Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

TUTORIAL: CONCEPTS OF CONCEPTUAL MODELING

Stewart Robinson[1]

[1] Newcastle University Business School, Newcastle upon Tyne, UNITED KINGDOM

ABSTRACT

Conceptual modeling entails the abstraction of a simulation model from the system of interest to create a simplified representation of the real world. This activity is vital to successful simulation modeling, but it is not well understood. In this tutorial we aim to develop a better understanding of conceptual modeling by exploring this activity from various perspectives. Our initial focus is on defining both a conceptual model and the activity of conceptual modeling. The activity is further explored from a practice-based perspective with reference to an example that is based on actual events. Conceptual modeling is then discussed from three further perspectives: understanding the relationship between a simulation model's accuracy and its complexity; the role of assumptions and simplifications in modeling; and frameworks for guiding the activity of conceptual modeling. This exploration of the concepts of conceptual modeling provides the underlying knowledge required for improving our conceptual models for simulation.

1 INTRODUCTION

Models are used as means of representation across all scientific endeavors and beyond, and computer simulation is among the most utilized modeling approaches. Underlying any model is the modeler's concept of the model, that is, the conceptual model. It is the modeler's conception of how the model will represent the real world. Conceptual modeling is, therefore, at the heart of modeling, but as an activity it remains opaque and not well understood. It is the purpose of this tutorial to provide a better understanding of this activity, in the context of simulation, by providing an elucidation of the underlying concepts of conceptual modeling.

Before delving into the specifics in relation to simulation modeling, it is useful to briefly explain modeling from a scientific perspective. Giere (2004) describes representation (modeling) as similarity, arguing that:

$$S \text{ uses } X \text{ to represent } W \text{ for purposes } P$$

where:

S is an individual scientist, a scientific group, or a larger scientific community

X *(the model)* can be words, equations, diagrams, graphs, photographs, computer-generated images, …

W is an aspect of the real world.

Giere argues that it "is not the model that is doing the representing; it is the scientist using the model who is doing the representing." Following from this, it is our contention that the model artifact exists within the mind of the modeler (scientist). Meanwhile, documented models and computer models are merely representations of that artifact.

Focusing on simulation models as they are used primarily in operations research and management science, we describe the model artifact which exists in the mind of the modeler as the conceptual model; that is, how the modeler conceives the model X to represent an aspect of the world W for the purpose P. Meanwhile, at the core of modeling is abstraction, as a result of which the model represents the world more simply than it actually is. Abstraction is both forced on the modeler because of there being a limited knowledge of, and information about, the world, but it is also a deliberate choice on behalf of the modeler to enable purpose *P* to be addressed with the minimal level of effort.

In this tutorial we aim to develop a better understanding of the activity of conceptual modeling by exploring it from various angles. First, we define both a conceptual model and the activity of conceptual modeling with reference to its artifacts. The discussion then moves to understanding conceptual modeling from a practice-based perspective; for which we reference an example sequence of events. The discussion then moves to exploring the relationship between model accuracy and complexity, the role of assumptions and simplifications in modeling, and finally to frameworks for guiding the activity of conceptual modeling. The focus on concepts is deliberate, as fully understanding these provides the necessary underlying knowledge for improving the activity of conceptual modeling. A previous tutorial sets out in more detail a practical guide to conceptual modeling (Robinson 2017).

2 DEFINING CONCEPTUAL MODELING

As stated above conceptual modeling involves the abstraction of a simulation model from the part of the real world it is representing ('the real system'). The real system may, or may not, currently exist. In this section we define the term conceptual model and then go on to discuss the artifacts of conceptual modeling by setting the activity within the wider context of the modeling process.

2.1 Definition of a Conceptual Model

More formally we define a conceptual model as follows: '… a non-software specific description of the computer simulation model (that will be, is or has been developed), describing the objectives, inputs, outputs, content, assumptions and simplifications of the model.' (Robinson 2008a)

Let us explore this definition in some more detail. First, this definition highlights the separation of the conceptual model from the computer model. The latter is software specific, that is, it represents the conceptual model in a specific computer code. The conceptual model is not specific to the software in which it is developed, but it does form the foundation for developing the computer code.

Second, the conceptual model is the modeler's conception of the computer simulation model. It describes how the modeler conceives the model; it does not describe the real system. In other words, the conceptual model describes how the modeler has abstracted the model away from his/her understanding of the real world. This distinction is important because of the need for model abstraction in simulation.

Third, it is stated that the description is of a computer simulation model 'that will be, is or has been developed.' This serves to highlight the persistent nature of the conceptual model. It is not an artifact that gets created and is then dispensed with once the computer code has been written. It describes the concept of the computer model prior to development, during development and after development. Indeed, the conceptual model persists long beyond the end of the simulation study because we cannot dispose of the model concept. Of course, because the modeling process is iterative in nature (Balci 1994; Willemain 1995; Tako and Robinson 2010), the conceptual model is continually subject to change throughout the life-cycle of a simulation study. It can even change beyond the end of the simulation study as the modeler reflects on the success or otherwise of the work, even though the actual model code will not be changed.

Finally, the definition is completed by a list of what a conceptual model describes. It is vital that the *objectives* of the model are known in forming the conceptual model. The model is designed for a specific purpose and without knowing this purpose it is impossible to create an appropriate simplification. Indeed, poorly understood modeling objectives can lead to an overly complex model.

It is useful to know the model *inputs* and *outputs* prior to thinking about the content of the model. The *inputs* are the experimental factors that are altered in order to try and achieve the modeling objectives. The *outputs* are the reports that inform us as to whether the modeling objectives are being achieved and if not, why they are not being achieved.

Knowing the objectives, inputs and outputs of the model help inform the *content* of the model. In particular, the model must be able to receive the inputs and it must provide the outputs. The model content can be thought of in terms of two dimensions:

- *The scope of the model*: the model boundary or the breadth of the real system that is to be included in the model; "what to model."
- *The level of detail*: the detail to be included for each component in the model's scope; "in how much detail to model it."

The final two items in the list of what a conceptual model describes are the *assumptions* and *simplifications* of the model. These are quite distinct concepts (Robinson and Brooks 2024):

- *Assumptions*. In the presence of inaccurate, incomplete or absent information about the real system, an assumption is the best possible information available that is considered acceptable to enable a model to be completed.
- *Simplifications* are reductions in model complexity that are incorporated into a model to enable easier and more rapid model building, testing, use and maintenance; and/or to improve the transparency of the model.

