CHAPTER 13

METHODS FOR CONCEPTUAL MODEL REPRESENTATION

Stephan Onggo

Department of Management Science, Lancaster University Management School,
Lancaster University, Lancaster, LA1 4YX, UNITED KINGDOM

13.1 INTRODUCTION

Simulation conceptual model (or conceptual model for brevity) representation is important in a simulation project because it is used as a tool for communication about conceptual models between stakeholders (simulation analysts, clients and domain experts). There is a point in the simulation project when the conceptual modelling process happens inside the individual stakeholder’s mind. This ‘thinking’ process includes reflection on how to structure the problem and how the simulation model should be designed to help decision makers solve the problem at hand, subject to certain constraints. At some point in the simulation project, the conceptual model needs to be communicated to other stakeholders. Hence, the role of conceptual model representation is crucial. Moreover, different stakeholders may have different views on the system; their reasons may include different levels of understanding of the system, prior experience and personal objectives. Nance (1994) refers to conceptual model representation for this purpose as the communicative model. When communication involves different types of stakeholders, a standard representation that can be understood by all stakeholders is essential.
The fact that communication between stakeholders is important for the success of a simulation project (Robinson and Pidd 1998) makes the need for good conceptual model representation become even more essential.

The main challenge in designing conceptual model representation is to devise a representation that can be understood by all stakeholders and yet which is expressive enough to handle the varying levels of complexity in the conceptual model. To complicate matters further, there is no single accepted definition of what a conceptual model is (see Robinson/chapter 1) as what is to be represented will surely affect its representation. Given the different definitions for a conceptual model, it is not surprising to see that a wide variety of conceptual model representations have been proposed.

One of the surveys conducted by Wang and Brooks (2007) listed the popularity of a number of methods for conceptual model representation. They are, in order of popularity, textual representations (e.g. list of assumptions and simplifications, component list and text description), process flow diagram, logic diagram (or flow chart), activity cycle diagram and UML. We can group these representation methods into three categories: textual representation, pictorial representation and multi-faceted representation. The objective of this chapter is to discuss the three methods for conceptual model representation and issues related to their use in practice. In the examples, we will demonstrate how the methods are applied to represent components of a conceptual model based on Robinson’s definition. The same principle can be applied to other conceptual model definitions.

---

1 Simulation analysts often deal with clients and domain experts who have little knowledge about simulation.
Robinson (2008) categorizes the components of a conceptual model into problem-domain components and model-domain components (see also Robinson/Chapter 1). The problem-domain components are used as a means of communication mainly between clients/domain experts and simulation analysts, between clients, or between domain experts. These components include objectives, inputs, outputs, contents (scope/structure, level of detail, assumptions and simplifications) and data requirement. These components define parts of the system that are important for the objectives at hand. These components are independent of any modelling technique that is going to be used. At this stage, we need to decide whether simulation is the right tool to model the system.

Assuming that we have decided that simulation is the best option, we need to specify the model domain components. At this stage, we need to decide the most suitable paradigms such as: discrete-event simulation, system dynamics and agent-based simulation. The choice between the different paradigms depends on the objective of the simulation project. Discrete-event simulation is suitable when it is necessary to track entities from their arrival in the system until they leave it (or until the simulation is completed). The results from individual entities are aggregated in the simulation outputs. System dynamics is suitable when the population of entities and the rates of entities moving from one place to another are more important than the individual entities. System dynamics also provides a way to explore complex feedback systems and it enables us to analyze the mutual interactions among entities over time. Agent-based simulation is particularly useful when the entities are adaptive, have the ability to learn, or can change their behaviours. Agent-based simulation is also useful when the behaviours of entities are affected by their spatial
locations and the structure of their communication networks.

