Proceedings of the 2002 Winter Simulation Conference
E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.

GUIDELINES FOR DESIGNING SIMULATION BUILDING BLOCKS

Edwin C. Valentin
Alexander Verbraeck

Systems Engineering Group
Faculty of Technology, Policy and Management
Delft University of Technology
P.O. Box 5015, 2600 GA Delft, THE NETHERLANDS

ABSTRACT

Instinctively, it seems better to support decision making by simulation studies carried out with domain specific simulation building blocks, than by simulation studies that start without the knowledge captured in these building blocks. However, only a limited number of project examples using simulation building blocks exist, which showed improved results as a result of the use of building blocks. We identified a number of requirements to overcome the problems in complex simulation studies. We believe that these requirements can be met by using building blocks and by carrying out the simulation studies in a predefined way. First of all a good building block architecture should be developed that supports the complexities in simulation studies. In this paper we will describe a design approach that in our point of view results in a usable set of building blocks. A proof of concept of the design approach and the architecture are given using a case for passenger modeling at airports.

1 INTRODUCTION

In contrast to what has been expected for a long time (Pegden et al. 2001), discrete event simulation is not used on a day-to-day basis by decision makers, not even for data-intensive decisions that require quantitative analysis. The term "decision maker" represents for us one or more experts in the problem domain who need the output from a simulation study to help them answering their questions. Many decision makers do not rely on simulation to provide them with the needed support, because simulation studies often demonstrated to be unable to adequately support them in making their decisions. Most often, this is because the level of support the decision maker needs in a certain stage of the decision making process cannot be provided by the simulation experts (regarding the duration of the project or the quality of the results).

In this paper we look at decision making processes that use simulation studies, and that have the following characteristics:

- multi-actor environment, in which different actors have different perspectives and interests
- simulation studies that normally take a couple of months to a year, due to the complexity of the environment, the number of processes and the required detail
- simulation studies that have a repetitive character and are used several times in decision making processes, not one-shot-models
- models containing repetitive parts such as resources and infrastructure for which several options need to be researched in the simulation study.

The simulation models that are used in studies like this are often large, touch the boundaries of simulation tools and include a lot of hard to understand coding, in best cases separated over different simulation submodels. When the model is partitioned, the important connections between different submodels often are not clear.

In decision making projects that use large simulation models, two kinds of problems can be recognized: organizational problems and technical problems. The organizational problems are for instance changes in the organization before the simulation study is finished and "language" differences between the simulation experts and the domain experts. An example of a technical problem is a lack of knowledge how to map a certain process within the problem domain onto a simulation language. Another complicating situation is when the first answers from the simulation models lead to new questions, which are usually hard or even impossible to answer using the developed models, because the simulation expert did not expect the questions. Many decision making processes have a low effectiveness due to these problems with simulation studies. The following performance indicators are used to determine whether

we succeeded in improving the effectiveness of decision making using simulation studies.

- Ability to answer new questions from problem owners, i.e. the number of questions that can be answered adequately within a certain time.
- Time needed to perform the study, i.e. total time for conceptualization, development of the model, verification and validation of the model and analysis of the outcomes.
- Effort required to translate conceptual models into a simulation model.
- Ability of simulation novices to perform the simulation study. We define a simulation novice as someone who has knowledge of a domain, but only limited knowledge of simulation. If it proves possible to train a domain expert to develop models of their problem situations, no knowledge is lost in the communication process between domain expert and model developer.

In section 2 of this paper, we will work out a number of requirements for complex simulation studies, which fit the characteristics mentioned above. In the third section we will briefly explain the proposed architecture for building blocks. The use of these building blocks will improve the effectiveness of the simulation studies, but requires a problem driven design approach, which is presented in section 4. In section 5 it is described how this design approach is applied for passengers at airports. Finally section 6 will look forward to new research in this domain.