So, assumptions are a facet of limited knowledge or presumptions, while simplifications are a facet of the desire to create simpler models. There is further discussion on the topic of assumptions and simplifications in section 5.

2.2 Artifacts of Conceptual Modeling

To understand conceptual modeling further it is useful to set it within the wider context of the modeling process for simulation. Figure 1 shows the key artifacts of conceptual modeling. The 'cloud' represents the real world (current or future) within which the problem situation resides; this is the problem that is the basis for the simulation study. The four rectangles represent specific artifacts of the (conceptual) modeling process. These are as follows:

- *System description*: a description of the problem situation and those elements of the real world that relate to the problem.
- *Conceptual model*: as defined in section 2.1.
- *Model design*: the design of the constructs for the computer model (data, components, model execution, etc.) (Fishwick 1995).
- *Computer model*: a software specific representation of the conceptual model.

These artifacts are quite separate. This is not to say that they are always explicitly expressed, with the exception of the computer model. For instance, the system description, conceptual model and model design may not be (fully) documented and can remain within the minds of the modeler. It is, of course, good modeling practice to document each of these artifacts and to use this as a means of communicating their content with other stakeholders in the simulation project.

The model design and computer model are not strictly part of conceptual modeling, but they do embody the conceptual model within the design and code of the model. As highlighted in Figure 1, they make-up the executable model and they are included here for completeness. Of course, our main interest in this paper is in the system description and conceptual model which are artifacts of the activity of conceptual modeling; as represented by the irregular shape with a dashed outline in Figure 1. Unlike the model design and computer model, these two artifacts are independent of the software that will ultimately be used for developing the simulation model.

It is important to recognize the distinction between the system description and the conceptual model. The system description relates to the problem domain, that is, it describes the problem and those elements of the real world that relate to the problem. The conceptual model belongs to the model domain in that it describes those parts of the system description that are included in the simulation model and at what level of detail they are modeled. The author's experience is that these two artifacts are often confused and seen as indistinct. Indeed, a major failure in any simulation project is to try and model the system description (i.e. everything that is known about the real system) and to not attempt any form of model abstraction; this leads to overly complex models.

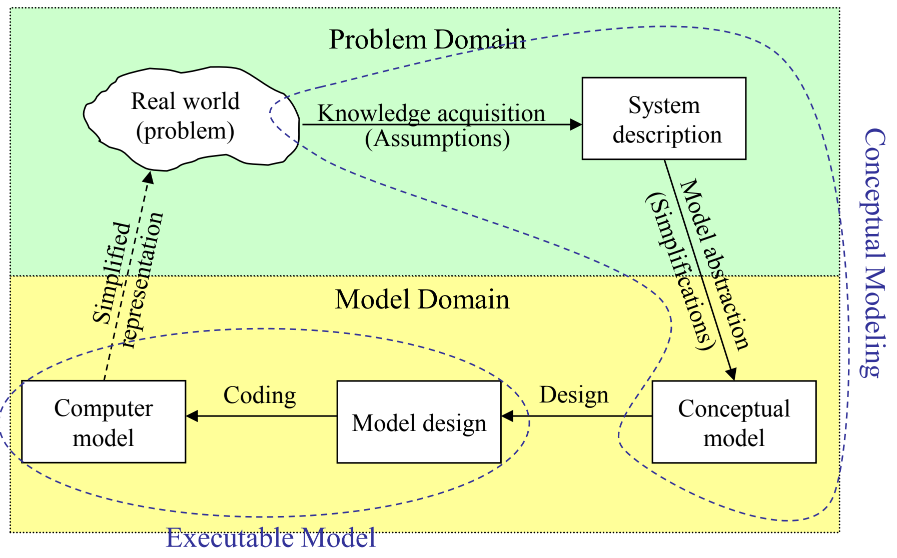


Figure 1: The artifacts of conceptual modeling (Robinson 2010).

The arrows in Figure 1 represent the flow of information, for instance, information about the real world feeds into the system description. The activities that drive the flow of information are described as knowledge acquisition, model abstraction, design and coding. The arrows are not specifically representative of the ordering of the activities within the modeling process, which we know is highly iterative. In other words, a modeler may return to any of the four activities at any point in a simulation study, although there is some sense of ordering in that information from one artifact is required to feed the next artifact. As such, it is incorrect to describe conceptual modeling, or any of the other modeling activities, as steps; they are activities that are performed in a highly iterative manner. As a result, the conceptual model is constantly open to being changed.

The specific and different roles of assumptions and simplifications are highlighted in Figure 1. Assumptions relate to knowledge acquisition, that is, they fill in the gaps in the knowledge that can be acquired about the real world. Meanwhile, simplifications relate to model abstraction, since they are deliberate choices to model the world more simply.

The dashed arrow in Figure 1 shows that there is a correspondence between the computer model and the real world. The degree of correspondence depends on the extent to which the real world is understood, the model contains assumptions that are correct, the simplifications maintain the accuracy of the model, the data in the model are accurate, and the model design and computer code are free of errors. Because the model is developed for a specific purpose, the correspondence with the real world only relates to that specific purpose. In other words, the model is not a general model of the real world, but a simplified representation developed for a specific purpose. The issue of whether the level of correspondence between the model and the real world is sufficient is an issue of validation (Landry, Malouin, and Oral 1983; Balci 1994; Robinson 1999; Sargent 2013). Both conceptual modeling and validation are concerned with developing a simulation of sufficient accuracy for the purpose of the problem being addressed. As a result, there is a strong relationship between the two topics, conceptual modeling being concerned with developing an appropriate model and validation being concerned with whether the developed model is appropriate.

The artifacts described in this section are similar to Zeigler's concepts of the real system, the experimental frame, the base model, the lumped model, and the computer. The interested reader is referred to Zeigler (1976).

3 UNDERSTANDING THE ACTIVITY OF CONCEPTUAL MODELING: A PRACTICE-BASED PERSPECTIVE

In order to further understand the activity of conceptual modeling, we now take a practice-based perspective by situating the activity with reference to an example. While our main interest is in the activity of conceptual modeling, for the purposes of understanding, we also discuss the model design and computer model.

3.1 An Example of a Model (Based on Actual Events)

I am waiting in a long waiting line at a self-service restaurant wondering why the service is so slow. I observe the activities being undertaken by the staff and realize that the problem lies in their inefficient working, for which the layout of the facility does not help, and they are almost certainly under resourced; for instance, there are insufficient pay points.

As my expertise is in modeling queuing systems, I form a model (simplified representation) of the system in my mind, identifying the elements of the system I would represent and their level of detail in order to find ways of reducing the waiting time of the customers.