Each simulation-modelling paradigm views the system of interest differently. Discrete-event simulation sees a system as a collection of events, entities, resources, queues, activities and processes. System dynamics views a system as a collection of stocks, flows and delays. From an agent-based simulation perspective, a system is formed by a collection of agents and their environment. The communication at this stage, i.e. the development of a simulation model based on one of the paradigms, happens mainly between simulation analysts. The output of this stage is a simulation model that is independent of any software implementation.\(^1\) The components of the simulation model are referred to as the model-domain components because they depend on the modelling paradigm used in the development process. Consistent with the theme of the book, this chapter focuses on the conceptual model representation in the discrete-event simulation. The examples given in this chapter are based on the District General Hospital Performance Simulation (DGHPSim) project to demonstrate how the methods discussed in this chapter could be applied in a real simulation project. DGHPSim is a collaborative research project that involves three British universities. The project aims to develop generic simulation models of entire acute hospitals so as to understand how hospital performance can be improved (Gunal and Pidd, in press).

The remainder of this chapter is organized as follows. This chapter divides conceptual model

---

\(^1\) This may not be true in a simulation project where the requirement dictates the use of a specific implementation-dependent model representation (for reasons such as the familiarity to the simulation software). See Robinson/Chapter 1 for the discussion on the importance of the software independency.
representation methods into three categories: textual, pictorial and multi-faceted. Section 2 will discuss the textual representation. Section 3 will focus on the most widely-used pictorial representation in simulation, i.e. diagrams. Section 4 will discuss the multi-faceted representation. Finally, concluding remarks are made in Section 5.

13.2 TEXTUAL REPRESENTATION

As mentioned earlier, at some stage in the simulation project, a conceptual model needs to be communicated to other people. The communication can be done by passing the information verbally or via texts. In this chapter, we are more interested in written communication. A written document describing a conceptual model can become an important part of the simulation project. For example, the document can be used in any form of electronic communication and can even be used as part of the contract for the simulation project. The main objectives of the textual representation are to describe the content of each conceptual model component and to elicit visual imagery for the structure of the conceptual model components using narrative texts. The following excerpts shows how the conceptual model of a hospital simulation project is represented using narrative texts.

“The objective of this project is to improve overall hospital performance. The performance is measured based on the waiting times of patients at various departments at the hospital. The key departments included in the model are: Accident & Emergency (A&E), outpatients and in-patients. Patients arrive in the system through A&E and Outpatients. Depending on the condition, a patient can be admitted to hospital (in-patient) or discharged…”
The excerpt describes a number of components in a conceptual model, such as: the objective, the output of the model, the scope of the model and the flow of patients in the model. The main advantage of textual representation is its flexibility. Simulation analysts can write the description of a model in various ways and in different styles, for example, the previous excerpt could have been written in a bullet-point format or in tabular form. Textual representation can be done quickly, especially for some conceptual model components such as assumptions (and more naturally, perhaps). Most software that supports simulation modelling provides a facility for text annotations so that analysts can easily provide descriptions of the model and any part of it. This might explain why textual representation is very popular for documenting the assumptions in the survey carried out by Wang and Brook (2007). Robinson (2004, Chapter 6, Appendix 1, Appendix 2) shows examples of how to specify conceptual model components using textual representation.

Textual representation is not without its disadvantages. First, the flexibility of textual representation may lead to an ambiguous description of the simulation model. As in any types of representation, the challenge here is to ensure that the mental model encoded in the text is decoded correctly by the target recipients. Effective textual representation should pay attention to the structure and content of the text and the assumptions about the target recipients (in this case, the stakeholders in a simulation project). Good organization of the text (sections, subsections, bullet-point lists, succinct description, etc.) may reduce ambiguity in the description. It may be necessary to develop a common understanding of a set of keywords (such as: objective, model, assumptions, etc.) among the stakeholders before the conceptual model is discussed. Another disadvantage of textual representation is that the correctness of the conceptual model cannot be
verified elegantly using mathematical techniques. However, the conceptual model can still be validated using a more subjective validation technique such as the use of domain experts’ opinions (see Bilgen and Tanriover / Chapter 16 for various validation methods in conceptual modelling). Finally, and rather obviously, communication can work only if all stakeholders understand the language used in the texts.