2 REQUIREMENTS FOR COMPLEX SIMULATION STUDIES

Those who study the literature on simulation will notice that there are a lot of methods for carrying out a simulation study, see e.g. Banks (1998), Kelton et al. (1998), and Law and Kelton (1999). These prescriptive methods form the basics of any simulation study: start with a problem definition, make a conceptual model of the system, implement a simulation model in a simulation environment, gather the data, perform experiments including validation and verification, and analyze the outcomes. In the available literature, each of these steps is explained in detail, aiming at providing instructions for engineers who are new in the field of simulation. Kelton et al. (1998) use for example a post office single-server simulation study to explain each of the steps. Unfortunately, fewer methods and examples exist that provide guidelines for performing complex studies with the characteristics described in section 1. Each simulation expert should find his or her own way to handle these kinds of studies and to translate the introductory explanations into guidelines for complex situations. In this section we show some of the general requirements for complex studies based on real-life cases. Detailed requirements and solutions will be given in sections 2.2 and 2.3.

2.1 Case Example Passenger Modeling

We take a close look at a simulation project for passengers (LOT = Logistic Operation Terminals) at Amsterdam Airport Schiphol (Gatersleben and Weij 1999), to explain which challenges we face in complex simulation studies. In this simulation study several actors play a role, each with its own goals. Examples are the management of Schiphol that is aiming at a better service level for the passengers but that also has to pay for these extra services, airline companies with the desire that passengers can easily transfer between flights, and the KMAR (Dutch border control) that is responsible for the safety and border crossings at the airport.

This simulation study was started to evaluate the capacity of the airport terminal for the year 2020. Possible design solutions to match the required performance indicators were expansion of the number of available gates, new lounges, and different ways of check-in methods and passport checks for the passengers.

The initial simulation models have been developed in the simulation language Arena and figure 1 shows a screen-dump of the model logic of one of these models. The screen-dump shows that the model logic is hard to interpret and hard to judge, due to size and complexity. Model runs also tend to be slow, and it is hard to find where improvements can be made, because there is no easy way to replace parts of the model by reduced submodels.

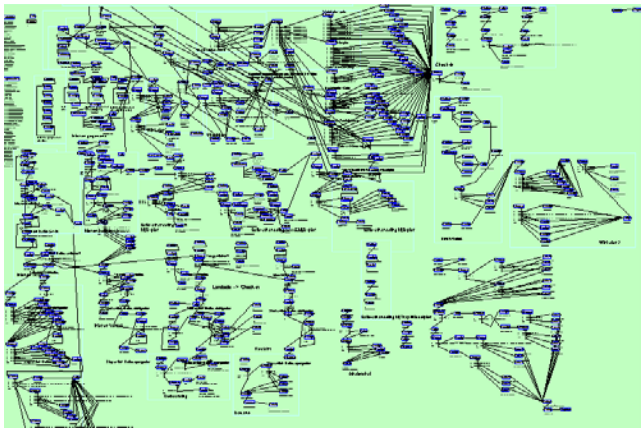

Figure 1: Logic Screen-dump of LOT Simulation Model

Two requirements can be abstracted from this view:

- *Requirement 1: Provide an architecture that is able to handle the complexity we want to model, including support for different levels of reduction.*
- *Requirement 2: Even though the model should support a high level of complexity, a recognizable and flexible model structure should be provided for maintainability.*

Each complex model can be made more understandable for the domain expert by adding visualization. In the case of the passengers at the airport, however, the chosen implementation prohibited a flexible animation, other then dots moving in a straight line behind each other. As a result, the domain expert had a hard time understanding the model and he did not have much faith in the way the model was implemented.

- *Requirement 3: Use concepts that represent functionalities as found in reality and that can also be used for visualization purposes.*
- *Requirement 4: Visualize a system in such a way that complexity is reduced but that the essential processes are still shown.*

After some demonstrations and discussions, the models were accepted by the domain expert and results of the study could be analyzed. The analysis was the basis for experiments to be performed. Some of the experiments had to do with the availability of resources (extra gates, extra seats, extra walking spaces in passageways), some with the routing of passengers, and some with the way resources are allocated (mainly check-in counters and gates). At a rather late stage, it turned out that one possible experiment should be to add a new corridor for passengers. Unfortunately, this small change in infrastructure was extremely hard to implement in this simulation model. As a result this experiment was not taken into account.

The experiments regarding the alternatives for the expansion of the infrastructure had comparable modeling problems. However, these experiments were too important for the domain experts to be ignored. Several model developers have been trying to change the model in such a way that the experiments would be valid, but this took more time than the initial model development effort.