Following this experience, I contact the owner of the restaurant to see if she would like me to work with them, using my model, to improve their service system. After some discussion, she agrees that we should work together and we organize to meet. At the meeting I present my ideas for the model, now written down as a simple process flow diagram. This enables me to confirm my understanding of the system, update elements of the model that are incorrect, and to add or takeaway detail as is necessary. It also gives the owner confidence in the proposed model and that it will help identify ways of improving the waiting time. The documented model also makes it possible to identify the data required, which the owner will provide.

I then develop a computer model based on the agreed model using a simulation software package. In the process of development, I present the model to the owner on three occasions. Each time, refinements to the model are identified which are reflected back in the process flow diagram. After the third meeting, we agree that the model has been developed to the point where the owner is confident to use it as an aid to decision-making.

3.2 Contextualizing the Modeling Activity

In order to make sense of the story above, Figure 2 provides a summary of the modeling activity, building on the four artifacts identified in Figure 1. The story starts in the problem domain with a *real system* that could be improved through a modeling study. The specific problem is to identify ways of reducing waiting times. The modeler forms an understanding of the system with respect to the problem (*system description, S*) by collecting information, in this case by observing the system in operation. That understanding is not complete and can never be complete; there will always be elements of the system that are not fully known or understood. For an individual modeler there is just one *system description* which consists of all that he/she knows about the system with respect to the problem being addressed. Of course, that system description may not match the description held by others, for example, the owner, restaurant customers or other potential modelers. An option is for the modeler to create a *communicative form* of the *system description* (*CF_S*) in order to share his/her understanding of the system with the owner, which may in turn lead to the modeler refining his/her *system description*. However, there is no evidence of this happening in the example above and it appears that the *system description* remained in the mind of the modeler. This is not to say that the *system description* is static. It is always open to change as new facets of the system come to light during the activity of modeling.

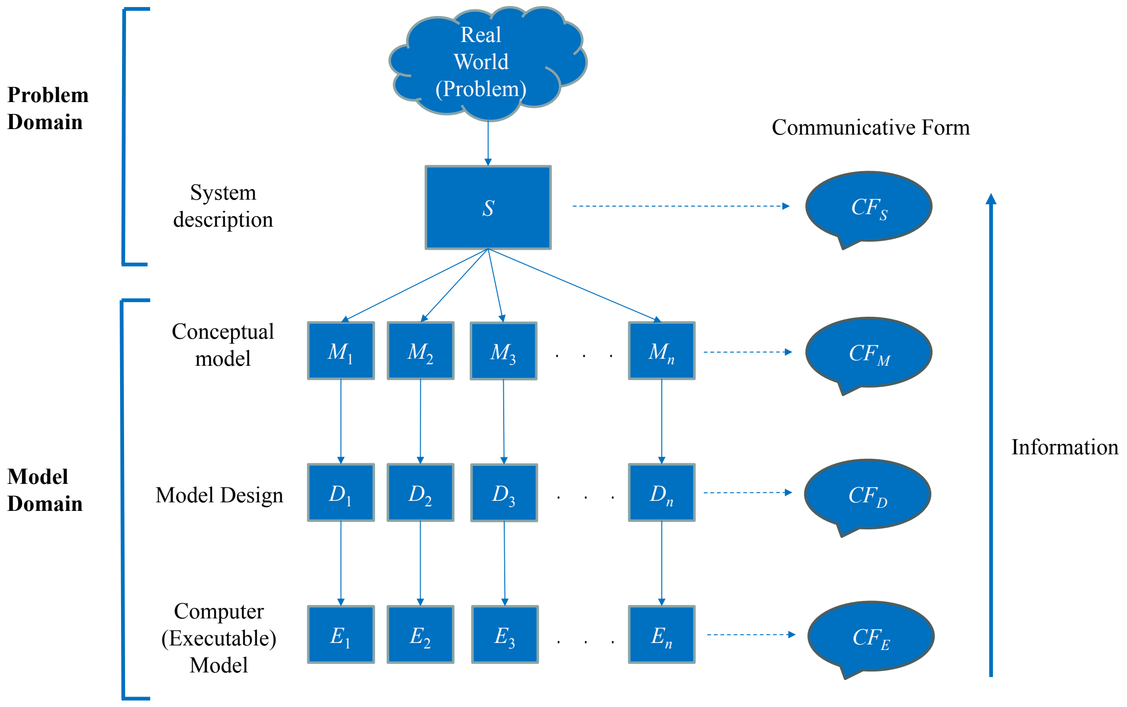


Figure 2: The activity of modeling.

From the *system description* the modeler develops a concept for the model in his/her mind (*conceptual model, M*). This is an internal and implicit model describing how the system needs to be represented to address the problem at hand. There are a great many (n) potential models, $M_i = M_1, M_2, \ldots, M_n$, where higher values of i imply greater complexity. Indeed, Page et al. (2021) theorize that there is an infinite range of models ($n \rightarrow \infty$). Whether the set is infinite or not, the modeler is in effect selecting from one of a great many potential models.

In order to share this *conceptual model*, in this story the modeler creates an external and explicit version by creating a *communicative form* (CF_M) using a simple process flow diagram. Through this process and discussion with the owner the *conceptual model* is refined, that is, $M_i \rightarrow M_j$. In general, as the process progresses the model tends to greater complexity, i.e. $j>i$, but this is not necessarily the case as the model could also be simplified as it is refined.

In order to move to a *computer model*, the modeler considers how the *conceptual model* (M_j) might be represented using a specific commercial off-the-shelf (COTS) simulation package. In doing so, a *model design* (D) is formed. This activity is not explicitly mentioned in the example above, but it must have taken place for the modeler to determine how the *conceptual model* could be coded into the chosen software. Also, there is no evidence of the modeler creating a *communicative form* of the *model design* (CF_D). In creating and reflecting on the *model design*, the conceptual model is open to refinement and so $M_j \rightarrow M_k$. In doing so, there is a tendency for the model to become more complex through this process, i.e. $k>j$, but as above, this is not necessarily the case.

The *computer model* (E - executable) is developed based upon the *model design*, and the model is demonstrated to the owner at various stages of completion on three occasions. As a result of these model demonstrations further refinements are made to the *conceptual model* ($M_k \rightarrow M_l$). Again, the tendency is for the model to become more complex, i.e. $l>k$, but not necessarily so. These refinements are reflected in the *communicative form* of the *conceptual model* (CF_M). This reverse flow of information from the *computer model* towards the *conceptual model* (and potentially to the *system description*) contributes to the iterative process described around the artifacts of conceptual modeling discussed in section 2.2; here this is shown by the arrow on the right of Figure 2. Meanwhile, model verification and validation activities also serve to increase this flow of information back to the *conceptual model*. Ultimately, the aim is for the modeler and owner to have an agreed model (M_l) which is used for aiding decision-making.

