

GETTING TO GOOD ANSWERS: EFFECTIVE METHODS FOR VALIDATING COMPLEX MODELS

Frank Chance
FabTime Inc.
325M Sharon Park Drive #219
Menlo Park, CA 94025
www.FabTime.com

Jennifer Robinson
FabTime Inc.
325M Sharon Park Drive #219
Menlo Park, CA 94025
www.FabTime.com

Nancy Winter
Industrial Design Corp.
2020 SW Fourth Avenue
Portland, Oregon 97201

KEYWORDS: Model Validation, Model Credibility

ABSTRACT

Model validation is like a trip to the dentist. You know it's necessary, but other tasks always seem to take precedence. The actual event can be painful, and there's always an element of uncertainty about problems that may come to light. But after it's over, you've either got a clean bill of health, or a good idea where the true problems lie. And in your heart, you know these problems are best addressed sooner rather than later. Validation for semiconductor and electronics manufacturing models is particularly challenging (some would say painful) due to the amount of input and output data involved. Good answers, however, require valid models. In this paper, we present several case studies in model validation, and describe validation methods we have found to be particularly effective.

INTRODUCTION AND SUMMARY

We use models to obtain answers for our clients. These answers, then, are a highly distilled function of all model elements – data, assumptions, relationships. When presenting answers to clients, we typically do not present this underlying detail. Instead, we summarize the process through which we have established model credibility, or the model's ability to provide good, credible answers to the right questions. Throughout this paper, when we say *model validation*, we refer to this process.

By contrast, we will not be discussing *model verification* – the process of ensuring that the model produces provably correct outputs for given sets of inputs. Not that model verification is any less important than model validation – validating a model that's not been verified is like taking a cross-country trip in an airplane

with navigation instruments of unknown quality. It's simply not wise.

Our modeling experience lies primarily in semiconductor and electronics manufacturing. In this arena, model validation is a daunting and painful task simply due to the huge number of model inputs. For example, it's common to see a factory with ten different process flows, each containing 250 process steps. Conservatively, each process step requires five data elements: the tool required for processing, the operator required for processing, load time, process time, and unload time. At this point, we already have a model with a minimum of $10 \times 250 \times 5 = 12,500$ inputs. And that's before we account for scrap, rework, cost data, tool data, or operator data.

But we have to validate. The sheer quantity of data means that we cannot personally gather each model input. There is always a wide variety of data sources – corporate databases, engineering spreadsheets, individual observations, etc. Stitching these data sources together into one model inevitably introduces subtle problems. These problems, in turn, undermine the model's credibility. In this paper, we present three methods we have found to be particularly effective at flushing out these problems. Each of these methods is illustrated by a case study drawn from our experience in the semiconductor and electronics manufacturing industry.

- Method 1: When creating the project schedule, deliberately plan to execute and present analysis results mid-way through the project. The model will not be fully validated at this point, but the process of performing the analysis and presenting initial results often brings to light otherwise hidden model validation problems.

- Method 2: Throughout model development, carefully document all model assumptions. Regularly review the list of model assumptions for continuing applicability. Undocumented assumptions are a particularly insidious source of validation errors, because unlike input data, assumptions are often invisible.
- Method 3: Use sensitivity analysis to identify one or two classes of input data that have a strong impact on model results, and then manually check this data for errors. Be especially cautious of data that is gathered by automated systems, but which has not been used extensively for other purposes. Data of this nature is apt to contain missing or out-of-range values.

We have found these three methods effective in reducing the pain normally associated with complex model validation. Since model validation is a sizeable milestone in most modeling projects, establishing model credibility is a big first step toward getting good answers.

METHOD 1: DELIBERATELY PLAN TO PERFORM THE ANALYSIS TWICE

Purpose

The client had developed a combined simulation and capacity model of their factory, and wished to use this model to plan tool purchases over a twelve month factory ramp-up. The client hired us to perform a quick-turn ramp analysis, as they did not have sufficient in-house staff to perform the analysis by the deadline.