The last experiments regarding the resource allocation influenced several parts of the model (the use of the resources, the definition of the resources and the allocation mechanism itself.) The allocation mechanism communicated with all these model parts and retrieved and wrote information to these parts whenever necessary. As a result, the interaction between the model parts was very complex, and the model crashed as soon as we tried to replace a part of the allocation mechanism.

Based on these experiences the following requirements can be identified:

- *Requirement 5: Model should be open for changes for future experiments.*
- *Requirement 6: Interactions between model parts should be explainable afterwards and should represent interactions in the real system.*
- *Requirement 7:The effort to support an analysis of radical design changes should be such that the design process is not delayed.*

2.2 Analysis of the Airport Example

The identified requirements were illustrated using the case of a modeling project for Schiphol Airport, but most of the mentioned problems are taking place in every complex simulation study. With every large simulation model of some complexity there usually is a problem in adjusting the model for a variety of experiments.

The standard literature regarding simulation studies (e.g. Banks 1998, Kelton et al. 1998) provides some tips and "do's and don'ts" but most of the tips are for simple examples such as single-server models, post offices or a factory with a limited number of machines. Standard literature does not provide an explanation how to model *complex* systems resulting in *structured* models.

One reason for the lack of structure in models and tips how to reach a structured model, is the lack of possibilities in COTS (common off-the-shelf) simulation tools to create structured models. The implementation of hierarchical models in some simulation languages can improved the model structure, but in practice, hierarchical models are "content-hidden" models, in which the unstructured coding is hidden at a layer which is one level beneath the model overview.

Another solution offered by commercial simulation tools, is based on reuse like often used in software engineering and component based thinking. Several simulation environments offer the ability to use predefined elements that can be used in several simulation studies or several times in the same model. This improves the structure of the model thanks to hiding the repeating code within such a simulation element. Some simulation tool vendors are convinced of the generic applicability of their sets of elements for a special domain, and sell them to their customers. Some examples of these sets of elements are transporters and conveyor belts within the simulation packages Arena (Rockwell Software, 2002) and eM-Plant (Tecnomatix, 2002) , PowerAndFree or Automatic Guided Vehicles in the simulation package Automod (Brooks Automation, 2002) and MedModel of the simulation package Promodel (Promodel, 2002). These sets of elements hide the low level of coding within a simulation environment, with the result that model construction is much easier and faster, and that simulation novices (yet experts in the domain) can perform the simulation studies. These sets also ensure the use the domain specific terminology and have domain specific user interfaces.

These sets of domain specific elements solve a couple of problems, but not all as the Schiphol example shows. The ability to easily create models for new experiments is available, and the execution of the experiments is much easier, as long as the experiments are nothing more than

adjusting the values of some parameters of the elements or replacing one element by another element from the library. As soon as an experiment needs to be performed that can not use the pre-defined set of elements, it is the end of the story. An important reason of the limited support of changes, is that the elements are usually stemming from one or more successful simulation studies in that domain with a set of specific questions. In these studies the models have been developed in a hierarchical and modular way and the different layers have been migrated to an element by providing a domain specific user interface. As a result the sets of elements for many domain specific simulation libraries contain simulation elements that have been developed for a small number of questions, and the design of these elements does not contain the rich set of solutions needed by successive simulation studies.

2.3 Solution: Simulation Building Blocks

In our view, domain specific simulation elements should be more than reusing the proven code of previous projects. Elements should be defined from a problem solving perspective, so they can contribute to improve the effectiveness of simulation studies. In addition, elements should also include more of the component based software developments (Szyperski 2001). However, the components described in literature are focussing at software development, they do not contain a reduction or an abstraction of reality and as a result they are different from the elements needed in simulation studies. Therefore, we will use the term "building blocks" in this paper instead of "components". These sets of building blocks should support a more effective way for simulation to contribute to the decision making processes in organizations.

As these building blocks are different from components, a new architecture is needed as well. The architecture describes how we take care of the technical requirements posed earlier in this paper. A summary of the proposed architecture is described in section 3.