We now look in more detail at the *conceptual model*, *model design* and *computer model*, and the *communicative forms*.

3.3 The Conceptual Model

There is a very large set of potential models that could be appropriate for representing the real system in order to address the problem at hand. Henriksen (1988), in some measure, demonstrates this multiplicity of models through the example of representing a simple battle. As further examples, the very simplest model (i=1) for a queuing system is a straightforward single-server queue model. For an agent-based model, the simplest model is a single agent that makes random moves on a grid, and for a system dynamics model, a single stock and flow with no information feedback represents the simplest possible model. The most complex model (i=n, where n is very large) would be a full representation of every known facet of the real system, that is, a full representation of the *system description*. Of course, by spending more time understanding the real system, the *system description* will become more complete, and with it more complex, and as a result there is greater potential to develop a more complex model; the value of n would increase. What we also know is that, even given the same *system description*, different modelers will almost certainly come-up with different *conceptual models* ($1 \leq i \leq n$) for the same problem. This is a result of a variety of factors including different levels of experience and modelers' different cognitive processes for forming the *conceptual model*.

As the modeling activity progresses the *conceptual model* is continuously refined as new information becomes available. An initial model (M_i) is formed very quickly (in the story, while the modeler stands in the waiting line), but through successive meetings and model presentations, the model changes in the light of the new information ($M_i \rightarrow M_j \rightarrow M_k \rightarrow M_l \rightarrow \ldots$).

3.4 The Model Design and Computer Model

Moving beyond the conceptual model, to the *model design* and *computer model*, a range of approaches could be adopted from spreadsheets, through commercial off-the-shelf packages to programming languages. Other model forms could be created, for instance, a physical model or a pen and paper exercise, hence the use of the broader term *executable model* in Figure 2. The modeler could choose to create multiple *model designs* and associated *computer models*, each of the same *conceptual model*, by using different approaches to the model design or different software; however, it is questionable whether there is any practical benefit in doing so. If this were to happen, all the models are executable versions of the same *conceptual model*. Certainly, there is no unique *model design* associated with a *conceptual model*, and there could be multiple *computer models* associated with a *model design*. As such, there is not a single mapping of $M_i \rightarrow D_i$ or of $D_i \rightarrow E_i$.

As stated above, in the activity of creating the *model design* and *computer model*, changes may be made to the *conceptual model*. It is good practice to document these changes by updating any associated *communicative forms*. Meanwhile, the *model design* may not faithfully capture the *conceptual model* or the *computer model* may not exactly match the *model design*. This will lead to verification errors since the *computer model* is not a perfect representation of the intended *conceptual model*; in other words, $E_l \neq D_l \neq M_l$.

3.5 Communicative Forms

A *communicative form* can be created for each of the modeling artifacts: *system description* (*CF_S*), *conceptual model* (*CF_M*), *model design* (*CF_D*) and *computer (executable) model* (*CF_E*). These could build from one another or could be represented in very different forms. Further, more than one *communicative form* could be created for any of the modeling artifacts, particularly if they are needed for different audiences such as an expert modeler and a client.

It is good practice to create *communicative forms* of the modeling artifacts, as these support model verification and validation, and help to build confidence in the model. However, there is no need to create all or any of *communicative forms* of the modeling artifacts, and certainly, the existence of the model is not contingent on there being *communicative forms*.

Here we focus on *communicative forms* for the *conceptual model* (*CF_M*). There are many methods for representing *conceptual models* ranging from quite simple representations to much more elaborate means of documentation. Onggo (2011) provides an overview of some key methods. Examples include concept maps, process flow diagrams, state charts, UML diagrams, SysML, logic and ER models. The choice of method of representation is primarily a matter of modeler and client preference, but also the way in which the representation will be used. The representation may only need to be quite basic to communicate the key facets of the model. Alternatively, the *CF_M* might need to be highly specified so a programmer could create a computer program that faithfully executes the model without the need for further reference to the real system or the modeler.

There could be multiple *CF_Ms* of the model, for example, a simple communicative representation for sharing with the client and a detailed model specification for use by the model developer. However many *CF_Ms* are created, they are all representations of the same *conceptual model*.

It is good practice to create a *CF_M*, but it is not strictly necessary. The *conceptual model* exists in the mind of the modeler whether it is made explicit or not. Of course, without a *CF_M*, there is no means for sharing, validating or explicitly improving the *conceptual model*. It is also good practice to continually revise the *CF_M* as the *conceptual model* is updated ($M_i \rightarrow M_j \rightarrow M_k \rightarrow M_l \rightarrow \ldots$).

It is unlikely that the *CF_M* faithfully captures the *conceptual model* in every aspect. So, for instance, the model that is communicated to the client is not a perfect representation of the intended model. As a result of this miscommunication, overlaid by the client's interpretation of the *conceptual model*, the client's perception of the model differs from the modeler's conception, i.e. $M_i^c \neq M_i^m$, where M_i^c and M_i^m are the client's and modeler's perception of the *conceptual model* respectively.

4. EXPLORING THE RELATIONSHIP BETWEEN ACCURACY AND COMPLEXITY

As stated above, conceptual modeling entails the abstraction of a model from the real world in which the problem being addressed resides. The need for abstraction naturally leads to the question of how far to abstract. A low level of abstraction would imply that the model aims to closely represent the real system with a limited number of simplifications; the model is a "near" abstraction. A high level of abstraction would imply that the model has many simplifications and does not closely represent the real system; it would be a "far" abstraction. The decision on how far to abstract is primarily guided by the objectives of the model and with them the level of accuracy required to provide the necessary confidence to use the model for decision-making. In other words, the model needs to be sufficiently accurate for the purpose at hand, that is, valid (Carson, 1986).

The natural tendency would be to include as much detail as possible as this would seem to provide the "best" model. However, this tendency should be tensioned against the benefits of creating a simpler (more abstract) model, as summarized in Table 1. As such, we advise that the modeler should create the simplest model possible that is capable of addressing the modeling objectives.

Robinson (2008a) proposes that the relationship between model accuracy and its complexity is as shown in Figure 3. Leinweber (1979) also suggests that a similar relationship exists. What this shows is that with increased levels of complexity (that is, reduced levels of abstraction) the marginal benefits to accuracy diminish. Ultimately, adding complexity may lead to reduced accuracy as the knowledge and data about the real system are not available to model at that level of detail.

