4. MODEL VALIDATION

4.1 Introduction

In every simulation experiment a large number of invalid models are constructed. This fact has many negative impacts on the reputation of manufacturing simulation. It also increases the costs of performing simulation experiments, makes them less useful, and makes it difficult to believe in the results. The effects it has are different depending on at which stages the models are identified as invalid. This section deals with these effects, what can be provided to avoid them, and what tools to help the user to avoid them, have been implemented in SIMMEK. All this is explained in the context of manufacturing simulators. But first there is a need to present a clarification of how the terms verification and validation are used in this context.

4.2 An approach to verification and validation

There may seem to be a conflict between our approach to and use of the terms verification and validation of simulation models in this paper, and that of traditional simulation theory, as described in among other these publications [4, 17, 18].

Verification	-	To check whether the computer simulation program and the model do what it is intended to do
Validation	-	To check whether the simulation model is a good (enough) description of the real system

By an extended use of existing simulators, and also by developing a manufacturing simulator at our laboratory, we have learned that much of traditional computer program verification job must be, and is done, by the persons developing the simulator. The end users have no need, or maybe more correct, often no possibility to do computer program verification. The rest of verification job, i.e., to assure yourself that the computer model does what you think it does, is not so unlike the validation job. The tools and techniques are very much the same. Therefore we here speak about model validation, although a part of what we are dealing with is model verification.

4.2.1 Model verification

The problem of verification of a simulation model when using a manufacturing simulator is heavily dependent on at least the two following elements of quite different character;

* The quality of the user manual
* Understanding the difference between the strict logic of a computer program and the human brain and behaviour

The quality of the user manual is so important because quite a lot of the behaviour of the different objects is built into the system, and it is not possible for the user to change this behaviour. This means that the user manual must have a very precise and detailed description of the different functions.

On the other hand, no matter how good the user manual is, there is a danger that the user does not understand the logic of the system. And someone that has been working with a manufacturing system for many years, knows every detail of it, every peculiarity and how these are handled. When asked to describe it, there is a danger that the small control routines that really make it work are omitted. And that the fact that although the system is normally controlled

by a set of rules, there is another set of rules being applied "when things are out of hand", is forgotten.

4.2.2 A completely valid model is not possible to achieve

The very strong statement in the section heading should not be misunderstood. And spending one whole section on model validation hopefully indicates further how important model validation is considered.

But nevertheless, it is important to be aware of the following three facts;

* A one-to-one relation between all elements of the real system being modelled, and the simulation model is not possible

* Models often pictures systems, or versions of systems, that do not exist

* The use of uncertainty (statistical distribution) in the models, makes the probability that the same sequence of events should occur in reality equal to zero

Remember that the conclusion from a simulation experiment is NOT; the simulated result will be exactly the result of the real system. But it may be; the simulated results are likely to occur. Or; it is likely that the real system performance will be within +- x % of the simulated results.

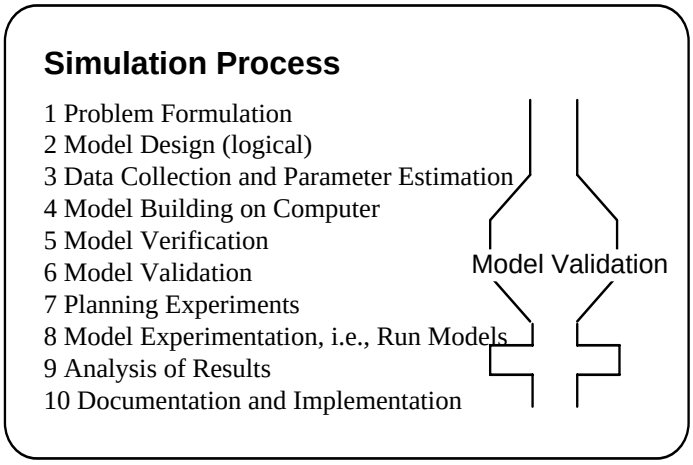

Figure 4.1 Validation in the simulation process

Partly following from this, we have the approach that model verification and validation are something done all the way from deciding to using simulation, either using a programming/simulation language or a simulator, till the conclusion from the simulation experiment is made. This is very close to Carson's approach in [18], although he distinguishes more clearly between verification and validation. But model validation and verification have, of course, an intensive phase as it is tried to illustrate in Figure 4.1.