Project Description

The project consisted of three main tasks: validate the current model against the factory, produce a list of tool purchases required to meet capacity throughout the ramp, and develop a menu of low-cost tool purchases that would improve cycle times.

Model Description

The project starting point was an existing model of the factory. This model had been developed over a period of several years, but had not recently been validated against current factory performance. The model consisted of three process flows, each quite detailed (around four hundred steps per flow). The model included scrap, rework, operators, equipment failures, and equipment preventive maintenance.

Validation Efforts

The first step in the project was to refresh the model with current factory data, and then validate model results against the factory. Through this process, a variety of small problems with the model were uncovered and resolved. These problems included incorrect staffing levels, bad load and unload time assumptions, and incorrect travel times.

Once the model was validated to the client's satisfaction, we added a monthly wafer starts plan (provided by the client) to the model. From this data, we used capacity analysis to develop a month-by-month list of tool purchases required to support the factory ramp-up. We also performed extensive simulation runs to narrow in on a list of low-cost tool purchases that would reduce cycle time.

By this time, however, the project deadline was fast approaching. Anxious to complete the project on time, we summarized the results in a written report, developed presentation slides, and arranged a meeting with our client contact and his immediate supervisor. At this meeting (held the same week our client contact was to present the results to a wider audience), we made a startling discovery: the model was valid for the *current* factory, but not for the *future* factory. We were using an outdated wafer starts plan.

What had happened? The wafer starts plan we were using had been given to us by our client contact. He, in turn, had pulled it from a corporate plan stored on the company's network. Like those in most semiconductor manufacturing facilities, this plan was subject to occasional revisions, as the industry environment fluctuated. Our wafer starts plan was pulled not from the most recent plan, but from an earlier version.

With this discovery, much of our analysis work was invalidated. The new wafer starts plan differed significantly from the old, leading us to the conclusion that we must redo both the tool planning and cycle time reduction analyses. So that is what we did. We were still able to meet the original deadline, but only with several very late nights, and much more stress than would have otherwise been required. And this time, we double-checked the starts plan for accuracy before doing the analysis.

Lessons Learned

The underlying mistake we made was to assume that model validation was complete when we obtained a model that matched the current factory. We didn't take the obvious step (obvious in hindsight, anyway) of validating the wafer starts plan relative to the question that we were trying to answer. We believe that the best method to avoid this type of mistake is not, however, to redouble our data validation efforts. Instead, we now routinely schedule our projects to ensure that we consciously do what we were forced to do in this case – perform the model analysis twice.

We usually perform a first-pass analysis after the majority of model development has been completed, but before the bulk of model validation is started. We present first-pass analysis results to our client contact, and if possible, to at least one level of management above our contact. The intent is to catch errors that might otherwise slip through model validation, using the ultimate test – can the model answer the question posed by the client? A side benefit to this method is that early in the project, we get excellent feedback on whether the questions we are answering are in reality those posed by the client. And we get this feedback while there's still time for corrective action.

It is important to manage expectations prior to the presentation of first-pass results. The client must understand that the model is not in its final form, and that results may change significantly as the model is refined. It helps to start this presentation with a list of model assumptions, and a detailed progress report on the status of model validation efforts. These steps help the client to understand what can realistically be expected of the model, and also help to temper the disappointment that occurs when the inevitable problems are uncovered. Given that these problems will be discovered sooner or later, we'd prefer sooner.

METHOD 2: DOCUMENT AND REGULARLY REVIEW MODEL ASSUMPTIONS

Purpose

The client had a detailed simulation model that represented their factory. They had been using the model for planning equipment purchases, six to twelve months in advance, but now wanted to use it to understand and improve cycle times. The trouble was that the model underestimated cycle times by approximately 50%. The client hired us to

help determine which factors were causing cycle time deltas between the model and the actual factory.