Table 1: Summary of benefits of simpler simulation models identified in the literature (Robinson and Brooks 2024).

Model Building and Testing	
• Less time consuming to develop and maintain	Fishwick (1988), Ward (1989), Brooks and Tobias (2000), Chwif et al. (2000), Urenda Moris et al. (2008)
• Require less input data	Innis and Rexstad (1983), Ward (1989) Salt (1993)
• Easier to test, verify and validate	Brooks and Tobias (2000), Chwif et al. (2000)
Model Use	
• Less time to run	Fishwick (1988), Sevinc (1991), Salt (1993), Brooks and Tobias (2000)
• Easier to perform sensitivity analysis	Ward (1989), Brooks and Tobias (2000)
• Easier to learn to use	Lucas and McGunnigle (2003)
Model Maintenance	
• More flexible: easier to change	Salt (1983), Ward (1989), Chwif et al. (2000), Lucas and McGunnigle (2003)
• Easier to throw away and start again	Salt (1983)
• Easier to combine with another model	Innis and Rexstad (1983)
Model Understanding	
• Easier to make the model transparent	Sevinc (1991), Salt (1993), Lucas and McGunnigle (2003)
• Assumptions are more apparent and accessible	Ward (1989)
• Easier to interpret and understand why a result has happened	Innis and Rexstad (1983), Sevinc (1991)

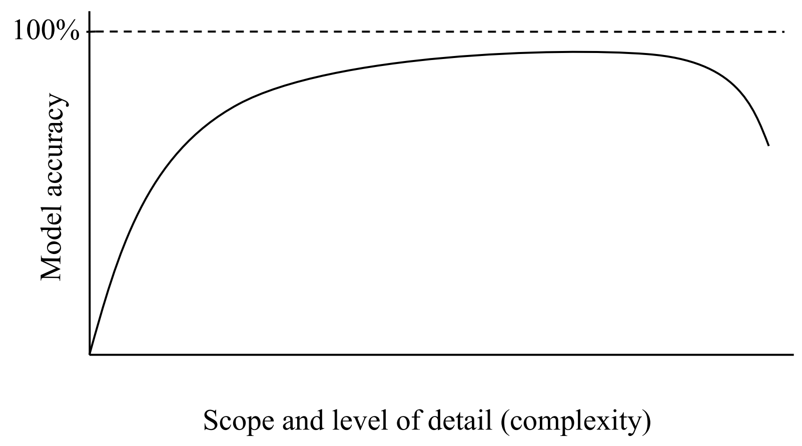


Figure 3: Relationship between simulation model accuracy and complexity (Robinson 2008a).

This relationship can be further understood with reference to the causal loop diagram in Figure 4. In theory, increasing the level of complexity will lead to an increased level of accuracy. Meanwhile, the tendency to desire more accuracy will push for more complexity. Hence the positive feedback loop between complexity and accuracy. On its own, this relationship would suggest that complexity should continue to be increased and the end point would be a perfectly accurate model. However, this positive loop is moderated by the need for knowledge and data (both quantitative and qualitative) required for the model. As complexity increases, more knowledge and data are needed. In the absence of complete and fully accurate knowledge and data, this means more assumptions need to be made, and increased assumptions act to reduce accuracy. Hence, there is a negative feedback loop between complexity, knowledge and data, assumptions and accuracy. The rate at which the marginal benefits of additional complexity diminish, or even reverse the accuracy of a model, depends on the rate at which the benefits of complexity are moderated by the need for further assumptions. We refer to this as the "accuracy/complexity fallacy", in other words, the incorrect belief that adding complexity is always beneficial.

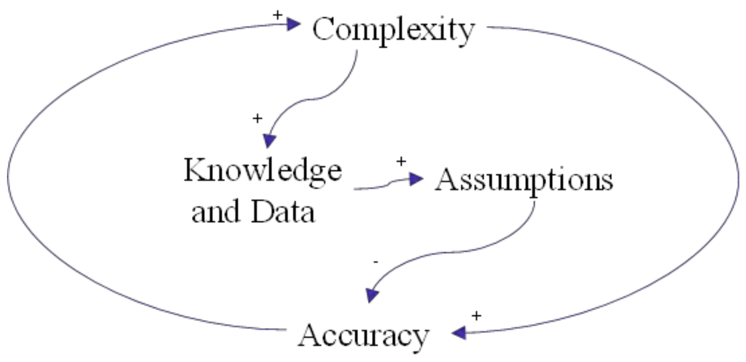
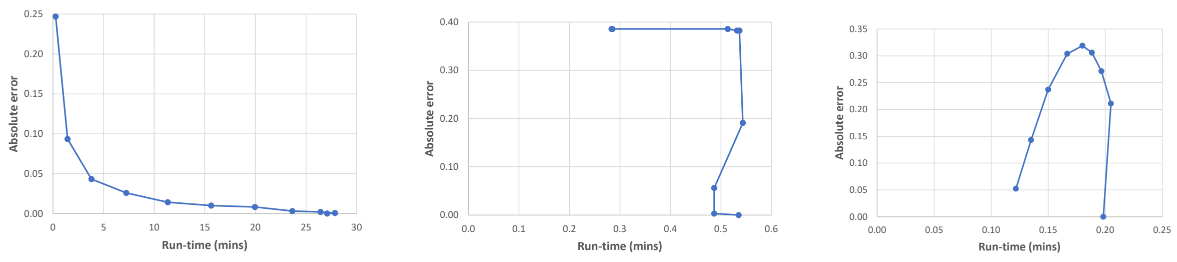


Figure 4: The accuracy/complexity fallacy.

Robinson (2023) performs an empirical investigation of the relationship between model accuracy and complexity. He successively reduces the complexity of three models (two of manufacturing systems and one of a service system) and measures the change in a key model output (throughput and average time-in-system respectively). Complexity is measured both as a simple calculation based on the number of components, the number of connections and the calculational complexity of the model (Brooks and Tobias 1996), and as the run-time of the model. Accuracy is measured as the absolute error between the output of the simplified model and the original model with the highest level of complexity.



Manufacturing Model 1 Manufacturing Model 2 Service Model

Figure 5: Relationship between absolute error and complexity (run-time) (Robinson 2023).

Figure 5 shows three examples of the accuracy vs. complexity relationships Robinson found. Noting that the graphs show absolute error rather than accuracy, the relationship for Manufacturing Model 1 is very close to that proposed by Robinson (2008a) and shown in Figure 3. However, the relationship for the other two models is very different, especially for the service model where beyond a certain threshold the model becomes more accurate as it is simplified.