A third thing to point out is that the validity of a model is measured in which degree the model is capable of describing the real or designed system, considering what the objectives of the experiments are. This is shown in Figure 4.2. The sizes of the boxes in the figure is meant to illustrate that the "area of validity" is decreasing, the important question being whether it is valid on the topics under examination. The names of the arrows show the main things to consider at each transformation step. The objectives should be considered at all these steps.

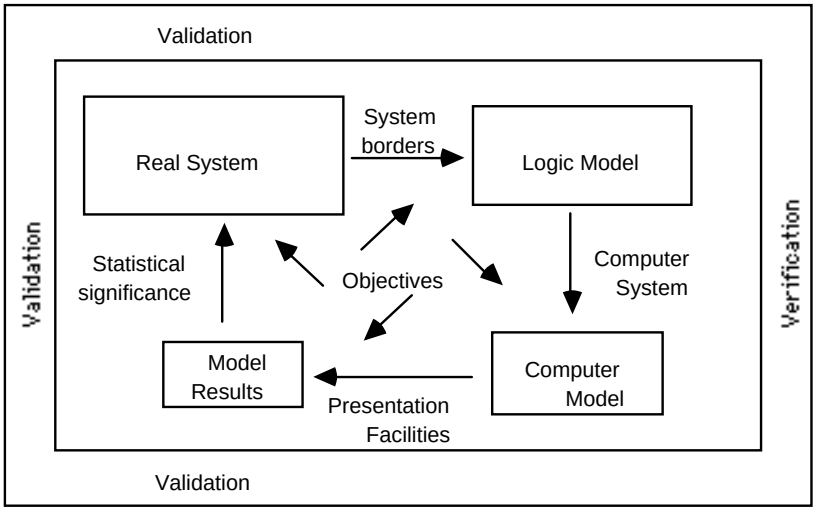

Figure 4.2. Validation and objectives

4.3 The consequences of invalid models

The consequences of making invalid models are mainly of two types,

* Wrong conclusions
* Increased costs

These consequences will in turn lead to

* Reduced use of simulation

Costs include time spent both by the modeler and the computer, and direct costs connected to the extra effort put into model building and experimentation.

Invalid models at early stages of an experiment, i.e., model building phases, are costly, but rarely fatal at this stage. But it may, of course, be fatal if it is not discovered at this stage.

The modeler uses a lot of time and effort on correcting errors of many different types. Errors which are strictly typing errors are annoying, and can be costly if the simulator has a strict path which the modeler must follow. Invalidity of models caused by the modelers' lack of knowledge of the simulators' interface or modelling possibilities are more rare but may be fatal. A possible

consequence of these early mistakes is that the modeler may loose confidence in either the method, simulation, or in the chosen simulator.

Invalidity of models at stages where the model is running, but the process is still in the traditional validation phase (although the user may not be aware of it !), is often not as costly, but may be more fatal. A model which may be 99% "correct", but that remaining 1% makes the results crazy. In many cases this leads to the conclusion that simulation is a useless technique for doing analysis of manufacturing systems.

Invalidity of models at the conclusive phases is, of course, both fatal and costly, especially when the wrong conclusion is made. Fatal both because of the effects of the wrong conclusion, and because these persons will probably never use simulation again.

All these, and probably many other negative effects that I have not covered, all clearly show that proper tools for doing model validation are a necessity in any simulation tool. In fact it will in many cases be fatal if they are not present or are inadequate.

4.4 Model validation - A key to extended use

In our work in the manufacturing simulation field we have tried to figure out why simulation is not as much used as it could be, considering the possibilities of this method. Our conclusion has been that most tools are not easy accessible, and it takes too long time to learn to use them. In this background we have decided to develop a manufacturing simulator with a graphical environment where no programming is needed.