13.3 PICTORIAL REPRESENTATION

The next type of conceptual model representation is pictorial representation where the conceptual model is communicated through pictures. Research in cognitive science has shown that a pictorial representation is very effective (for example, Larkin and Simon 1987). Unlike textual representation that presents information sequentially, pictorial representation can show the information in two dimensions which allows non-sequential flows to be represented more effectively. In simulation, diagrams are the most widely-used pictorial representation for conceptual models. A diagram is a special type of pictorial representation that represents information using shapes/symbols that are connected by links (such as arrows and lines). The use of diagrams in simulation modelling has increased, especially after graphical workstations became more affordable. Pooley (1991) conducted one of the earliest surveys on the use of diagrams in simulation modelling. Recently, Wang and Brooks (2007) conducted another survey that showed a number of popular diagrams used in simulation modelling. This section discusses two of the most popular diagrams in the survey, i.e. the activity cycle diagram and the process flow diagram. We will also discuss another widely-known diagram called the event relationship graph. The three diagrams are chosen because they focus on different aspects of a system that is to be modelled.
13.3.1 Activity cycle diagram (ACD)

ACD (Hills 1971) is an implementation-independent diagram that is used to model a system by focusing on the changes in the states of key entities in the system. When an entity arrives at the system, it must go through a set of activities that may change the state of the entity (for example, in service or waiting) until the entity leaves the system. In ACD, the change in the state of each entity is represented by a series of alternate dead and active states. A dead state is represented as an oval and corresponds to a state where an entity must wait until the required resources are available. An active state is represented as a rectangle and corresponds to a state where an entity is in an activity with a specific duration (it may be sampled using a predefined distribution function).

Figure 13.1 shows how ACD can be used to represent an A&E simulation model. The diagram shows the cycle of entity patients. This simulation model assumes that the arrival of patients follows a certain distribution function (hence, an active state). Once a patient arrives at A&E, the patient waits until the clerk is ready for the registration process (a dead state). When the clerk is ready, the registration takes a certain amount of time which may be sampled from a distribution function. The process continues until the patient leaves A&E.
In some cases, we may be interested in the state of a specific resource in the system over time. For this purpose, we can add the cycle of the resource to the diagram. For example, if we are interested in the utilization of each clerk, we can add a cycle for the clerk. The state of the clerk will constantly switch between being in a dead state of waiting for a patient to arrive, and being in an active state, registering a patient. The patient cycle meets the clerk cycle at the registration process. A complete ACD for the system under study should show the cycles of key entities and key resources. One of the modellers’ main tasks is to decide which key entities and resources should be included in the model. ACD is commonly used to represent the model-domain component in a discrete-event simulation-modelling paradigm. One of the main reasons for this is that ACD could easily show key components such as: queues (all dead states), system state, resources, activities (most active states) and processes (all cycles).
13.3.2 Process flow diagram

A process flow diagram is commonly used to model the flow of processes in a system. A process in simulation is often defined as a sequence of activities (and events) in chronological order. A process flow diagram focuses more on the sequence (or structure) of activities and the flow of entities from the point where they enter the system until they leave the system. This is different from ACD which focuses more on the states that the entities and resources are in. Most commercial visual interactive modelling software (VIMS) that supports discrete-event simulation uses some sort of process flow diagram. The VIMS have their proprietary symbols to represent activities and their sequence in a process. In fact, some VIMS call the activities by other names such as tasks and machines.

In this section, we choose one of the widely-known process flow diagrams called Business Process Diagram (BPD). BPD is the diagram that is specified by BPMN (Business Process Modeling Notation). BPMN is a standard that has been developed to provide a notation that is understandable by all business stakeholders (business people, business analysts and technical developers) to model business processes. BPD is chosen because it is a widely-known standard and is independent of any proprietary notations that may trap us into implementation-dependent components. Hence, it is suitable for our objective, to provide a tool for communication about conceptual models between stakeholders which are independent of any software implementation. The four main BPD elements are activities (shown as rounded rectangles), events (circles), connectors (lines) and gateways (diamonds). As we know, a process is a sequence of activities and events. Hence, these four BPD elements can be used to model many different processes.
BPD activities are used to represent real-world activities. The activities can be further decomposed into sub-activities. This facility is important to allow a hierarchical modelling process. In other words, the activity at one level in the hierarchy can be viewed as a process from a lower level in the hierarchy. The lowest-level activity, i.e. the activity that will not be decomposed further, is called a task. BPD events are used to represent events that happen in the real world. An event can start a process (i.e. start the first activity in the process), start an intermediary activity, or end a process (i.e. end the last activity in the process). BPD connectors are used to represent flows. BPD gateways are used to represent decisions in the process flow, i.e. joins, forks and mergers. BPMN (http://www.bpmn.org/) provides a more detailed explanation of each element and other elements that are not mentioned here, such as pool and lane.