Robinson (2023) provides detailed explanations of these relationships, which depend, in part, on the type of simplifications and the order in which they are made, or additional complexity is added. For our purposes, it is sufficient to note that the proposition that there are diminishing returns from additional complexity is not strictly correct, although it remains a useful heuristic for guiding modelers to avoid overly complex models.

5. ASSUMPTIONS AND SIMPLIFICATIONS IN SIMULATION MODELING

At the core of conceptual modeling are assumptions and simplifications. Section 2 and Figure 1 highlight the very specific roles of assumptions and simplifications. Assumptions sit alongside knowledge acquisition. They are a facet of having limited knowledge about the real system. Their role is to fill in the gaps in that knowledge. Simplifications sit alongside model abstraction. They are deliberate choices to model the real system more simply with the aim of gaining the benefits of a simpler model (Table 1). Robinson and Brooks (2024) also identify the role of simplifying assumptions. These are assumptions that lead to model simplifications. For instance, because we do not know how customers behave in the self-service restaurant waiting line, we assume a first-in-first-out queue discipline which also has the impact of simplifying the model. As such, a simplification to modeling the waiting lines is a consequence of an assumption about queue discipline.

Robinson and Brooks (2024) discuss in detail approaches for assessing and treating assumptions and simplifications. They recommend that assumptions are assessed for *confidence* and *impact*. *Confidence* is the level of certainty that the assumption about the real system is correct. *Impact* is the anticipated effect on the model output of interest if an assumption is incorrect. Both confidence and impact can be assessed on a simple high/medium/low scale. Of most concern are assumptions for which there is low confidence of them being correct and for which it is expected that there is a high impact on the model output if they are incorrect. Robinson and Brooks recommend using the template shown in Table 2 for documenting and assessing assumptions, with specific actions to be identified for treating critical assumptions, that is, those with confidence/impact at L/M or M/H (denoted by X) or confidence/impact at L/H (denoted by XX). They also suggest identifying the proposed direction of effect on the key outputs of the model.

Table 2: Suggested layout for documenting assumptions (Robinson and Brooks 2024).

				If impact is M or H	**Critical assumptions**	
Description of assumption	**Simplifying assumption? (✓)**	**Confidence (L / M / H)**	**Impact on model output (L / M / H)**	**Direction (+/-)**	**X/XX**	**Action**

Critical assumptions can be treated in a number of ways, including:

- Adopt a worst-case/best-case approach: test the model with extreme worst-case and best-case values for the assumption.
- Treat the assumption as a decision variable: assume there is some level of control over the assumption in the real system and use the model to identify good values for the assumption.
- Improve the information available about the real system.
- Perform a sensitivity analysis (Kleijnen 2005).
- Clearly state the assumptions that are identified as critical: so managers can account for them when interpreting the results and making decisions.

- Change the objectives of the modeling study: to engineer an assumption out of the model.

In a similar way, simplifications should be assessed for their *impact*. Because simplifications are a deliberate choice to model the real system incorrectly, *confidence* in their correctness will always be low and does not need to be assessed. Table 3 provides a template for documenting and assessing simplifications. As for assumptions, actions for treating critical simplifications (X – medium impact and XX – high impact) should be identified, for example:

- Remove all or part of the simplification.
- Interpret the model results in the light of the simplification.
- Test the impact of the simplification on the computer model once it is developed.

Table 3: Suggested layout for documenting simplifications (Robinson and Brooks 2024).

Description of simplification	**Impact on model output (L / M / H)**	**Model transparency (↑/-/↓)**	**If impact is M or H**	**Critical simplifications**	
			Direction (+/-)	**X/XX**	**Action**

6. FRAMEWORKS FOR CONCEPTUAL MODELING

A framework for conceptual modeling provides a set of steps and tools that guide a modeler through the development of a conceptual model. It is also useful for teaching the activity of conceptual modeling, especially to novice modelers. Some examples of frameworks in the literature that the reader may wish to explore further are:

- Conceptual modeling framework for manufacturing (van der Zee 2007)
- Robinson's conceptual modeling framework (Robinson 2008b)
- Karagöz and Demirörs (2011) present a number of conceptual modeling frameworks: Conceptual Model Development Tool (KAMA), Federation Development and Execution Process (FEDEP), Conceptual Models of the Mission Space (CMMS), Defense Conceptual Modeling Framework (DCMF), and Base Object Model (BOM)
- Conceptual modeling with Onto-UML (Guizzardi and Wagner 2012)
- Conceptual modeling using the Structured Analysis and Design Technique (Ahmed, Robinson, and Tako 2014)
- The PartiSim framework (Tako and Kotiadis 2015)
- The ABCmod conceptual modeling framework (Arbez and Birta 2016)
- Dimensional Analysis Conceptual Modeling (DACM) framework (Coatanea et al. 2016)
- Model simplification framework (van der Zee 2019)

For a more detailed discussion on conceptual modeling frameworks see Robinson et al. (2010). Here a very brief outline of Robinson's framework for conceptual modeling is given (Figure 6). For a more detailed account, and an illustration of the framework in use, see Robinson (2008b). In this framework, conceptual modeling involves five activities that are performed roughly in this order (but with iteration):

- Understanding the problem situation
- Determining the modeling and general project objectives

- Identifying the model outputs (responses)
- Identifying the model inputs (experimental factors)
- Determining the model content (scope and level of detail), identifying any assumptions and simplifications

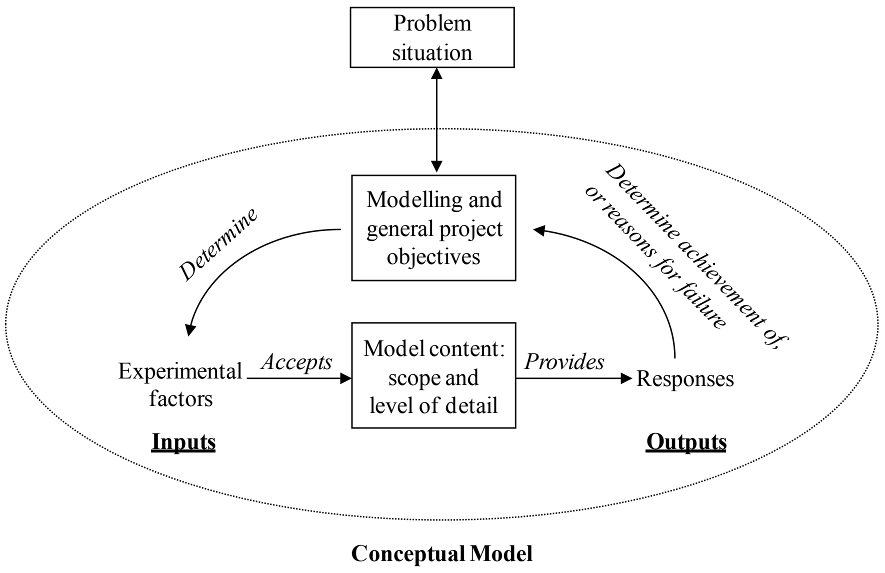

Problem situation
Modelling and general project objectives
Determine
Determine achievement of, or reasons for failure
Experimental factors
Accepts
Model content: scope and level of detail
Provides
Responses
Inputs
Outputs
Conceptual Model

Figure 6: A framework for conceptual modeling (Robinson 2008b).

