are made to fill gaps in an incomplete problem definition or because of a conscious decision to keep the simulation model simple. Often these assumptions are documented poorly, if documented at all. These hidden assumptions may lead to the rejection of the simulation model during validation or later. Hidden assumptions may also lead to invalid conclusions and bad decisions. Therefore, *all* assumptions must be documented and regularly discussed with the user.

5.1.4 Risk 4: Validation by the Wrong People

Sometimes, due to time pressure or indifference of the user, the simulation model is only validated by its maker(s). Discrepancies between the model and the ideas of the user may thus be discovered too late, if at all. Therefore, the user should be involved in the validation of the simulation model before any experiments are conducted.

5.1.5 Risk 5: Forcing the Model to Fit

In the validation phase, often the results of the simulation model do not match the observed or recorded actual data. One is then tempted to make the model "fit" by changing certain parameter values, i.e., the analyst fiddles around with the parameter settings until a match is found. This, however, is very dangerous, since this match with reality is most likely caused by sheer luck and not by a model that adequately reflects reality. Parameters should be adjusted only after having understood why the model deviates from reality. This prevents the conscious or unconscious obscuring of errors in the model.

5.1.6 Risk 6: Underexposure of the Sensitivity of the Model

Certain model parameters (e.g. the intensity of the arrival process) are often set at one specific value. The chosen parameter settings should be justifiable. However, even if this is the case, small variations in the arrival process can have dramatic effects.

Consider for example the $M/M/1$ queue describing the situation with a Poisson arrival process (the inter-arrival times are distributed negative exponentially), negative-exponentially distributed service times and one server (i.e., at most one customer is served at a time). Assuming an arrival rate λ (average number of customers arriving per time unit) and service rate μ (average number of customers that can be handled per time unit), the average flow time is $\frac{1}{\mu-\lambda}$. If $\lambda = 98$ (on average 98 customers arrive per day) and $\mu = 100$ (the average service time is approximately 14 min), then the average flow time is $\frac{1}{100-98} = 0.5$ (12 h). If λ increases to 99 (an increase of approximately 1 %), then the average flow time doubles to $\frac{1}{100-99} = 1$, i.e., a full day. The example illustrates that a small increase in workload may have dramatic effects on the mean flow or waiting time. Therefore, the sensitivity of the model to minor adjustments of its parameters should be seriously accounted for.

5.1.7 Risk 7: No Subruns

Some people say: "A sufficiently long simulation yields correct results!" They execute a simulation run for a night or weekend and then blindly trust, e.g., the mean waiting time measured. This is a very risky practice, as no assertions about the reliability of the result can be given. Others derive a confidence interval from the mean variance measured. This is also wrong because, for example, the mean variance of the waiting time measured is unrelated to the reliability of the estimated mean waiting time. The only way to derive independent measurements is by having independent subruns!

5.1.8 Risk 8: Careless Presentation of the Results

Interpreting the results of a simulation study may require complex statistical analyses. This is often a source of errors. Translating the results from statistics into language a user can understand, can be very tricky indeed. In Darrel Huff's book "How to lie with statistics" (Huff 1954), there are numerous examples of sloppy and misleading presentations. As an example, suppose the final report of a simulation study contains the following conclusion "Waiting times will be reduced by 10 %". This conclusion is very incomplete, as it contains no reference whatsoever to its reliability. It is good practice to give a confidence interval. The same conclusion suggests that waiting times will be reduced by 10 % for each customer. This, however, may not be the case. The average waiting time may be reduced by 10 % while it increases for certain customers and is reduced somewhat more for others.

5.1.9 Risk 9: Dangers of Animation

Modern simulation tools allow for impressive visualizations of simulation results. Animation facilities graphically show the process while it is unfolding. These facilities improve communication with the user. However, there is an inherent danger in animation. As animation only shows the tangible aspects of the simulation model, the user may develop an unfounded faith in the model. The choice of parameters or decision making rules deeply influence the simulation results, yet are barely visible in an animation. The same hold for the presentation of simulation results. Impressive 3D charts do not replace a sound statistical analysis.

