

PROFESSIONAL SPREADSHEET-BASED OPERATIONS RESEARCH MODEL DEVELOPMENT

Hervé Thiriez
HEC School of Management
78350 Jouy FRANCE
e-mail: thiriez@hec.fr

ABSTRACT

The spreadsheet development of professional models presents many advantages: the natural interactivity of spreadsheet models, the unbeatable performance to cost ratio it offers in comparison with standard development solutions, the absence of need for any specific license, the universality of environments in which the model may run, etc. There are some drawbacks, e.g. the size and speed limits of spreadsheet models, and the difficulty to audit or validate such models. This presentation will show how, using several specific techniques, a number of professional models were developed, often with Crystal Ball®, and how, in some cases, these models were validated. A strongly positive side effect is that, since these models were developed very fast, not only were they far cheaper than competitive models, but they were also definitely more realistic, since less time has elapsed between the initial analysis and the delivery of the final model.

KEYWORDS

Spreadsheet-based models, Decision support systems, Model design, Interactive simulation.

1 INTRODUCTION

There are a number of advantages to using spreadsheets in the development of professional models : spreadsheets are present on all computers, they are very interactive, they allow for fast and cheap developments, and there is little human resistance to the use of spreadsheet-based models.

The limits of using spreadsheets for professional models are size limits, i.e., currently 256 columns and 65,536 rows (the number of rows is less of a problem), calculation speed, calculation errors or problems, difficulty of model auditing... We will comment on these limits of spreadsheet-based model development and show how some of the problems related to these limits may be solved in practice. At the end of this paper, we will also see how some techniques may be used in order to better use Crystal Ball with professional models, and how to enhance Microsoft® Excel graphs for such models.

2 ROW AND COLUMN LIMITS

The 256-column limit is probably the major problem I ever had to solve when using spreadsheets for the development of professional models.

Let us assume you create a model for the time simulation of a waiting line, and the time unit is 30 seconds. With 256 columns, you can only simulate a 2-hour process. At some time, I was developing a planning model for a rolling mill. The base task of the mill was 6 minutes long, which meant that a model dedicating a column to each 6-minute slot could not represent more than one day of activity. The problem was that the planners also needed to have a 5-day view of the situation, where they did not necessarily want to see all the details.

We see below the model for the mill (Laminage, in French, in row 2) where each of the following rows represents one of the heating ovens. Other ovens are hidden by TT and RW.