Starting with an understanding of the problem situation, a set of modeling and general project objectives are determined. These objectives then drive the derivation of the conceptual model, first by defining the outputs (responses) of the model, then the inputs (experimental factors), and finally the model content in terms of its scope and level of detail. Assumptions and simplifications are identified throughout this process.

The ordering of the activities described above is not strict. Indeed, we would expect much iteration between these activities and with the other activities involved in a simulation study: data collection and analysis, model coding, verification and validation, experimentation and implementing the findings.

The framework is supported by a conceptual model template which provides a set of tables that describe each element of the conceptual model. These tables describe:

- Modeling and general project objectives (organizational aim, modeling objectives, general project objectives)
- Model outputs/responses (outputs to determine achievement of objectives, outputs to determine reasons for failure to meet objectives)
- Model inputs/experimental factors
- Model scope
- Model level of detail
- Modeling assumptions
- Model simplifications

These tables provide a *communicative form* for the *conceptual model*. It is also useful to provide a diagram of the model, for instance, a process flow diagram (Robinson 2025).

The modeler works through these tables with the support of the stakeholders and domain experts, iteratively improving them to the point that the modeler and stakeholders are satisfied that the conceptual model meets the requirements for validity, credibility, feasibility and utility (Robinson 2008a). The framework provides a structured approach for making conceptual modeling decisions and for making those decisions explicit through a *communicative form*. In turn, this supports the conceptual modeling activity by providing a conduit for debating ways to improve the conceptual model.

7. CONCLUSION

This tutorial aims to develop a better understanding of the activity of conceptual modeling. In doing so it explores the concepts of conceptual modeling from various angles. Both the notion of a conceptual model and the activity of conceptual modeling are defined. Using a practice-based perspective and a reference example, the activity is further explored. Conceptual modeling is then discussed with respect to the relationship between model accuracy and complexity, the role of assumptions and simplifications, and finally frameworks for conceptual modeling. By better understanding the concepts of conceptual modeling, the aim is to provide the underlying knowledge required to better perform this important activity in the simulation modeling life-cycle.

There remain many challenges to improving conceptual models (Robinson 2020). Continued work is needed to better understand the cognitive processes underlying successful conceptual modeling, that is, how do modelers actually conceive their models? This would help us to develop better frameworks to guide modelers. We also need to improve the quality of communicative forms of conceptual models as well as for the other modeling artifacts. Much could be achieved by strengthening the links between research and practice in conceptual modeling in order to learn from one another.

REFERENCES

Ahmed, F., S. Robinson, and A. A. Tako. 2014. "Using the Structured Analysis and Design Technique (SADT) in Simulation Conceptual Modeling". In *2014 Winter Simulation Conference (WSC)*, 1038-1049 https://informs-sim.org/wsc14papers/includes/files/093.pdf.

Arbez, G. and L. G. Birta. 2016. "A Tutorial on ABCmod: An Activity Based Discrete Event Conceptual Modelling Framework". In *2016 Winter Simulation Conference (WSC)*, 88-102 https://www.informs-sim.org/wsc16papers/011.pdf.

Balci, O. 1994. "Validation, Verification, and Testing Techniques Throughout the Life Cycle of a Simulation Study". *Annals of Operations Research* 53:121-173 https://doi.org/10.1007/BF02136828.

Brooks, R. J. and A. M. Tobias. 1996. "Choosing the Best Model: Level of Detail, Complexity and Model Performance". *Mathematical and Computer Modeling* 24(4):1-14 https://doi.org/10.1016/0895-7177(96)00103-3.

Brooks, R. J. and A. M. Tobias. 2000. "Simplification in the Simulation of Manufacturing Systems". *International Journal of Production Research* 38(5):1009-1027 https://doi.org/10.1080/002075400188997.

Carson, J. S. 1986. "Convincing Users of Model's Validity is Challenging Aspect of Modeler's Job". *Industrial Engineering* 18(6):74-85.

Chwif, L., M. R. P. Barretto and R. J. Paul. 2000. "On Simulation Model Complexity". In *2000 Winter Simulation Conference (WSC)*, 449-455 https://informs-sim.org/wsc00papers/063.PDF

Coatanea, E., R. Roca, H. Mokhtarian, F. Mokammel, and K. Ikkala. 2016. "A Conceptual Modeling and Simulation Framework for System Design". *Computing in Science & Engineering* 18(4):42-52 https://doi.org/10.1109/MCSE.2016.75.

Fishwick, P. A. 1988. "The Role of Process Abstraction in Simulation". *IEEE Transactions on Systems, Man, and Cybernetics* 18(1): 18-39 https://doi.org/10.1109/21.87052.

Fishwick, P. A. 1995. *Simulation Model Design and Execution: Building Digital Worlds*. Upper Saddle River, New Jersey: Prentice-Hall, Inc.

Giere, R. N, 2004. "How Models are Used to Represent Reality". *Philosophy of Science* 71(5):742-752 https://doi.org/10.1086/425063.

Guizzardi, G. and G. Wagner. 2012. "Tutorial: Conceptual Simulation Modeling with Onto-UML". In *2012 Winter Simulation Conference (WSC)*, 52-66 https://informs-sim.org/wsc12papers/includes/files/inv284.pdf.

Henriksen, J. O. 1988. "One System, Several Perspectives, Many Models". In *1988 Winter Simulation Conference*: 352-356 https://informs-sim.org/wsc88papers/1988_0050.pdf.

Innis, G. and E. Rexstad. 1983. "Simulation Model Simplification Techniques". *Simulation* 41(1):7-15 https://doi.org/10.1177/003754978304100101.