5.1.10 Risk 10: Unnecessary Use of Simulation

Simulation is a flexible analysis tool that can be applied in almost any business context. Therefore, one may be tempted to use it regardless of the circumstances. Often, however, a simple mathematical model (e.g. a queuing model) or a simple spreadsheet calculation is sufficient. In such cases simulation is “overkill”. It should only be used if and when the situation requires it. Simulation is a means and not a goal!

5.2 Specific Risks

The ten risks highlighted in Fig. 6 cover the different phases of a simulation project. Besides these general risks there are more specific risks related to not incorporating relevant contextual factors (that may be changing over time) and not capturing characteristics of human resources (working patterns, partial availability, and varying working speeds). For example, human resources are typically modeled in a rather naïve manner. As a result, it is not uncommon that the simulated model predicts flow times of minutes or hours while in reality flow times are weeks or even months (van der Aalst et al. 2014).

5.2.1 Risk 11: Abstracting Away Relevant Contextual Factors

Processes unfold in a particular context (Rosemann et al. 2008) that is often neglected in simulation studies. Not capturing this context may result in simulation models with limited predictive value. To explain the notion of “context” consider Fig. 7 (taken from (van der Aalst and Dustdar 2012)). In (van der Aalst and Dustdar 2012) four levels of context data are considered:

- *Instance Context*. Process instances (that is, cases) might have various properties that influence their execution. Consider the way businesses handle a customer order. The type of customer placing the order can influence the path the instance follows in the process. The order’s size can influence the type of shipping the customer selects or the transportation time. These properties can directly relate to the individual process instance; we refer to them as the instance context. Typically, discovering relationships between the instance context and the case’s observed behavior is not difficult. We might, for example, discover that an activity is typically skipped for VIP customers.

- *Process Context.* A process might be instantiated many times—for example, the process can handle thousands of customer orders per year. Yet, the corresponding process model typically describes one order's life cycle in isolation. Although interactions among instances are not very explicit in most simulation models, they can influence each other. Instances might compete for the same resources, and an order might be delayed by too much work-in-progress. Looking at one instance in isolation is not sufficient for understanding the real behavior. Simulation models should also consider the process context, such as the number of instances being handled and resources available for the process. When analyzing the flow time of cases, the simulation model should consider not only the order's status (instance context) but also the workload and resource availability (process context).
- *Social Context.* The process context considers all factors directly related to a process and its instances. However, people and organizations typically are not allocated to a single process and might be involved in many different processes. Moreover, activities are executed by people operating in a social network. Friction between individuals can delay process instances, and the speed at which people work might vary due to circumstances that are not fully attributable to the process being analyzed (see also Risk 14). We refer to all these factors as the social context, which characterizes how people work together within a particular organization. Today's simulation tools tend to neglect the social context even though it directly impacts how people and organizations handle cases.
- *External Context.* The external context captures factors that are part of an ecosystem that extends beyond an organization's control sphere. For example, the weather, the economic climate, and changing regulations might influence how organizations handle cases. The weather might influence the workload, as when a storm or flooding leads to increased insurance claims. Changing oil prices can influence customer orders, as when the demand for heating oil increases as prices drop. More stringent identity checks influence the order in which a government organization executes social-security-related activities. Although external context can have a dramatic impact on the process being analyzed, selecting relevant variables is difficult. Learning the external context's effects is closely related to identifying concept drift (see also Risk 12)—for example, a process might gradually change due to external seasonal effects.

Simulation models tend to focus on the first two levels of the "union model" depicted in Fig. 7. This may be valid in many studies. However, if the social context and external context matter, they should be incorporated explicitly.

5.2.2 Risk 12: Ignoring Concept Drift

The term *concept drift* refers to a situation in which the process is changing while being analyzed (Jagadeesh Chandra Bose et al. 2011; Widmer and Kubat 1996).