The process oriented requirements are answered by paying attention to the adaptation of building blocks for different simulation studies. The design process for sets of building blocks starts from a problem oriented view. In section 4 the design process for building blocks is described in more detail.

3 SIMULATION BUILDING BLOCKS AS SOLUTION

3.1 What are Building Blocks

We see building blocks as the basics structure for future simulation models. We aim at models that are constructed only out of building blocks, if necessary in close interaction with the concepts available within a simulation environment. Building blocks have in our point of view the following characteristics that describe what a building block should be:

- self-contained (nearly-independent),
- interoperable (independent of underlying technology),
- reusable,
- replaceable,
- encapsulating its internal structure,
- providing useful services or functionality to its environment through precisely defined interfaces,
- customizable in order to match any specific requirements of the environment in which it is used (plugged).

This list is based on results from the research program BETADE (Verbraeck et al. 2002), which pays attention to development of software and models based on building blocks. In the following subsection we will describe in more detail how we incorporate these characteristics of building blocks into commercial available simulation tools.

3.2 Architecture for Building Blocks

In previous publications we explained the architecture we have in mind for building blocks (Verbraeck and Valentin 2001, Valentin and Verbraeck 2002). The most important characteristics of this architecture are domain specificity and composition.

The domain specificity is reached by structuring the domain under study using the view of the decision maker. This leads to a list of objects the decision maker directly notices when he/she is describing his problem domain. E.g. a decision maker looking at passenger flows at airports would mention "check-in counter", "gate" and "passenger" as possible building blocks. We call the simulation representation of these objects *model building blocks* (MBB). The domain specificity is not only shown by the identification of model building blocks, but also by using the terminology of the domain and a visual representation the way the decision maker expects it. This domain specificity is thus integrated into the structure of the model (model building blocks), the parameter input (terminology), and output (results and visualization).

Over-sizing of model building blocks and thus implementation of overhead, which introduces functionality not needed in certain experiments, is avoided by dividing model building blocks according to the functionality they need to represent. Each functionality that is likely to be changed during an experiment is provided as a *building block element* (BBE). Figure 2 shows an example of two different model building blocks (XYZ and XYZ'), using different building block elements (A-1; B-1; C-1 or C-2).

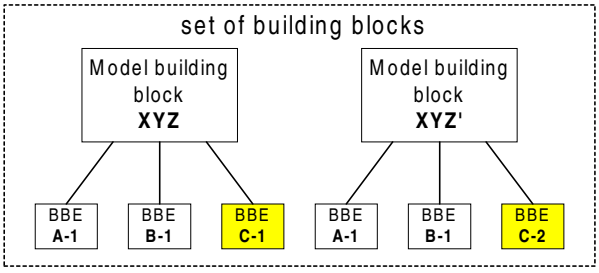

Figure 2: Example of a Set MBBs using BBEs

4 DESIGN APPROACH FOR DESIGNING BUILDING BLOCKS

As explained before, most of the elements that are used for model development by the simulation vendors are coming from successful simulation projects. This leads to a limited number of problem situations that can be solved using the set of elements within a domain. In this section a four phase design approach will be sketched that avoids this problem. Phase one will take care of identifying the problem domain and the problems worth modeling; phase 2 will be used to perform an object oriented analysis, and the outcome will be turned into the model building blocks and building block elements in phase 3. Finally in phase 4 the MBBs and BBEs will be implemented in a simulation environment. Added to the design approach a few practical points are added in sub-section 4.5.

4.1 Phase 1: Conceptualization of Problem Domain

The first step to be performed in defining a new set of building blocks to support simulation studies in a problem domain, is to limit and describe that domain. This domain can be best described by comparing the conceptual models of different simulation studies. Banks (1998) describes how a simulation study should be performed, starting with a problem definition to get to a conceptual model. The result of this step will be a list of problem descriptions that fits the domain. The selection whether a problem fits in a domain and whether it is worth taking into account while developing the set of building blocks is an economical selection and it should be well-documented.

The list that will be created initially will probably be too large to develop building blocks that will support all identified problems. The 80/20 rule is relevant here. The goal is to develop that 20% of the building blocks that can support 80 percent of the problem situations. In addition it should be extendible later to cover more exotic problems that happen less often.