Figure 13.2 shows the BPD of a typical hospital operation which includes three activities: Accident and Emergency (A&E), Outpatients and In-patients. Patients arrive in the system through A&E and Outpatients. The arrivals of patients are events that start the processes. Depending on the condition, a patient can be admitted to hospital (in-patient) or discharged. Discharge is an event that terminates a process. If we want to add to the level of detail in the model, we can move to a lower layer in the system hierarchy and treat any of the activities as a process which can be decomposed further into a number of activities.
13.3.3 Event relationship graphs (ERG)

ERG (Schruben 1983) provides a concise representation of causality in a system. ERG is effective in representing model-domain components in discrete-event simulation modelling paradigms. In an ERG, an oval represents state changes when an event occurs and an arrow shows that an event at the start of the arrow generates an event at the end of the arrow (hence, it shows the causality between the two events). The arrows also specify the conditions (/) and the times for events to be scheduled (arrows with time delays are drawn in bold).

Figure 13.3 shows the ERG of a typical outpatient department. A ‘GP referral’ event triggers the whole process. This event serves as a bootstrap event that will generate subsequent arrivals to the outpatient department (with a specified time delay $t_o$). The ‘GP referral’ event generates a ‘start first appointment’ event when at least one consultant is available. The ‘GP referral’ event also changes the system state, i.e. increases the number of patients waiting for their first appointments. The ‘start first appointment’ event leads to a ‘finish first appointment’ event (with
a specified time delay $t_1$). Subsequently, if the patient needs a follow-up appointment, a ‘finish first appointment’ event may generate a ‘start follow-up appointment’ event. Otherwise, treatment for the patient is complete. The ‘finish first appointment’ event changes the system state, i.e. reduces the number of patients waiting for their first appointments and, in some cases, increases the number of patients waiting for their follow-up appointments.

**Figure 13.3 Outpatients – Event Relationship Graphs**

13.4 MULTI-FACETED REPRESENTATION

Despite the differences in the definitions of a conceptual model, researchers agree that a conceptual model comprises a number of components. Hence, it is unlikely that a single diagram can be used to represent completely a conceptual model. For this reason, a multi-faceted representation is more suitable for a more complete documentation of a conceptual model. In a multi-faceted conceptual model representation, a set of diagrams and textual representation are used to represent different conceptual model components. Multi-faceted representation has been
used widely in software engineering. One of the most widely-used multi-faceted representations in software engineering is the Unified Modeling Language (UML). UML 2.0 defines thirteen types of diagrams to represent three aspects of software system: static application structure, behaviour and interactions. A more detailed description can be found at http://www.uml.org. A multi-faceted representation such as UML has the potential to provide a more comprehensive representation of a conceptual model.

13.4.1 UML and SysML

Richter and Marz (2000) proposed the use of four UML diagrams for documenting a simulation project. They use the ‘use case’ diagram to document the interaction between users and the simulation model. The structure of a simulation model is represented using a class diagram. The dynamics of the model are represented using an interaction diagram and a state diagram. Vasilakis et al. (2009) used UML to specify the requirements for a patient flow simulation model. They use the activity diagram to specify the flow of patients, ‘use case’ diagram to give the detail function of each activity in the activity diagram and state diagram to capture the state-transition of patients. These works show that we can use UML diagrams to provide a multi-faceted representation of a conceptual model. We can extend their work to include the use of sequence diagrams and collaboration diagrams to show the dynamics of a model. Figure 13.4 shows a sequence diagram of the same A&E system that was shown earlier in Figure 13.1.
The Object Management Group (OMG) publishes the systems modelling language (SysML) standard which is an extension of UML and is designed to support system modelling. Huang et al. (2007) explored the use of SysML in representing conceptual models. One of the ultimate objectives is to provide a conceptual model representation (independent of any implementation software) that could be translated automatically to any simulation software (implementation dependent). SysML uses four UML diagrams (sequence diagram, state transition, use case diagram and package diagram), three modified UML diagrams (activity diagram, block definition diagram and internal block diagram) and two new diagrams (requirement diagram and parameter diagram). These diagrams are used to specify a system’s structure, behaviour and requirements. Heavey / Chapter 14 discusses SysML in greater detail.
13.4.2 Unified conceptual model