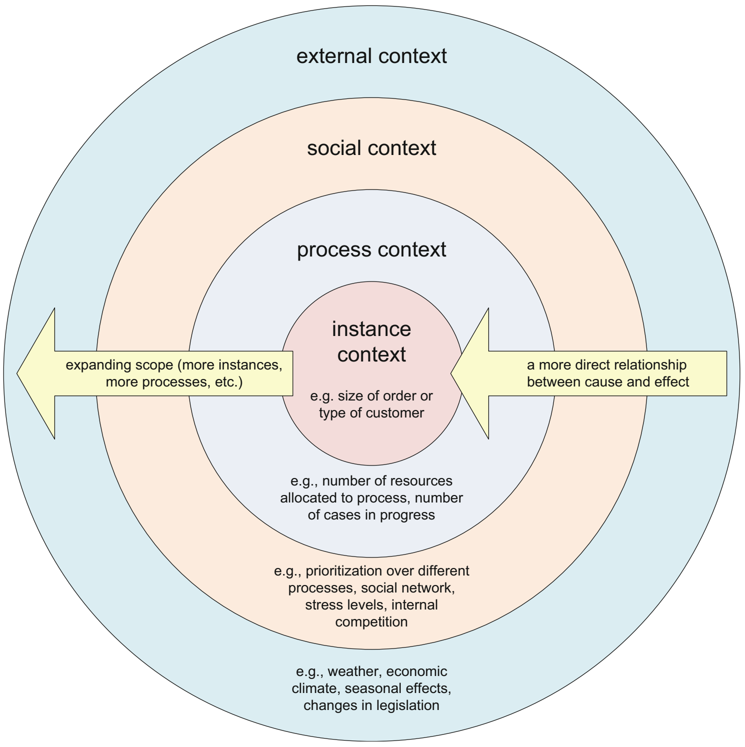

external context
social context
process context
instance context
e.g. size of order or type of customer
expanding scope (more instances, more processes, etc.)
a more direct relationship between cause and effect
e.g., number of resources allocated to process, number of cases in progress
e.g., prioritization over different processes, social network, stress levels, internal competition
e.g., weather, economic climate, seasonal effects, changes in legislation

Fig. 7 Levels of context data. Context can influence processes and may change over time. Nevertheless, simulation models seldom explicitly model the outer two context levels and do not anticipate context changes

Processes can change due to periodic or seasonal changes ("in December, there is more demand" or "on Friday afternoon, fewer employees are available") or to changing conditions ("the market is getting more competitive"). Such changes affect processes, and organizations must detect and analyze them. The notion of concept drift is closely related to the context notion illustrated in Fig. 7. Large parts of the context cannot be fully controlled by the organization conducting a simulation study. Therefore, contextual variability needs to be considered and cannot be ignored.

Predictable drifts (e.g., seasonal influences) with a significant influence on the process need to be incorporated in simulation models. For unpredictable drifts (e.g., changing economic conditions), several "what if" scenarios need to be explored.

5.2.3 Risk 13: Ignoring That People Are Involved in Multiple Processes

In practice there are few people that only perform activities for a single process. Often people are involved in many different processes, e.g., a manager, doctor, or

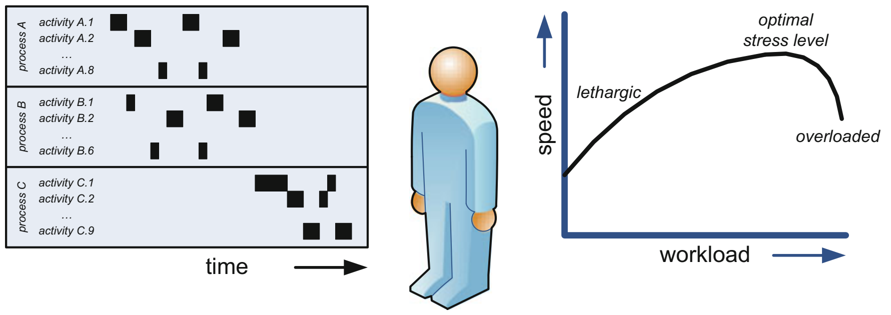


Fig. 8 People are typically involved in multiple processes and need to distribute attention over these processes and related activities (*left*). Moreover, people do not work at constant speed (*right*). The "Yerkes-Dodson Law of Arousal" (Yerkes and Dodson 1908) describes the phenomenon that people work at different speeds based on their workload

specialist may perform tasks in a wide range of processes. The left-hand side of Fig. 8 shows a Gantt chart illustrating how an individual may distribute her time over activities in different processes. Simulation often focuses on a single process, often ignoring competing processes.