Project Description

The project consisted of three main tasks: validating the data currently in the model, adding detail to the model to more accurately reflect cycle times, and running what-if experiments on the model to determine which factors had the most significant effect on cycle time. The first two phases together absorbed approximately 80% of the project time and effort, leaving relatively little time for experimentation. In our experience, this type of ratio is typical when working with large, complex models (see the next case study for another example).

Model Description

The model consisted of four process flows, each of which was used to model both regular and high-priority lots. The process flows were very detailed, each consisting of over 1000 steps (including transfer steps, rework steps, and automated material handling steps). The model included scrap, batching rules, equipment failures, equipment preventive maintenance schedules, and operators. The model did not include setups or equipment dedication.

Validation Efforts

Our first step in validating this model was to ask the client what assumptions had been made in developing it. The client engineers had carefully documented revisions whenever they changed data in the model, and these revision logs were a useful source of assumptions. Other assumptions, however, had not been documented at all (in most cases because they were considered so basic), and only came to light through discussion and review of the model outputs.

It quickly became apparent that several major assumptions about the model were contributing to the low cycle times. These assumptions stemmed directly from the fact that the model had been developed as a planning tool for projecting equipment purchases. The engineers were using a feature of the software that allowed the model to calculate the number of tools in each tool group such that the overall loading of the tool group was 85%. These planned tool quantities were then being used in the model. Similarly, they were using this planning feature to calculate the number of operators in each group, using an even lower capacity loading. For operators in particular, these calculated numbers were significantly larger than the

actual number of resources in the factory. This was primarily because the factory was ramping production, and for the period being validated, the equipment and operators were actually loaded at more than 85% of capacity. When we replaced the tool and operator quantities with actual values from the shop floor, cycle times in the model increased so much as to become unstable. This in turn led us to re-examine other assumptions about how much operator time was required at the various steps, and to uncover other problems.

Similarly, the start rates being used in the model were based on the average planned start rate for the factory. The assumption was that the same fixed number of starts was released each day, with releases spread uniformly throughout the day. This is a common assumption in planning models, where more detailed data is not available. In this case, however, the lot releases in the actual factory were very different. First of all, the total number of starts per day fluctuated significantly from day to day. Second, the starts for a given day were released according to a particular pattern. Half of the lots were released at the beginning of each 12-hour shift. Making these two changes to the model increased the lot release variability dramatically, and thus increased model cycle times.

We also discovered that the process flows in the model represented the four primary flows that were expected to be in place several months in the future. In several cases, these flows were very different from the flows currently in the factory. This situation is common in a model that is used for planning, particularly when a factory is relatively new. In order to accurately validate the model against current cycle times, we had to make several process flow changes. Note that we do not advocate trying to change every detail in a model to reflect a constantly changing shop floor. However, in this case some of the tools that contributed the most to actual cycle time were affected. We turned to the client's assumption list for the model, and reversed some of the most futuristic of the assumptions, so that the model better represented the current situation.

Other assumptions in the model also failed to stand up to scrutiny. The batching rule used at most steps said that all lots at the same operation could be batched together. The engineers actually meant to say that all lots of the same product at the same operation could be batched together. They did not intend cross-product batching, but did not fully understand the batching rules in the software. In this case, the assumption stated by the client did not

match the data in the model, and we were able to fix the problem.

An assumption that only came to light when we asked about it concerned equipment dedication. Each equipment group in the model was assumed to be identical, so that each step could take place on any one of a number of identical tools. When we asked about this, we found that the tools in a number of equipment groups were not identical in the steps that they were qualified to process. There were a number of steps that could only be performed on a single, specific piece of equipment. Because the plan overall was to eliminate such equipment dedication, it had not been included in the model. In the meantime, however, the equipment dedication was driving up cycle times, because when those tools went down for repair, there was no alternative for processing certain steps. Because this assumption concerned detail that was not modeled, it would have been particularly easy to miss during validation. We only asked about it because in other factories we have observed equipment dedication to significantly drive up cycle times.