It is suggested to use more then one case to check whether the problem definitions are representative for the kind of studies that need to be performed in the future use of the building blocks. Ideally, these cases should cover all the different activities that occur within the problem domain. However, here again the 80/20 rules should be taken into account and besides, it is impossible to find a case in reality that covers all different activities within the domain but is still small and easy to interpret. A workable alternative is to define a number of example cases that contain a number of complicated issues of the problem domain in a realistic way. This selection of cases is important, as in every stage of the development process, these cases will be used to judge the quality of the building block library.

Based on the description of the problem domain and one or more example cases, a list of desired experiments needs to be defined. These experiments are expected to be performed during the execution of the simulation studies, using the building blocks. These experiments will form the starting point to define the required output of simulation result and to enable evaluation of the different solution alternatives of the problem under study.

Before this phase will be finished, the building block developer should check whether the example case studies can be carried out, e.g. by checking whether a rich set of input parameters is available to develop models for the example cases. It is also good to challenge the first design in a small team and search for the boundaries of the concepts and see if and how they can be widened. A similar process should be performed for the identified output categories, and one or more real decision makers might be involved to identify shortcomings.

4.2 Phase 2: Analysis of Domain as seen by Domain Expert

In this phase it will be analyzed what parts of the problem domain need to be modeled. A reduction of complexity will be performed based on an object oriented analysis (Meyer 1997) and on process flow descriptions. It is important that for this phase the viewpoint of the domain expert will be used, to ensure that the domain expert will be able to recognize the models.

An object oriented analysis can be carried out in several ways. We suggest to use standard approaches like described by Rumbaugh, Booch and Jacobson (1999) in their UML modeling approach. This object oriented analysis will result in a list of objects that are important within the problem area, seen from the domain expert, so it will not focus on the topics interesting for simulation models, but deliver a detailed static representation of the domain.

The process analysis is a second way to reach the same goal. A part of the flow is already described in the UML-diagrams but to develop simulation building blocks, more detailed insight is necessary in the relation between processes. Preferable this is not done based on the objects, but from a different viewpoint. These process schemes should not just describe the physical flow, but also the information flow that is required for allocation mechanisms that are part of the system.

The example cases should be used to check whether the defined processes and objects provide a good overview of the problem domain. Experience showed that building blocks like generators and control mechanisms are often not included. The reason is that the objects have been defined from the viewpoint of the domain expert and the expert will for example describe a door, which will be replaced in the simulation by a generator of passengers. Also additional objects that need to provide statistics and other output values are often forgotten, but a structured walkthrough based on the identified cases and output needed should solve this problem.

4.3 Phase 3: Building Blocks

Probably the hardest phase is the translation of the objects and processes into building blocks. In this phase choices are often made that limit the extendibility of the building blocks and/or lead to building blocks that cannot support alternative solutions. To avoid this problem as good as possible, the design of the building blocks should be done very carefully, and usually in an iterative process. The MBBs (model building blocks) should be a direct translation of the defined objects in phase 2 because MBBs aim to represent the world-view of the domain expert and the objects are defined from that viewpoint. The example cases should be evaluated with the identified MBBs immediately. If the identification of the MBBs has been performed correctly, models for the example cases can be easily developed on paper from the identified MBBs. It should of course be kept in mind that this a conceptual evaluation without the real simulation environment available, but it is a first good evaluation of the "strength" of the MBBs.

The BBEs (Building block elements) are seen in our point of view as the needed internal and external functionalities of the MBBs. We would get too many different BBEs if each functionality would be seen as a separate BBE, so we use the following rule in our definitions: *"is it likely that the model developer wants to change or replace this functionality in some kind of experiment?"* If yes, this shows that different variants of a functionality might be available, resulting in different BBEs that are easily replaceable or changeable.

With the definition of the MBBs and the BBEs we also look at the interaction *and* information exchange between the MBBs, and between the BBEs within the MBBs. It is important to define the information exchange, because it is the information exchange on which new BBEs or MBBs should standardize.