Suppose a manager is involved in a dozen processes and spends about 20 % of her time on the process that we want to analyze. In most simulation tools it is impossible to model that she is only available 20 % of the time. Hence, one needs to assume that the manager is there all the time and has a very low utilization. As a result the simulation results are too optimistic. In the more advanced simulation tools, one can indicate that resources are there at certain times in the week (e.g., only on Monday morning). This is also an incorrect abstraction as the manager distributes her work over the various processes based on priorities and workload. Suppose that there are 5 managers all working 20 % of their time on the process of interest. One could think that these 5 managers could be replaced by a single manager (5*20 % = 1*100 %). However, from a simulation point of view this is an incorrect abstraction. There may be times that all 5 managers are available and there may be times that none of them is available.

People are involved in multiple processes and even within a single process different activities and cases may compete for shared resources. One process may be more important than another and get priority. In some processes cases that are delayed may get priority while in other processes late cases are "sacrificed" to finish other cases in time. People need to continuously choose between work-items and set priorities. Although important, this is typically not captured by simulation models.

5.2.4 Risk 14: Assuming That People Work at Constant Speeds

Another problem is that people work at different speeds based on their workload, i.e., it is not just the distribution of attention over various processes, but also the absolute working speed that determines the resource's contribution to the process. There are various studies that suggest a relation between workload and performance of people. A well-known example is the so-called "Yerkes-Dodson Law of Arousal" (Yerkes and Dodson 1908). The Yerkes-Dodson law models the relationship between arousal and performance as a ∩-shaped curve (see right-hand side of Fig. 8). This implies that, for a given individual and a given type of task, there exists an optimal arousal level. This is the level where the performance has its maximal value. Thus work pressure is productive, up to a certain point, beyond which performance collapses. Although this phenomenon can be easily observed in daily life (Nakatumba and van der Aalst 2010), today's business process simulation tools typically do not support the modeling of workload dependent processing times.

5.2.5 Risk 15: Ignoring That People Work in Batches

As indicated earlier, people may be involved in different processes. Moreover, they may work part-time (e.g., only in the morning). In addition to their limited availabilities, people have a tendency to work in batches (cf. Resource Pattern 38: Piled Execution (Russell et al. 2005)). In any operational process, the same task typically needs to be executed for many different cases (process instances). Often people prefer to let work-items related to the same task accumulate, and then process all of these in one batch. In most simulation tools a resource is either available or not, i.e., it is assumed that a resource is eagerly waiting for work and immediately reacts to any work-item that arrives. Clearly, this does not do justice to the way people work in reality. For example, consider how and when people reply to e-mails. Some people handle e-mails one-by-one when they arrive while others process their e-mail at fixed times in batch. Related is the fact that calendars and shifts are typically ignored in simulation tools. While holidays, lunch breaks, etc. can heavily impact the performance of a process, they are typically not incorporated in the simulation model.

In (van der Aalst et al. 2014) a general approach based on "chunks" is used to model availability more adequately. The basic idea is that people spend "chunks of time" on a particular process or task. Within a period of time a limited number of chunks is available. Within a chunk, work is done in batches. As chunks become more coarse-grained, flow times go up even when the overall utilization does not change (van der Aalst et al. 2014).

6 Advanced Simulation

The 15 risks described in Sect. 5 illustrate that many things can go wrong in a simulation project. Fortunately, modern IT infrastructures and the enormous amounts of event data collected in many organizations also enable new forms of simulation. IT systems are becoming more and more intertwined with the business processes they aim to support, resulting in an "explosion" of available data that can be used for analysis purposes. Today's information systems already log enormous amounts of events and it is clear that data-based analytics like process mining (van der Aalst 2011) will become more important. Increasingly, *simulation techniques will need to incorporate actual event data*. Moreover, there will be a *shift from off-line analysis at design time to on-line analysis at run-time*.