Onggo (2009) proposed the use of another set of diagrams to represent the different conceptual model components. Table 13.1 shows Onggo’s multi-faceted conceptual model representation. In this chapter, we add a number of representation methods that were not part of Onggo’s original methods. The first column gives the domains of a conceptual model’s components. The second column lists the components of a conceptual model. The last column shows the diagrams selected to represent the conceptual model’s components.

Table 13.1  Diagrams used in the unified conceptual model (Adapted from Onggo, B.S.S., Journal of Simulation, 3 (1), 42, 2009. With permission.)

<table>
<thead>
<tr>
<th>Domain</th>
<th>Component</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem</td>
<td>Objectives</td>
<td>Objective Diagram, Purposeful Activity Model</td>
</tr>
<tr>
<td></td>
<td>Inputs</td>
<td>Influence Diagram</td>
</tr>
<tr>
<td></td>
<td>Outputs</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Contents</td>
<td>Business Process Diagram with textual representation</td>
</tr>
<tr>
<td></td>
<td>Data requirement</td>
<td>Textual representation, Data dictionary</td>
</tr>
<tr>
<td>Model</td>
<td>Discrete-Event</td>
<td>Activity Cycle Diagram, Event Relationship Graph</td>
</tr>
<tr>
<td></td>
<td>System Dynamics</td>
<td>Stock and Flow Diagram, Causal Loop Diagram</td>
</tr>
<tr>
<td></td>
<td>Agent-based</td>
<td>Flow Chart, Business Process Diagram, UML Activity Diagram</td>
</tr>
</tbody>
</table>

Objectives component

The objective is the most important component in simulation modelling. Objectives are used to
judge the success of a problem-solving exercise and to compare the quality of various decision alternatives. Onggo uses an objective diagram to represent the objectives component of a simulation conceptual model. Objective diagrams (Keeney 1992) are commonly used to structure objectives in decision science. They classify objectives into two categories: fundamental objectives and means objectives.

The fundamental objectives are the end result that we want to achieve and are organized into hierarchies. In an objective diagram, each fundamental objective is represented as a node in a tree. The higher-level fundamental objectives represent more general objectives and their measurement can be obtained from lower-level fundamental objectives. Thus, the lowest-level fundamental objectives provide the basis on which various design alternatives are measured. Consequently, the highest-level fundamental objective provides the ultimate measurable consequence that will be used to evaluate and compare various design alternatives. Figure 13.5 shows an example of fundamental objectives from a project that seeks to improve the performance of a hospital. The performance is linked to the waiting times of patients at the hospital, which are the averages of waiting times of patients in its various departments: Accident & Emergency (A&E), outpatients and in-patients. These measurements will be used to compare alternatives. Second-level fundamental objectives can be further expanded if necessary. For example, A&E performance is obtained from two components: patient total time (98% of patients must spend less than four hours in A&E) and staff utilization.
Means objectives are important because they help us to achieve fundamental objectives and they are often used when the fundamental objectives are difficult to measure directly. In some cases, identifying means objectives can help us to characterize new alternatives. In the objective diagram, means objectives are organized into networks. Two examples of means objectives are shown in Figure 13.5. Maximizing the number of day cases and reducing patients’ lengths of stay are important because they increase the number of available beds. Hence, this may reduce the waiting times for both emergency and elective admissions. In general, fundamental objectives can be differentiated from means objectives by continuously asking the question of why an objective is important. An objective is a means objective if it is important because it helps achieve another objective. The same question is repeated until we find an answer where an objective is important because it is important.