4.4 Phase 4: Implementation

The implementation of the building blocks is a phase that has been carried out too early in many studies. Often the implementation was started as soon as a first idea about the need and the kind of building blocks was available, just to see whether it works or not. This results in an implementation phase that takes much longer than expected, because every time new building blocks need to be added that do not fit the structure. We found out in our projects that postponing of the implementation till the moment the building block definition is finished, results in a faster implementation of the building blocks.

This fast implementation can be accomplished, due to a careful mapping of the building blocks to the available concepts in the simulation tool. This step is less difficult in an object oriented modeling environment such as eM-Plant, but is more important for flow oriented tools like Arena. The concepts of Arena are very different from the object oriented descriptions developed in phase 2 and 3, but close to the process description that also has been created.

As soon as it is known which concepts should be implemented to represent the building blocks in the simulation language, an implementation plan can be made. In this plan it will be defined in which order each functionality need to be modeled and how it can be tested in demo models. A black-box approach is recommended. This means that each important building block is modeled, but that only the functionalities needed to get a working building block are implemented in detail to start with. This will lead to building blocks that can be used early in this phase to do some first modeling activities, like developing test models and getting insight in the benefits or weaknesses of the building blocks regarding visualization, representation, ease-of-use, output, and use in model development process. If necessary new experiences can lead to changes in the building block design.

4.5 Guidelines by Adapting the Design Approach

In addition to the four phases described, we have gained some experiences that are important to take into account during the process of designing building blocks.

4.5.1 Iterative Process

The phases are described in a sequential order here, in which one works from milestone to milestone, but in reality the process is iterative and one should allow in the project to make changes to decisions in earlier phases when additional insight is gained in later phases of the project.

4.5.2 Building Block Designer should have a lot of Experience in Modeling

The design process of building blocks requires a choice for a clear architecture. In simulation models, the selected architecture does not matter that much, but in building blocks, this choice is far more important. Most often there are many different ways of modeling a situation, all with

their own advantages and disadvantages. One alternative might be very fast, but unfriendly to the user and/or hindering modularity. The building block designer should be capable of making a well based selection for selecting modeling concepts from the simulation language used, and only developers who know the simulation tool very well will be capable of making these selections.

4.5.3 Cooperation with Domain Expert

One of the goals of the approach is that by using the building blocks, the domain expert will recognize his problem situation in the models. This requires that the domain expert knows about the design choices, especially because the building block designer most often is not an expert in the domain. The building block designer is an expert in the field of simulation and needs some background in the domain to be able to communicate with the domain expert, but does not need years of experience in the field.

4.5.4 Check, Check, Check, Check

In each phase the example cases should be reused to check each design decision for correctness. These checks are primarily paper-based checks and do not have anything to do with implementation in the simulation study. As soon as example cases cannot be modeled anymore using the building blocks, a wrong design choice has been made and should be reversed. E.g. the decision to group several functionalities in one building block element, while example case studies show that several experiments with an individual functionality need to be performed.

4.5.5 Use of the Set of Building Blocks in Simulation Studies

The preparations of the use of the set of building blocks in a real simulation study could be seen as a fifth phase. In this phase a user-manual should be developed, as well as example models and training material that show the abilities of the building blocks. To transfer the building block library to the domain, a close cooperation between the building block developer and the model builder during the first project carried out with the new library might help. Other transfer mechanisms can be used as well. The transfer-process is important and the set of building blocks should not be "thrown over the wall" with the model builder at the other side of the wall having to find out the guidelines for use himself.

5 EVALUATION CASE

Section 2 of this paper described the types of problems that occur in complex simulation studies. In this section we want to show how we have been developing building blocks for the airport problem domain and how well these building blocks helped in solving the problems. More detail about the building blocks for airport passengers can be found in the paper of Verbraeck and Valentin (2002) which describes the use of the building blocks for airports terminal and passengers with example studies at Amsterdam Airport Schiphol and JFK Airport.

5.1 Phase 1: Conceptualization of Problem Domain

The domain description and boundary setting have been based on several projects carried out for airports, e.g. the ones described in Babeliowsky (1997), who also described the first implementations for the LOT studies. Based on the experiences described by Gatersleben and Van der Weij (1999) three different kinds of model change requests are defined:

- Change of the available capacity (space for passengers to stay) for the long-term
- Process changes for the passengers at airports
- Changes of allocation mechanisms of resources (E.g. check-in counters that are dedicated for an airline)

The example cases that are used for the development of building blocks are based on the LOT-study Schiphol (Gatersleben and Weij 1999), but separated into several smaller modeling problems to reduce complexity.