Figures 2 and 4 present a rather classical view on business process simulation. This is the type of simulation supported by hundreds, if not thousands, of commercial simulation packages. Some vendors provide a pure simulation tool (e.g., Arena, Extend, etc.) while others embed this in a workflow management system (e.g., FileNet, COSA, etc.) or a business process modeling tool (e.g., Protos, ARIS, etc.). All of these tools use the information presented in Fig. 2 to simulate business processes and subsequently measure obvious performance indicators such as flow time, utilization, etc. Using Fig. 9, we will show that it is possible to move beyond "traditional" simulation approaches.

The left-hand-side of Fig. 9 shows the role of a process-aware information system (a WFM/BPM system or any other process-oriented information system, e.g., an ERP system like SAP) in supporting operational business processes. The information system supports, controls, and monitors operational processes. The resources within the organization perform tasks in such processes and therefore also interact with the information system. The information system can only do meaningful things if it has knowledge of the process, the resources within the organization and the current states of active cases. Moreover, today's information systems often record historical information for auditing and performance analysis. The lower four ellipses in the middle of Fig. 9 show four types of data implicitly or explicitly available when an information system is supporting an operational process: (1) real event data, (2) process state, (3) process model, and (4) resource model. An *event log* (i.e., real event data) contains historical information about "When, How, and by Whom?" in the form of recorded events. The *process state* represents all information that is attached to currently running cases, e.g., Customer order XYZ consists of 25 order lines and has been in the state "waiting for replenishment" since Monday. The process state may also contain context information relevant for the process, e.g., the weather or economic trends. The *process model* describes the ordering of tasks, routing conditions, etc. The *resource model* holds information about people, roles, departments, etc. Clearly, the process state, process model, and resource model may be used to enact the process. The event log merely records the process as it is actually enacted.

The right-hand-side of Fig. 9 focuses on analysis rather than enactment; it links the four types of data to simulation. For traditional simulation (i.e., in the sense of Figs. 2 and 4) a hand-made simulation model is needed. This simulation model can be derived from the process model used by the information system. Moreover, information about resources, arrival processes, processing times, etc. is added (cf. Fig. 2). The arcs between the box *traditional simulation* and the three types of data (real event data, process model, and resource model) are curved to illustrate

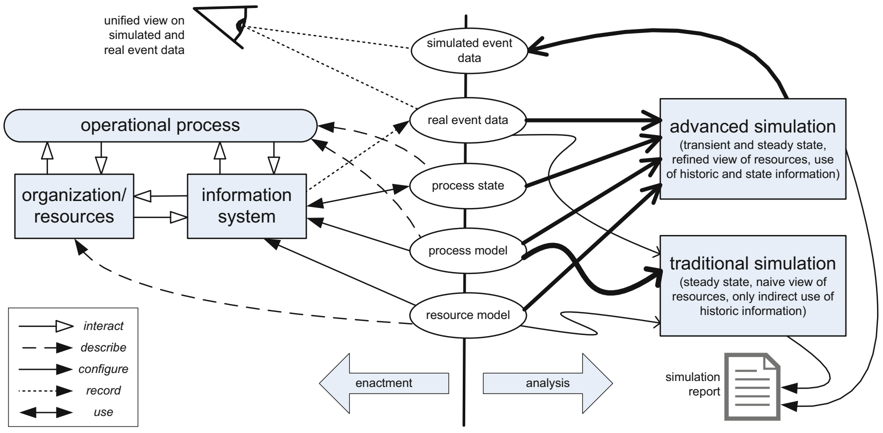


Fig. 9 Advanced simulation compared to traditional simulation. Note that real event data and simulated event data can be stored in event logs and analyzed using the same process mining tool. Due to this unified view on process behavior, simulation can be embedded in day-to-day management and decision making

that the relationship between the data used by the information system and the simulation tool is typically rather indirect. For example, the analyst cannot use the process model directly, but needs to transform it to another language or notation. The resource model used for simulation is typically rather simple compared to models that can be enacted by a WFM or BPM system. Often each activity has a single role and a fixed number of resources is available per role. Moreover, often it is assumed that these resources are available on a full-time basis. Real event data are not used directly. At best, event logs are used to estimate the parameters for some of the probability distributions. Traditional simulation models are *not* tightly coupled to the actual information and historical data and model resource behavior in a rather naïve manner. Moreover, the current state (including context information) is not used at all. As such, simulation focuses on steady-state behavior and cannot be used for operational decision making.