The objective diagram, however, only considers the structure of objectives. It may be useful to
show the conditions under which the structure is built. Kotiadis (2007) presented interesting work on using Soft Systems Methodology to determine simulation objectives (see Kotiadis / Chapter 10 for more detail). In particular, she presented steps to extract simulation objectives from the purposeful activity model (PAM). This work implies that PAM can be used to complement the objective diagram to show the conditions under which the objectives diagram is drawn.

**Inputs and outputs component**

Once the objectives have been defined, we need to translate them into output variables that can be quantified, and to identify the different alternatives (input variables) that will achieve the objectives. Outputs can be directly inferred from objectives. The controllable input variables are sometimes referred to as the decision variables. The inputs are sometimes specified explicitly in the objectives; otherwise they can be obtained from the clients. Onggo (2009) uses an influence diagram to represent the relationship between input variables and output variables.

The influence diagram (Howard and Matheson 1984) is commonly used to structure decisions by representing the relationship between key variables. An influence diagram consists of certain elements, as follows. Decision variables represent the decisions to be made (symbolized as rectangles in the diagram). Uncontrollable variables represent uncertainty or chance events (ovals). Outputs represent final consequences or payoffs (diamonds). Intermediary variables, including calculation nodes and constants, are used to compute the outputs (rounded rectangles). Relationships between nodes are represented using arcs. All arcs pointing to a rectangle (decision variable) show sequences. It means that the node at the beginning of the arc must be known.
before the decision can be made. All arcs pointing to ovals, diamonds or rounded rectangles (non-decision variables) show the relevance relations. The node at the beginning of the arc is relevant for the node at the end of the arc.

Figure 13.6 shows the representation of inputs and outputs component from an A&E department using an influence diagram. The output of the A&E simulation model is the A&E performance. The A&E performance is calculated from two intermediary variables, namely the total number of patients who spend four hours or less in the A&E department and staff utilization. The decision variables are the numbers of doctors, nurses and clerks. The uncontrollable variables (shown as ovals) are the arrival rate and severity of condition of the patients.

Figure 13.6  A&E – Influence Diagram (From Onggo, B.S.S., Journal of Simulation, 3 (1), 44, 2009. With permission.)

Contents component

Once the inputs and outputs have been specified, the next step is to specify the transformation processes or the contents. The contents component of a conceptual model describes the scope of the model, the level of detail, assumptions and simplifications. The scope of the model specifies
all relevant processes and their interactions within the boundary of the model. The level of detail specifies the required degree of detail for each process in the model and the required input data. Both scope and level of detail are determined based on the modelling objectives. Assumptions are necessary to address the uncertainty or unknown factors that may be important to the processes in the model. Simplifications are needed to handle the complexity of processes in the model. One of the possible diagrams that can be used to represent the contents component is the business process diagram (BPD) that was discussed earlier.

The scope of a conceptual model can be represented easily by specifying the relevant activities, events that start these activities, and the process flows (including decisions or branching of flows). Figure 13.2 shows the scope of a hospital simulation model which excludes the general practitioner (GP). Figure 13.2 also shows the level of detail of the processes. It considers A&E, outpatients and in-patients as three black boxes. It is possible to show a more detailed model for each activity in Figure 13.2. For example, Figure 13.7 shows a detailed model of the A&E activity in Figure 13.2. The figure shows that the process in the A&E department starts with patient arrivals. There are two types of patient arrival: voluntary and by ambulance. A patient who arrives voluntarily at the A&E department will need to register before being evaluated by a nurse (triage) to determine the severity of the patient’s condition. One who arrives by ambulance may, however, bypass registration (the triage is done on the way to the A&E department). Next, the patient will be seen and treated by a doctor and/or nurse (either in the resuscitation room or a cubicle). After treatment, patients will either be discharged or admitted to the hospital. Some patients may need tests and X-rays, and these patients then need a second session with a doctor and/or nurse before discharge or admission.
Figure 13.7  A&E – Business Process Diagram (From Onggo, B.S.S., Journal of Simulation, 3 (1), 45, 2009. With permission.)