We have also learned that by presenting users with such a tool which is easy to use, the validation job may be neglected. End users seem to be less critical to models constructed through such a modelling environment, than to a model they have programmed themselves in a programming or simulation language. Therefore we have concluded that such simulators must have extended facilities for performing model validation. And this validation must be done, partly automatically, all the way from the data collection phase to the decision making phase. And the user must be stressed how important this job is.

If the job is done properly, it will eventually increase the percentage of valid models, which again will increase the number of successful simulation experiments.

4.5 Invalid models

As described earlier we consider model validation as something to be done at a certain stage of an experiment, but should also be taken care of at all the other stages. And considering manufacturing simulators, verification is here included as a part of validation.

It is interesting to establish the most important reasons for models being invalid. Again we limit ourselves to dealing with manufacturing simulators. A first distinction is made between;

* User caused invalidity
* Invalidity caused by simulator

In both these two groups of causes we find a number of mistakes, pitfalls or "be aware signs".

The most important user caused reasons of invalidity are;

* Models are too much or not enough detailed

*	Typing-editing errors
*	Logical errors
*	Incorrect use of statistics

Table 4.1 User caused reasons for invalidity

On the other hand, the simulator package itself may contain errors or bugs that are hidden from the user. These are;

*	Program bugs
*	Logical errors
*	Incorrect computations
*	Incorrect use of statistics

Table 4.2 Simulator package caused reasons for invalidity

4.6 Model validation aspects at the different stages of simulation experiments

It is crucial that the objectives of the experiment are addressed when establishing a model. If this is not done, lots of effort are put into making parts of the model detailed, parts which could have been modelled as black boxes. This means that the system borders must be drawn carefully, and the input and output through these borders must be measured (existing systems), estimated (designed systems), and calculated with accuracy. And what happens inside these black boxes are not considered unless it in any way affects the input or the output from the modelled system.

The process of establishing a model with the purpose and objectives in mind is probably the most difficult job for an inexperienced user of simulation. This is really concerning the *Art of Simulation*, and can only be learned by repeated trials and failures, and eventually success. An interesting experience we have made from performing experiments is that the current model is often valid and credible in the context of the objectives of the previous experiment. It is a human habit to do things "the same way we did it the last time we did something like this".

Having established the experiment objectives, and by applying these on the system when establishing the model borders and main elements, the data collection job following is always a time consuming process. It should be done with great care, because invalidity of models may be caused by the input data in at least three different ways.

They may be typing errors occurring in the handling of large amounts of names and figures. Such errors are trivial to track if you just have the stamina in the searching process.

In most cases data is used to find the type of statistical distribution to use, and which parameters to use in this distribution. Invalidity may be caused by inadequate use of the statistical methods applied to find the type of distributions and the parameters.

The most severe problems occur if the data is not representative of what they are believed to be. The data may not be statistical reliable, or may come from a situation or state that is not representative. This is often not revealed, and the model based on these data are not valid, at least not for situations different from that situation where the data was collected.

A lot of typing and editing errors is made when the model is built on the computer. Most of these are corrected immediately, but a small amount, mostly figures, is not found. It is often difficult to track these errors if they are not found by the first simulation run. Once the first run is accepted as more or less valid, one is very unwilling to disapprove it later, even if later runs show that the early model was far from valid.

After the model is running on the computer, the traditional verification and validation jobs start. A number of runs with a varying number of replications are made, and it is often difficult to see the connections in the model. Books can be written about what to do and not to do (and in fact have been), but I will only point out two things that are very much left to the user, and not easily put into a simulator.

The first is the need to be systematic in doing this job. Make a list of check points to look for after each modification is made, and use it every time.

The other is to have a plan for tests to perform, and to mark out which changes that has actually been tried out. Make a track list of the mistakes corrected, and what changes in the results they made. Many hours have been wasted by doing a lot of testing in late afternoon, forgetting all about it back home, and doing them again next morning.

No simulation experiment can be completed without extended use of statistical analysis. There is a fear amongst many developers to overload the system with statistical functions and facilities. This is based on the fact that many of the people in manufacturing feel that to use statistical analysis reflects an academic approach. At this point I will only state that the realism in simulation experiments is totally dependent on the use of such analysis. A simulation study does not predict what will happen in detail, it is the average situations that are predicted with a confidence interval, or by the significance of a hypothesis.