Lessons Learned

The main problem with this model was that we were trying to validate a model based on somewhat futuristic planning data against actual factory cycle times. The only way to successfully validate the model was to revisit the assumptions, and modify those that did not apply to the current factory situation. This was complicated in this case by the fact that the model had been used for more than a year, and did not have a comprehensive list of assumptions. We ended up spending a lot of time decoding the model results to figure out what the initial assumptions had been.

Based on what we learned in this project, and what we have seen in other projects, we have two recommendations. One, you should always maintain a current, detailed list of assumptions about your models. Two, when you are using a model to answer a new question, you should revisit the list of assumptions to make sure that each one still applies.

Incidentally, we have observed that a primary reason why people don't keep track of assumptions for their models is that they think that the model will never be used to answer other questions. They believe that they are building a single-use model that will only be used by the original developer, and only in the short term. Therefore, they don't need an assumption list. In our experience,

however, single-use models are relatively rare, and models believed dead often resurface. We have been grateful many times for assumptions that we documented, that we never would have otherwise remembered months later.

METHOD 3: MANUALLY VERIFY HIGH-IMPACT INPUT DATA

Purpose

The client was looking into combining several assembly lines. They contracted with us to develop a simulation model to determine the optimal configuration of the new line(s) with the goal of increasing throughput and decreasing throughput variability. The client wanted to look at the effect of building machine redundancy into the new line.

Project Description

The project was broken into two phases. The first phase consisted of developing a baseline model of the current line and validating the model against the floor. Validation was considered a critical step in the project's success. The second phase consisted of developing several alternative line configurations and modeling them to determine the best one. We worked with the client to develop alternative configurations based upon physical limitations of processing as well as results of the initial simulation analysis. Approximately 75% of the project time was spent on the first phase.

Model description

The baseline model simulated the existing assembly line, which consisted of approximately 30 machines linked by a series of conveyors. The conveyors served as both a conveyance between machines and an additional buffer. There was only one machine of each type on the serial line. Rework, scrap, and blockage were all considered in the model. The alternative configuration consisted of one serial line broken into 3-4 sections of parallel lines that fed into the next section. The alternative model allowed for looking at varying number of parallel lines.

Validation Efforts

During initial meetings with the client, we established that the two measures we would validate against were throughput and throughput variability. The productivity of the line was measured in the number of units produced per hour. The variability in throughput from hour to hour

was unacceptable to the manufacturing supervision since it made meeting the targeted throughput very difficult. The model would be considered valid if these model outputs came within 10% of those on the floor.

The first step in the project was to collect input data and validate it with the manufacturing and industrial engineering staff. Processing times were collected from engineering databases and verified through time studies. Scrap and rework data were collected from engineering data. Unscheduled downtime data were collected from on-line records. Every hour the amount of time each machine was down was captured along with the number of down events that occurred.

The next step in the project was to compile the data and present it to the engineering staff for validation. The downtime data set was of particular interest for two reasons. First, it was considered to be the only significant source of variability in the line since processing times were fairly constant. Second, it was the only set that was collected not from direct observation but from electronically recorded data. The data had never been used the way we planned to use it, but had only been used to get high-level estimates of down time percentages.

First pass validation of the input data showed some problems with the downtime data for several machines. Some machines were showing too high a downtime percentage while others appeared too low. Looking at the recorded data with the client engineers, we noticed periods where the data looked suspicious. In fact, there were many periods where the time spent down was either exactly .02 hours or .998 hours. After checking into this phenomenon, it was found that these recordings reflected times when the computer system was either down or not functioning properly. Therefore, we threw out all periods with either of these values. Since we had collected data for a month, we were still confident in the number of observations we had. After this scrubbing of the data, we found accurate downtime percentages for all machines.