We advocate more advanced forms of simulation. First of all, we propose a tight coupling with the information system supporting the process that is being analyzed. Simulation should exploit event logs and process state information. Second, analysis should not only focus on steady-state behavior but also on transient behavior in order to also support operational decision making. This is illustrated by the box *advanced simulation* in Fig. 9.

Advanced simulation should exploit real event data to semi-automatically learn better simulation models. Therefore, we advocate using process mining techniques (van der Aalst 2011). Process mining exploits the information recorded in audit trails, transaction logs, databases, etc. Process mining includes (automated) process discovery (i.e., extracting process models from an event log), conformance checking (i.e., monitoring deviations by comparing model and log), social network/organizational mining, model extension, and process model repair. The automated construction of simulation models is possible by combining existing process mining techniques (Rozinat et al. 2009a).

It is essential to note that, through process mining, events in the log can be related to model elements. This allows for the projection of dynamic information onto models: the event log "breathes life" into otherwise static process models. Consider a control-flow model, e.g., the Petri net, BPMN, or EPC model shown in Fig. 1. Such a model may have been discovered or made by hand. By replaying the event log on the model, it is possible to enrich the model with frequencies, probabilities and delays (Rozinat et al. 2009a). This illustrates that the additional information described in Fig. 2 can indeed be discovered, thus resulting in a full-fledged simulation model.

Establishing a good connection between event log and model may be difficult and require several iterations. However, when using a WFM or BPM system, this connection already exists. WFM and BPM systems are driven by explicit process models and provide excellent event logs. Moreover, internally such systems also have an explicit representation of the state of each running case. This enables a new type of simulation called *short-term simulation* (van der Aalst 2011; Rozinat et al. 2009b). The key idea is to start all simulation runs from the current state and focus on transient behavior. This way a "fast forward button" into the future is provided. To understand the importance of short-term simulation, see Fig. 5 which explains the difference between transient analysis and steady-state analysis. The key idea of simulation is to execute a model repeatedly. The reason for doing the experiments repeatedly, is to not come up with just a single value (e.g., "the average response time is 10.36 min") but to provide confidence intervals (e.g., "the average response time is with 90 % certainty between 10 and 11 min"). For transient analysis the focus is on the initial part of future behavior, i.e., starting from the initial state the "near future" is explored. For transient analysis the initial state is very important. If the simulation starts in a state with long queues of work, then in the near future flow times will be long and it may take quite some time to get rid of the backlog. For steady-state analysis the initial state is irrelevant. Typically, the simulation is started "empty" (i.e., without any cases in progress) and only when the system is filled with cases measurement starts. Steady-state analysis is most relevant for answering strategic and tactical questions. Transient analysis is most relevant for operational decision making. Lion's share of contemporary simulation support aims at steady-state analysis and, hence, is limited to strategic and tactical decision making. Short-term simulation focuses on *operational decision making*; starting from the current state (provided by the information system) the "near future" is explored repeatedly. This shows what will happen if no corrective actions are taken. Moreover, "what if" analysis can be used to explore the effects of different interventions (e.g., adding resources and reconfiguring the process).

Figure 9 shows that advanced simulation uses all information available, e.g., event data to learn process characteristics, the current state to enable short-term simulation ("fast forward button"), and a more refined resource model to better capture working patterns.

Process mining techniques are driven by event logs recorded for the actual process. Similar event logs can be generated by simulation. In both cases events are described by a reference to some process instance (the case), an activity, a timestamp, a resource, and other attributes (e.g., costs). The top-most ellipse in the middle of Fig. 9 (tagged "simulated event data") refers to event logs produced by simulation rather than reality. As shown, *both simulated and real events can be viewed using the same tools*. This is very important for operational decision making and "what if" analysis. Different future scenarios can be explored using visualizations also used for past and current event data.

7 Conclusion

This chapter provides a "survival guide" to business process simulation. Besides providing a basic introduction to the topic, the chapter lists 15 risks, i.e., potential pitfalls, when using simulation. Moreover, the chapter also shows that more advanced forms of simulation come into reach as IT and business processes get more intertwined.