When presenting the results from a simulation study, they have to be defended, and there will always be someone who does not believe that they are valid. One should, of course, have faith in own work, but not be blind. There may be important things that are left out in the models, although most often this will have no effect on the overall results.

4.7 Model validation in SIMMEK

4.7.1 Model validation aspects of the SIMMEK system

As pointed out earlier, program verification is very much put in the hands of the system developers when dealing with simulators like the SIMMEK system. This increases the need for the end users to be able to perform model validation. We have developed a system where the user is guided all the way through the modelling phases, and with excellent means of model validation at all stages of a simulation study. The following list shows the different facilities and aids in the sequence that they are normally used, starting with how validity is secured in modelling;

* Use of statistical packages for testing input
* Check of input values
* Input by pointing and referring

*	Check of model logic
*	Aggregated schemes of input values
*	On-line checks during simulation runs
*	Monitoring

Table 4.3 SIMMEK; How validity is secured in modelling

This is in the core of model validation. As we now go on to how validity is secured and assisted in experimenting and result analysis, please remember that the validation is cyclic, and many facilities are used more than once during an experiment. The facilities are;

*	Advanced trace
*	Statistical tests
*	Graphical visualisation
*	Automatic use of replications

Table 4.4 SIMMEK; How validity is secured in experimenting and analysis

The next section explains how these facilities are used in validation, and why they are important.

4.7.2 Model validation during modelling; data collection, input, and test runs

4.7.2.1 Use of statistical packages for testing input

The first crucial job, the data collection, is done with help of the StatWorks™ package or with Excel. Data from the production may be loaded into files in the StatWorks™ format. Statistical tests to find the type of distributions to use, and parameters of the chosen distributions are performed. The parameters may be copied and pasted directly into the modelling environment. This secures the validity of both the distributions chosen and helps avoiding typing mistakes.

4.7.2.2 Check of input values

All parameters that are typed in the model are checked immediately. On most parameters there are limits to keep within; a simple example is that it is not allowed to use operation times that are negative. In other cases there are tests on the plausibility of parameter values; if the input says that more than 50 % of the parts of a production order is on the average to be rejected by quality measurements, the user is asked by the system to confirm this explicitly. The system points out all illegal values by shading the input fields when trying to save them. A message command is activated in the menu, and by choosing this command the system explains why the value was rejected. It also contains information about the legal range of this parameter.

4.7.2.3 Input by pointing and referring

The product flow, or product routing, is modelled by making a process plan for each product or part that is included in the model. These process plans consist of a number of steps or stages, where each stage may consist of several operations performed on the same resource, i.e., machine, transport unit and so on. These stages in the process plan are generated by making a reference to the resources present in the model. This function makes it unnecessary to write the names of the resources. In this way a lot of spelling mistakes is prevented, one can be more sure that the correct resource is pointed out, and it is more likely that the model is valid.

4.7.2.4 Check of model logic

When the model is more or less complete, it is possible to run a check to see if the system agrees. If one model object refers to another non-existing object, this is revealed by this check. The check also tests that all descriptions of resources and products contain a minimum of what it should do.

If there are process plans with stages with all zero operation times, this is given as a message to the user. But it is allowed to model this situation; imagine modelling the co-opting of a customer order with one, or a part of one, production order as an assembly. This can be modelled by the use of an imaginary resource called "Assembly station", but the time it takes is zero.

4.7.2.5 Aggregated schemes of input values

Obviously this check does not reveal all the mistakes or non realism that may have been put into the model. Before the start of the simulation runs, which may take some time, a quick overview of the model objects and data is available. To go through the entire model the same way as in the modelling phase will be a tedious and time expensive process. Therefore there is available a number of aggregated schemes, showing the main parts of what has been modelled. This will point out how much each resource is loaded, and also which products are routed to the different resources. It is important to point out that this is not a "simplified" simulation; the interaction between resources and products are not considered. All values are just ideally calculated, see section 5.3.3 for details.