First pass validation on the completed model showed very good results for average throughput rates. Throughput variability, however, was too low. As this was deemed to be critical, we went back to the input data and looked for sources of variability. It was by manually examining this data that we found that down events that were longer than one hour (the recording period) would not be captured by the summarizing techniques we had used. This meant that we had to manually go through all observations for all machines to find these long down

events. This turned out to be a very time consuming procedure but also the key to a successful validation.

At this point we had very good mean times between failures and mean times to repair, but did not have the distributions around these parameters. The nature of the collected data did not allow us to directly get these distributions. Therefore, we performed a sensitivity analysis to determine satisfactory distributions. The client's experience with the line was that most machines had a lot of very short down events with a few very long ones. Similarly, machines would go for a long time without failing and then go down several times in a short period. We used the exponential distribution as a starting point to capture this variability. We also tested several other distributions including normal and uniform with varying spreads.

Results were best for the exponential distribution. However, we found that the exponential distribution alone led to occasional extremely long down events that were unacceptable. Examining the data even further showed that separating down events in quartiles by down time would provide more realistic down distributions. The final sensitivity analysis showed that the best distributions were combinations of exponential distributions for the shorter down events, and triangular distributions for the longer times. Using the triangular distribution to model the longer down times allowed us to have long tails on them, yet still bound them.

Lessons Learned

A modeler should not rely on input data that has been compiled automatically. We initially did this with the downtime data, but after going through by hand, we found an unacceptable number of anomalies. Manually checking the data was a lengthy process but in the end we came out with a model that very closely tracked the floor and in which the client had considerable faith.

We were lucky that we were able to point to downtime as the source of critical data early in the process and therefore able to focus our efforts on it. We recommend that some time be spent early in the project to determine potential problems like this. We further recommend that a sample of data that is obtained from electronic collection systems like the downtime data in this project be manually verified. Had we taken just one or two machines with high downtime percentages and critically examined the data up front, we would have saved some time looking for problems down the line. Unfortunately, problems often

come up at a time when we are trying to complete the project or the current phase of the project. Like the first case in this paper, we learned our lesson about building time into our schedule for first pass validation.

AUTHOR BIOGRAPHIES

FRANK CHANCE is President and co-founder of FabTime. He has been developing software since 1983, and performing factory productivity improvement consulting for the semiconductor industry since 1993. Previously, he founded Chance Industrial Solutions, and developed and commercialized Factory Explorer®, an integrated factory-wide capacity, cost, and simulation analysis tool. From 1993 to 1995 Dr. Chance was a consultant to the SEMATECH, where he developed Delphi, the primary fab capacity analysis and simulation tool used by the Measurement and Improvement of Manufacturing Capacity project. Dr. Chance has dual AB degrees in Mathematics and History from Washington University. He also has M.S. and Ph.D. degrees in Operations Research and Industrial Engineering from Cornell University.

JENNIFER ROBINSON is co-founder and chief operating officer of FabTime Inc., a provider of wafer fab cycle time management software. She is also the editor of the FabTime Cycle Time Management newsletter, a monthly email publication dedicated to wafer fab cycle time improvement. Previously, she worked as a productivity improvement consultant in semiconductor manufacturing. Her clients in that industry have included AMD, Headway Technologies, SEMATECH, Digital Equipment Corporation, Seagate Technology, and Siemens AG. Jennifer holds a B.S. degree in civil engineering from Duke University, an M.S. degree in Operations Research from the University of Texas at Austin, and a Ph.D. degree in Industrial Engineering from the University of Massachusetts at Amherst.

NANCY WINTER is specialist in simulation and manufacturing analysis for the Industrial Design Corporation (IDC). Her areas of expertise include simulation modeling, static modeling and statistical analysis of manufacturing systems. Ms. Winter holds a B.S. in Industrial Engineering from Rensselaer Polytechnic Institute, and an M.S. in Industrial Engineering from Purdue University. Prior to joining IDC, she worked at Intel as an industrial engineer.