To conclude the chapter, we suggest books and articles for BPM academics and professionals that want to learn more about business process simulation:

- There are many (text) books on simulation, see for example (Altiok and Melamed 2007; Bratley et al. 1983; Hartmann 2009; Kelton et al. 2003; Kleijnen and van Groenendaal 1992; Law and Kelton 1982; Naylor et al. 1966; Pidd 1989; Ripley 2006; Robinson 1994; Ross 1990; Shannon 1975). Books like (Kleijnen and van Groenendaal 1992; Ripley 2006; Ross 1990) focus on the statistical aspects of simulation. Books like (Altiok and Melamed 2007; Hartmann 2009; Kelton et al. 2003; Law and Kelton 1982) focus on the creation of simulation models. The book "Successful Simulation: A Practical Approach to Simulation Projects" (Robinson 1994) is one of the few books focusing on simulation projects (including topics such as project management).
- In (Haas 2002; Ajmone Marsan et al. 1995) various techniques for the analysis of stochastic Petri nets (i.e., Petri nets extended with priorities, probabilities, and durations) are described. See (Baskett et al. 1975; Buzacott 1996; Kleinrock 1975) for some seminal papers on the analysis of processes using analytical methods.
- For more information on role of various analysis techniques (including simulation) in BPM we refer to (van der Aalst 2013; van der Aalst and Stahl 2011; Dumas et al. 2013; ter Hofstede et al. 2010; Weske 2007). See (van der Aalst 2011; Rozinat et al. 2009a) for techniques to automatically discover simulation models from event data and (Rozinat et al. 2009b) for operational decision support using simulation (e.g., short-term simulation).

This chapter is based on (van der Aalst 2010; van der Aalst and Dustdar 2012; van der Aalst et al. 2014; van der Aalst and Voorhoeve 2000): in (van der Aalst

2010) we elaborate on the relation between simulation and process mining, in (van der Aalst et al. 2014) we focus on the proper modeling of resource availability, in (van der Aalst and Dustdar 2012) we emphasize the importance of incorporating context, and in (van der Aalst and Voorhoeve 2000) we provide a tutorial on conventional business process simulation.

Acknowledgements The author would like to thank Joyce Nakatumba, Marc Voorhoeve, Anne Rozinat, Ronny Mans, Hajo Reijers, Michael Westergaard, and Mariska Netjes for their joint work on business process simulation. This work was supported by the Basic Research Program of the National Research University Higher School of Economics (HSE).

References

Ajmone Marsan M, Balbo G, Conte G, Donatelli S, Franceschinis G (1995) Modelling with generalized stochastic Petri nets, Wiley series in parallel computing. Wiley, New York

Altiok T, Melamed B (2007) Simulation modeling and analysis with Arena. Elsevier Academic Press, Amsterdam

Baskett F, Chandy KM, Muntz RR, Palacios FG (1975) Open, closed and mixed networks of queues with different classes of customers. J Assoc Comput Mach 22(2):248–260

Bratley P, Fox BL, Schrage LE (1983) A guide to simulation. Springer, Berlin

Buzacott JA (1996) Commonalities in reengineered business processes: Models and issues. Manage Sci 42(5):768–782

Dahl OJ, Nygaard K (1966) SIMULA: an ALGOL based simulation language. Commun ACM 1:671–678

Dumas M, La Rosa M, Mendling J, Reijers H (2013) Fundamentals of business process management. Springer, Berlin

Haas PJ (2002) Stochastic Petri nets: modelling, stability, simulation, Springer series in operations research. Springer, Berlin

Hartmann AK (2009) Practical guide to computer simulations. World Scientific, Singapore

Huff D (1954) How to lie with statistics. Penguin Books, New York

Jagadeesh Chandra Bose JP, van der Aalst WMP, Zliobaite I, Pechenizkiy M (2011) Handling concept drift in process mining. In: Mouratidis H, Rolland C (eds) International conference on advanced information systems engineering (Caise 2011). Lecture notes in computer science, vol 6741. Springer, Berlin, pp 391–405