This function is as already mentioned important to secure that time is not spent in waste, running invalid models, the invalidity caused by an unrealistic overload of the resources. But it is also important when examining the results. What causes queues, extreme throughput times and waiting times may not be easy to track. A comparison done with the expected values from these schemes may give a clue about what is happening.

A future version of this function will also present the scheduled start-up times of orders, and the estimated throughput if everything goes as planned, and only averages are drawn from the distributions.

Simulation runs where all the drawings from distributions are replaced by constant values are another function yet to be implemented. The constants will, of course, be the expected means of the distributions. Examining the trace from a simulation run with such constant values will be much easier than examining traces with varying values. This again makes it easier to check the model logic, which is very important in model validation, and will be dealt with later. A simulation run like this may be run in today’s version, but only after a considerable amount of remodelling work. In the future this should be a "one command job".

4.7.2.6 On-line checks during simulation runs

Special tests may be chosen to be performed during the simulation runs. This has two purposes, the first being that unwanted situations may be discovered, and the simulation may be stopped. An example is if the queues in front of a resource is extreme, and there is no point in continuing the run. It also points out how early, or hopefully how late this event occurs, since the simulation time is shown on the top of the screen during the simulation.

The other reason is that it is possible to know how many times, or with what frequency, a certain situation occurs. A parameter may be modelled by a distribution with parameters which may give negative values. If there is a message before this negative value is set to be zero, this can give an indication that the distribution parameters given are probably not ideal for modelling this model parameter.

4.7.2.7 Monitoring

The monitoring function is thoroughly described in Section 5, along with the other results that SIMMEK produces. The monitor function can be regarded as five television-cameras that watch a specific part of the model during the simulation. With these cameras it is possible to watch the queue in front of any resource, or the pile of finished parts or products of any type.

It is mainly a function for model validation. If a queue is growing all the time, if the pile of products is growing, or on the other hand if there are no finished products, this is a signal that something is wrong in the model. It may be a logical mistake or the input values may be wrong. Again it saves the user from spending a lot of computer effort and time running a simulation with an invalid model.

The monitor function may also be used during the experimental phase, when the model is proven valid. But in this phase it should be used very carefully, see Section 5.4.

4.7.3 Validation through experimenting and result analysis

What I have mainly been describing so far, is how the SIMMEK system helps in making valid models. In this last part I will present the means available to perform the validation tests, that are doing the jobs numbered 5 and 6 in Figure 4.1.

4.7.3.1 Advanced trace

It is, of course, possible to get a trace of what has happened during the simulation run. A choice has to be made from which replication the trace should be taken. The trace is put into a file which is fully editable. This means to choose how long period is going to be examined. It is also possible to extract information from the trace. For example may one resource, and all information about this resource, be extracted for presentation, and the examination will be much easier.

4.7.3.2 Statistical tests and graphical visualisation

As mentioned earlier the results are produced in file formats that are fully compatible with Excel, Cricket Graph™ and StatWorks™. This makes it possible to present the results in tables and graphs. The labels of the lines and columns are the same as used in manufacturing terminology. It is also possible to perform statistical analysis of many kinds. If StatWorks™ has been used to find distributions and parameters, the input can be directly compared to the results. And it is, of course, also possible to paste real production figures into the result files, and do direct comparisons.

4.7.3.3 Automatic use of replications

The main results from a simulation run are the overall average results, covering all replications. But it is also possible to get detailed results from one of the replications. This is very useful when trying to find out when things started to go wrong, and what seemed to cause it.

A more elaborated description of the result presentation and analysis functions available is given in the next section.

4.8 Summary remarks on model validation

I have throughout this section given many reasons why model validation is so extremely important in every simulation experiment. And it is important as a discipline not only to be performed at a certain stage of a simulation study, but as a discipline to consider all the way from the problem definition phase to the implementation of the decisions made from the experiment are done.

If invalid values or invalid logic can be discovered at an early stage, this can save a lot of time and effort at later stages.

In SIMMEK a lot of facilities is available to ensure the validity of models. Some of these are automatically performed, while others are left to the modeler to use.

But the main conclusion of this whole Section 4 is not concerning SIMMEK, but has to do with user.

It still depends on the modelers awareness of the importance of model validity, whether a simulation experiment is successful or not.