5.2 Phase 2: Analysis of Domain as seen by Domain Expert

The object model of the airport contained at first over 50 different objects. With a generalization we reduced this amount to three main objects: Area (a location where passengers can stay), Group (one or more passengers that move from area to area with a certain goal), and Script (process description of activities for different kinds of passengers). These three objects are the very generic "concepts" on which a model of an airport terminal can be based. Of course, each of these generic objects needs to be worked out into more detailed building blocks later.

Besides these three objects, we needed generators that generate groups of passengers based on flight schedules, and control objects that control the capacity-use of gates, check-in counters, and luggage-reclaim locations.

5.3 Phase 3: Building Blocks

The model building blocks that have been defined are based on the object analysis. In our case, many different versions of MBBs have been defined, each based on the generic concepts of area, group, script, control mechanism, and generator.

Each MBB consists of different BBEs, especially the area contains a large set of BBEs. Some example BBEs of the Area are: capacity claiming of the area, occupation time of a group, statistical calculations and overall control (link with different control allocation mechanisms).

5.4 Phase 4: Implementation

The implementation of the building blocks has been performed in the simulation environment eM-Plant. Thanks to the object oriented character of the simulation tool only a few conceptual translations were necessary, but the area, group, script, generators, and control blocks could be used and can still be recognized in the toolbox.

5.5 Evaluation

By now, several simulation studies have been performed using the set of building blocks. So far the set of building blocks have been mainly used for future design processes, in which the building blocks enabled changes of the models. Studies carried out were modeling of the overall passenger handling processes at Amsterdam Airport Schiphol for a capacity study on a busy day, a detailed study for new passport control, and a project for allocation mechanisms for flights at the check-in counters of KLM. The building blocks have been used for JFK Airport to determine the number of needed check-in counters and to evaluate and improve the control mechanisms of allocating airlines to counters. Each of these cases fitted the description of complex systems as defined in section 1. As a result, we could see how these studies behaved with respect to the different requirements as described in section 2 of this paper.

Requirement one and two have to do with the model structure. Technically this has been solved by separating the infrastructure (the area) from the processes (the groups and the control mechanisms). In the areas BBEs are included, which represent different control mechanisms. This design selection resulted in a well-maintainable and well structured architecture.

Requirement three and four focus on the visualization for the decision maker, including the output. First of all, early in the design process the required output was defined, which was easy to identify in the processes of the area, in the activities of the group, and in the relations between groups and areas. We also decided to show a moving animation of groups and passengers. In the first implementation of the animation BBE for the areas, groups walk over the area and wait at the end of the area to enter the next one. Using a different animation BBE, a more advanced visualization could be shown, where the areas were reproduced on top of a bitmap that has the form of the airport terminal (based on a CAD-drawing) to show exactly where passengers are at a certain moment in time, without abstraction or reduction. Using this animation, the decision maker can recognize the queues of passengers he/she notices in his airport terminal in reality. Making a change like this only required replacing one BBE.

The last three requirements (5, 6, and 7) regard the ability to change models in order of experiments. Already in the first step of the design process it was taken into account which future experiments we expect. In each of the following design steps it was checked, whether our decisions did not prohibit the experiments from being carried out. Mainly because of the separation of area and passenger (processes) changes are easy to make. Further changes regarding the *behavior* of areas are even easier to make by just selecting a different building block element for the area.

6 CONCLUSIONS

In the introduction of this paper we indicated how hard it is to carry out a complex simulation study and satisfying a decision maker at the same time. Building blocks have shown their benefits in enabling this and facing other related requirements such as reusability and maintainability of simulation models. The used architecture provides us with an ability to develop models and maintain those models within the specified problem domain. In this paper we have also shown the importance of the design process for building blocks with relation to the ability to modify models for extra experiments.