Kelton DW, Sadowski R, Sturrock D (2003) Simulation with Arena. McGraw-Hill, New York

Kleijnen J, van Groenendaal W (1992) Simulation: a statistical perspective. Wiley, New York

Kleinrock L (1975) Queueing systems, vol 1, Theory. Wiley, London

Law AM, Kelton DW (1982) Simulation modeling and analysis. McGraw-Hill, New York

Nakatumba J, van der Aalst WMP (2010) Analyzing resource behavior using process mining. In: Rinderle-Ma S, Sadiq S, Leymann F (eds) BPM 2009 workshops, Proceedings of the fifth workshop on Business Process Intelligence (BPI'09). Lecture notes in business information processing, vol 43. Springer, Berlin, pp 69–80

Naylor TH, Balintfy JL, Burdick DS, Kong Chu (1966) Computer simulation techniques. Wiley, New York

Pidd M (1989) Computer modelling for discrete simulation. Wiley, New York

Ripley B (2006) Stochastic simulation. Wiley, Hoboken

Robinson S (1994) Successful simulation: a practical approach to simulation projects. McGraw-Hill, Maidenhead

Rosemann M, vom Brocke J (2014) The six core elements of business process management. In: vom Brocke J, Rosemann M (eds) Handbook on business process management, vol 1, 2nd edn. Springer, Heidelberg, pp 105–122

Rosemann M, Recker J, Flender C (2008) Contextualisation of business processes. Int J Bus Process Integr Manage 3(1):47–60

Ross SM (1990) A course in simulation. Macmillan, New York

Rozinat A, Mans RS, Song M, van der Aalst WMP (2009a) Discovering simulation models. Inform Syst 34(3):305–327

Rozinat A, Wynn M, van der Aalst WMP, ter Hofstede AHM, Fidge C (2009b) Workflow simulation for operational decision support. Data Knowl Eng 68(9):834–850

Russell N, van der Aalst WMP, ter Hofstede AHM, Edmond D (2005) Workflow resource patterns: identification, representation and tool support. In: Pastor O, Falcao e Cunha J (eds) Proceedings of the 17th conference on advanced information systems engineering (CAiSE'05), Lecture notes in computer science, vol 3520. Springer, Berlin, pp 216–232

Shannon RE (1975) Systems simulation: the art and science. Prentice-Hall, Englewood Cliffs

ter Hofstede AHM, van der Aalst WMP, Adams M, Russell N (2010) Modern business process automation: YAWL and its support environment. Springer, Berlin

van der Aalst WMP (2010) Business process simulation revisited. In Barjis J (ed) Enterprise and organizational modeling and simulation, Lecture notes in business information processing, vol 63. Springer, Berlin, pp 1–14

van der Aalst WMP (2011) Process mining: discovery, conformance and enhancement of business processes. Springer, Berlin

van der Aalst WMP (2013) Business process management: a comprehensive survey. ISRN Software Engineering. doi:10.1155/2013/507984, pp 1–37

van der Aalst WMP, Dustdar S (2012) Process mining put into context. IEEE Internet Comput 16 (1):82–86

van der Aalst WMP, Voorhoeve M (2000) Simulation handbook. BPM Center Report BPM00-04, BPMcenter.org, 2000. (Original Dutch version appeared as "W.M.P. van der Aalst. Handboek Simulatie. Computing science reports 95/32, Eindhoven University of Technology, Eindhoven, 1995")

van der Aalst WMP, Stahl C (2011) Modeling business processes: a Petri net oriented approach. MIT Press, Cambridge, MA

van der Aalst WMP (2014) Business process simulation survival guide. In: vom Brocke J, Rosemann M (eds) Handbook on business process management, vol 1, 2nd edn. Springer, Heidelberg, pp 337–370

Weske M (2007) Business process management: concepts, languages, architectures. Springer, Berlin

Widmer G, Kubat M (1996) Learning in the presence of concept drift and hidden contexts. Machine Learning 23:69–101

Yerkes RM, Dodson JD (1908) The relation of strength of stimulus to rapidity of habit-formation. J Comparative Neurol Psychol 18:459–482