Karagöz, N. A. and O. Demirörs. 2011. "Conceptual Modeling Notations and Techniques". In *Conceptual Modeling for Discrete-Event Simulation*, edited by S. Robinson, R. J. Brooks, K. Kotiadis, and D-J. van der Zee, 179-209. Boca Raton, FL: Chapman and Hall/CRC.

Kleijnen, J. P. C. 2005. "An Overview of the Design and Analysis of Simulation Experiments for Sensitivity Analysis". *European Journal of Operational Research* 164:287-300 https://doi.org/10.1016/j.ejor.2004.02.005.

Landry, M., J. L. Malouin, and M. Oral. 1983. “Model Validation in Operations Research”. *European Journal of Operational Research* 14(3):207-220 https://doi.org/10.1016/0377-2217(83)90257-6.

Leinweber, D. 1979. *Models, Complexity, and Error*. A Rand Note Prepared for the US Department of Energy (N-1204-DOE), Rand, Santa Monica, CA.

Lucas, T. W. and J. E. McGunnigle. 2003. “When is Model Complexity too Much? Illustrating the Benefits of Simple Models with Hughes’ Salvo Equations”. *Naval Research Logistics* 50:197-217 https://doi.org/10.1002/nav.10062.

Onggo, S. 2011. “Methods for Conceptual Model Representation”. *Conceptual Modeling for Discrete-Event Simulation*, edited by S. Robinson, R. J. Brooks, K. Kotiadis, and D-J. van der Zee, 337-354. Boca Raton, FL: Chapman and Hall/CRC.

Page, E. H., J. R. Thompson, and M. Koehler. 2021. “Sic Semper Simulation - Balancing Simplicity and Complexity in Modeling and Analysis”. In *2021 Winter Simulation* Conference https://www.informs-sim.org/wsc21papers/224.pdf.

Robinson, S. 1999. “Simulation Verification, Validation and Confidence: A Tutorial”. *Transactions of the Society for Computer Simulation International* 16(2):63-69.

Robinson, S. 2008a. “Conceptual Modelling for Simulation Part I: Definition and Requirements”. *Journal of the Operational Research Society* 59(3):278-290 https://doi.org/10.1057/palgrave.jors.2602368.

Robinson, S. 2008b. “Conceptual Modelling for Simulation Part II: A Framework for Conceptual Modelling”. *Journal of the Operational Research Society* 59(3):291-304 https://doi.org/10.1057/palgrave.jors.2602369.

Robinson, S. 2010. “Conceptual Modeling for Simulation”. In *Encyclopedia of Operations Research and Management Science*, edited by J. J. Cochran. New York: Wiley.

Robinson, S. 2017. “A Tutorial on Simulation Conceptual Modeling”. In *2017 Winter Simulation Conference*, 565-579 https://www.informs-sim.org/wsc17papers/includes/files/041.pdf.

Robinson, S. 2020. “Conceptual Modelling for Simulation: Progress and Grand Challenges”. *Journal of Simulation* 14(1):1-20 https://doi.org/10.1080/17477778.2019.1604466.

Robinson, S. 2023. “Exploring the Relationship between Simulation Model Accuracy and Complexity”. *Journal of the Operational Research Society* 74(9):1992-2011 https://doi.org/10.1080/01605682.2022.2122740.

Robinson, S. 2025. *Simulation: The Practice of Model Development and Use* (3rd ed.). London: Bloomsbury.

Robinson, S. and R. Brooks. 2024. “Assumptions and Simplifications in Discrete-Event Simulation Modelling”. *Journal of Simulation, forthcoming* https://doi.org/10.1080/17477778.2024.2407369.

Salt, J. 1993. Simulation Should be Easy and Fun. In *1993 Winter Simulation Conference,* 1-5 https://informs-sim.org/wsc93papers/1993_0001.pdf.

Sargent, R. G. 2013. “Verification and Validation of Simulation Models”. *Journal of Simulation* 7(1):12-24 https://doi.org/10.1057/jos.2012.20.

Sevinc, S. 1991. “Theories of Discrete Event Model Abstraction”. In *1991 Winter Simulation Conference*, 1115-1119 https://informs-sim.org/wsc91papers/1991_0149.pdf.

Tako, A.A. and S. Robinson. 2010. “Model Development in Discrete-Event Simulation and System Dynamics: An Empirical Study of Expert Modellers”. *European Journal of Operational Research* 207:784-794 https://doi.org/10.1016/j.ejor.2010.05.011.

Tako, A. A. and K. Kotiadis. 2015. “PartiSim: A Multi-Methodology Framework to Support Facilitated Simulation Modelling in Healthcare”. *European Journal of Operational Research* 244(2): 555-564 https://doi.org/10.1016/j.ejor.2015.01.046.

Urenda Moris, M., A. H. C Ng, and J. Svensson. 2008. “Simplification and Aggregation Strategies Applied for Factory Analysis in Conceptual Phase using Simulation”. In *2008 Winter Simulation Conference*, 1913-1921 https://www.informs-sim.org/wsc08papers/236.pdf.

van der Zee, D. J. 2007. “Developing Participative Simulation Models: Framing Decomposition Principles for Joint Understanding”. *Journal of Simulation* 1(3):187-202 https://doi.org/10.1057/palgrave.jos.4250020.

van der Zee, D. J. 2019. “Model Simplification in Manufacturing Simulation – Review and Framework”. *Computers & Industrial Engineering* 127:1056-1067 https://doi.org/10.1016/j.cie.2018.11.038.

Ward, S. C. 1989. “Arguments for Constructively Simple Models”. *Journal of the Operational Research Society* 40(2):141-153 https://doi.org/10.1057/jors.1989.19.

Willemain, T. R. 1995. “Model Formulation: What Experts Think About and When”. *Operations Research* 43(6):916-932 https://www.jstor.org/stable/171635.

Zeigler, B. P. 1976. *Theory of Modeling and Simulation*. New York: Wiley.

AUTHOR BIOGRAPHY

STEWART ROBINSON is Dean and Professor of Operational Research at Newcastle University Business School, UK, and Chair of the Chartered Association of Business Schools. His research focuses on the practice of simulation model development and use. Key areas of interest are conceptual modeling, model validation, output analysis and alternative simulation methods (discrete-event, system dynamics and agent based). Professor Robinson is author/co-author of seven books on simulation, co-founder of the Journal of Simulation and co-founder of the UK Simulation Workshop conference series. He was President of the Operational Research Society (2014-2015). His email address is stewart.robinson@newcastle.ac.uk.