BPD provides three artefacts that can be used to provide additional information about an activity that is not directly related to the structure of the process flow. One of them is text annotation which is suitable for representing the assumptions and simplifications used in the conceptual model. For example, in Figure 13.7, we can attach a text annotation to the activity ‘triage’ that provides a list of assumptions such as ‘the severity of condition of patients is modelled as a simple random sampling’. Similarly, we can attach a text annotation to activity ‘test’ that provides a list of simplifications such as ‘the service time for tests does not differentiate between the types of test (X-Ray, blood test, etc.)’.

Data requirement component

Data are an important part of any modelling technique. Hence, it is important to recognize the required data early. At this stage, given the previous problem-domain components, we should be
able to identify the data requirements. The required data should match the scope and level of detail of the conceptual model. Robinsons (2004, Chapter 7) discusses methods for dealing with unavailable data. The data requirement is often specified using textual representation. For example, Table 13.2 shows the data that need to be collected for entity patients in the A&E system in Figure 13.7.

**Table 13.2 Data Requirement for Entity Patient**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient details</td>
<td>Name, address, patient</td>
<td>This can be useful to identify patients and, if the analysis requires it, profile patients</td>
</tr>
<tr>
<td></td>
<td>identifiers, etc.</td>
<td></td>
</tr>
<tr>
<td>Admission time</td>
<td>Date/Time</td>
<td>This is needed to determine the distribution of admissions</td>
</tr>
<tr>
<td>Severity level</td>
<td>Minor or major</td>
<td>This is needed to find the proportion of patients needing minor treatment and major treatment</td>
</tr>
<tr>
<td>Time in A&amp;E</td>
<td>Minutes</td>
<td>This is needed to validate the output of the model</td>
</tr>
</tbody>
</table>

_Model dependent component_

As explained in the introduction, the method for representing model dependent components depends on the modelling paradigm. In the discrete-event simulation modelling paradigm, the
method should be able to represent key components such as: entities, resources, system states, queues, activities, events and processes. Onggo (2009) uses diagrams that are independent of any software implementation such as: ACD, BPD and ERG. A discrete-event simulation model that is represented using these diagrams can be implemented in any simulation software. It is relatively straightforward to develop software that is able to read a model that is represented using any of the implementation-independent diagrams and either simulates the model (for example: Araujo and Hirata 2004; Pidd and Carvalho 2006) or converts the model to specific simulation software (for example: Huang et al. 2007).

The representation of components in system dynamics (such as stocks and flows) has been influenced by the notation given by Forrester (1961). Nowadays, the stock and flow diagram and the causal loop diagram (both are independent of software implementation) are widely accepted as standards in representing system dynamics models (Sterman 2004). This explains why many system dynamics VIMS use similar diagrams to represent system dynamics models. In the agent-based simulation-modelling paradigm, a simulation model is formed by a set of autonomous agents that interact with their environment (including other agents) through a set of internal rules to achieve their objectives. Much of the literature represents agent-based simulation models using flow-charts or pseudo codes. The flow charts or pseudo codes are used to describe the internal rules of different agent types and the internal rules of a dynamic environment (i.e. its state is constantly changing, even if there is no action performed by any agent). Other than flow charts, we can also use a BPD or UML activity diagram to represent an agent-based simulation model where each agent type is implemented in a swim lane.
13.5 SUMMARY

We have discussed three categories of methods for conceptual model representation: textual representation, diagrams and multi-faceted representation. Textual representation can be used to give a brief description of a model. This is particularly useful when we have a repository of simulation models. The description allows others to decide quickly whether a model is suitable, or to search for the most suitable model to be used. The diagrams are effective during conceptual model development. A multi-faceted representation is the best representation for the complete documentation of a conceptual model. Multi-faceted representation has another advantage. It allows us to verify the consistency of conceptual model components (Onggo 2009). We have shown how to apply these methods to represent conceptual model components based on Robinson’s conceptual model definition. The same principle can be applied to other conceptual model definitions since most of the definitions have overlapping components. Although the author believes that the representation methods discussed in this chapter could be applied to many applications in discrete-event simulation, it must be noted that the methods have been tested using a few business process models only. As noted in Robinson (2002), simulation applications are far from homogeneous; hence, it is possible that some of the methods may not be suitable for some applications.

ACKNOWLEDGEMENTS

REFERENCES


Robinson, S. 2002. Modes of Simulation Practice: Approaches to Business and Military