The development of building blocks is a complex activity, and it differs from carrying out a “normal” simulation study. The phases we have shown in this paper provide a first set of guidelines for developing libraries of (re)usable simulation building blocks, that proved to work in real projects. The projects for which we researched the building block approach came from a domain where we already had experiences with a traditional way of simulating without building blocks. This gave us an ideal insight into the added value of the use of building blocks, both for the modeler and for the decision maker.

REFERENCES

Babeliowsky, M.N.F., Designing interorganisational Logistic Networks: A simulation based interdisciplinary approach, Doctoral Dissertation. Delft: Delft University of Technology, 1997.

Banks, J. Handbook of Simulation; Principles, Methodology, Advances, Applications and Practice. John Wiley & Sons, 1998.

Brooks Automation, web page: www.automod.com, visited 09-08-2002, 2002

Gatersleben, M.R.; S.W. van der Weij. “Analysis and simulation of passenger flows in an airport terminal”. In: P.A. Farrington, H.B. Nembhard, D.T. Sturrock and G.W. Evans (eds). Proceedings of the 1999 Winter

Simulation Conference, Piscataway, NJ: IEEE, 1999, p. 1226-1231.

Law, A.; D.W. Kelton. Simulation Modeling and Analysis. McGraw-Hill, 3rd edition, 1999.

Kelton, W.D.; R.P. Sadowski; D.A. Sadowki. Simulation with Arena. McGraw-Hill, 1998.

Meyer, B. Object-oriented software construction. Prentice Hall; 2nd edition, 1997.

Pegden, C.D.; V. Kachitvichyanukul; J.O. Hendrikson; R.G. Ingalls; B.W. Schmeiser. "Simulation environment for the new millennium (panel)" In: B.A. Peters et al. (Eds.) Proceedings of the 2001 Winter Simulation Conference, Piscataway, NJ: IEEE, 2001. p. 541-547.

Promodel , web page: <www.promodel.com>, visited 09-08-2002, 2002

Rockwell Software, web page: <www.arenasimulation.com>, visited 09-08-2002, 2002

Rumbaugh, J.; G. Booch; I. Jacobson. The Unified Software Development Process, Addison-Wesley Publishing, 1999.

Szyperski, C. Component Software, 2nd edition. Addison-Wesley Publishing, 2001.

Tecnomatix, web page: <www.eM-plant.de>, visited 09-08-2002, 2002

Valentin, E.C.; A. Verbraeck. "Simulation using building blocks". In: F. Barros ; N. Giambiasi (Eds.) Proceedings conference on AI, Simulation and Planning, San Diego: SCS, 2002, p. 65-70.

Verbraeck, A.; E.C. Valentin. "The use of building blocks to enhance flexibility and maintainability of simulation models and simulation libraries" In: N. Giambiasi, C. Frydman (Eds.), Proceedings ESS'2001, Gent, Belgium: SCS, 2001, p. 973-979.

Verbraeck A.; Y. Saanen; Z. Stojanovic; E. Valentin; K. van der Meer; A. Meijer; B. Shishkov. "Chapter 2. What are building blocks?". In: A. Verbraeck and A.N.W. Dahanayake (eds.). Building blocks for Effective Telematics Application Development and Evaluation, Delft: TU Delft, 2002, p. 8-21.

AUTHOR BIOGRAPHIES

EDWIN C. VALENTIN is a researcher in the Systems Engineering Group of the Faculty of Technology, Policy and Management of Delft University of Technology. His specialty is the development of domain-dependent generic discrete-event simulation libraries. Edwin participates in the BETADE research program on developing new concepts for designing and using building blocks in software engineering, simulation, and organizational modeling. His email and web addresses are <edwinv@tbm.tudelft.nl> and www.tbm.tudelft.nl/webstaf/edwinv

ALEXANDER VERBRAECK is an associate professor in the Systems Engineering Group of the Faculty of Technology, Policy and Management of Delft University of Technology, and a part-time full professor in supply chain management at the R.H. Smith School of Business of the University of Maryland. He is a specialist in discrete event simulation for real-time control of complex transportation systems and for modeling business systems. His current research focus is on development of generic libraries of object oriented simulation building blocks in C++ and Java. His email and web addresses are <a.verbraeck@tbm.tudelft.nl> and www.tbm.tudelft.nl/webstaf/alexandv