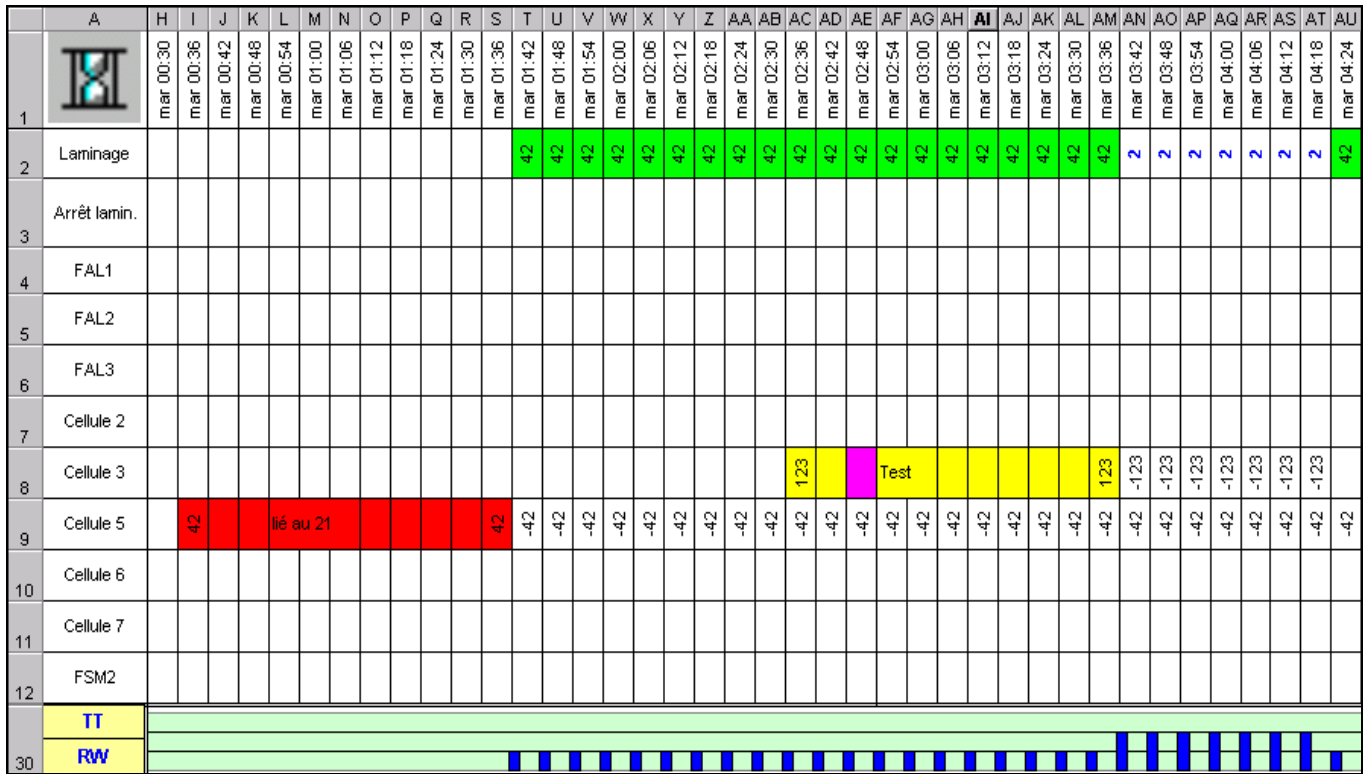


Figure 1: Planning model for a rolling mill

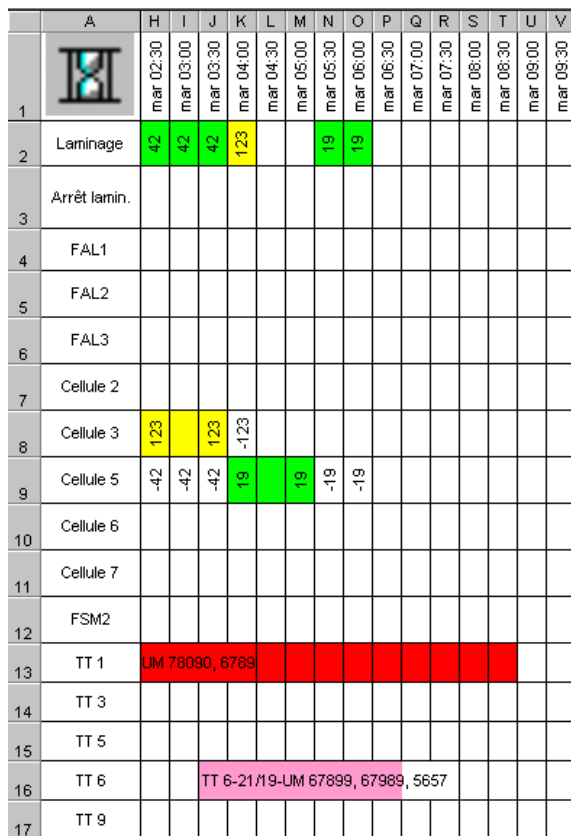


Figure 2: View with the TT ovens open

The solution found for this model was to have a switch, activated by a click on the sand-clock in cell A1, which transforms the 6' time interval into a 30' time interval. A macro then redraws the schedule accordingly.

Of course, a detailed analysis of the planning was meaningless in this representation, as several events could happen in the same 30' slot and, quite evidently, only one of them would have been visible in the cell. However, with this feature, it became possible to have a 5-day view of the jobs.

On the left, we can see such a view with the TT ovens open (the opening or the closing of the TT ovens is achieved by a click in the TT button in A30). Going back and forth between the 6' and the 30' views, it was easy to have a good understanding of the planned period.

The limit with the number of rows is not so stringent and, when a large database has to be used, it is often more efficient to handle the complete database with a tool such as Access and to extract databases that Excel still manages properly.

With adequate VBA developments, it is however possible to overcome this limit by having the code handle bases split between several sheets of the same workbook. And, since VBA code runs much faster than Excel database functions, specific macros often make it possible to do efficient analyses on such multi-sheet bases.

3 CALCULATION SPEED

Complex spreadsheet models often show calculations times which are too long, and sometimes not even acceptable. There are several techniques for solving this speed problem.

The first technique is to clean the model, i.e., to get rid of any redundant or obsolete formula:

- A formula is *redundant* if the same result is calculated in several cells : in such a case, it is often better to put in a cell the intermediate result which will be used by several other cells.
- A formula is *obsolete* when the data used by the formula are data which will not be modified in the future.

This often happens with sheets containing historical data, e.g. one column per month over three years. When you are quite sure some “old” columns will not be modified any more, you can select them, copy them, and “paste special” their values. One should always remember that Excel calculates from top to bottom and, in each row, from left to right. Another technique for speeding up calculations is, knowing this, to organize your sheets in such a way that all your formulas are evaluated according to this calculation order.

The last and most efficient way to speed up calculations is to have macros put the formulas in the cells, calculate, replace by the values, go to the next group of cells and do the same operation again,... Of course, you gain tremendously in speed, but at a cost: your models becomes difficult, at best, or impossible, at worst, to modify or audit properly. I did this on a model I had designed for the simulation of a toll booth system. When the model was initially developed with all its formulas, it took 45 minutes to simulate two hours of traffic. After I had developed macros to put the formulas and calculate the results, the simulation time has decreased to less than 2 minutes. There were even larger savings, in proportion, when I had – in another model – macros which replaced Excel’s database functions for very large databases.

4 CALCULATION PROBLEMS

Calculations problems are something else. There are two major such problems : Excel refuses to calculate even though you use the [F9] key, and Excel really gets wrong results. When Excel becomes unable to calculate with the [F9] key, you often solve the problem by using the [Ctrl]-[Alt]-[F9] combination. If this still does not work, you may resort to an astute “Replace All” of “=” by “=”!

Excel getting wrong results is altogether another problem. This is due to the fact that Excel only has a precision of around 14 digits, but this is the initial precision. The initial precision may even be much less than that, sometimes less than 6 digits, e.g. in the case of several Excel statistical functions until the Excel 2003 version appeared, with its revamped statistical functions. The more operations you do, the smaller the precision. This means that, when some data are subject to many operations, it is wise – whenever possible – to use at some point a technique which will restore the original precision. Numerous such techniques have been developed, which may be found in the numerical analysis literature.

5 MODEL AUDITING AND MODEL VALIDATION

What good is a model if you cannot understand how it works or if you cannot validate its results ?

By nature, spreadsheet models are not easy to audit. Model auditability apparently has never been a Microsoft priority, which can easily be verified by this small experiment: when you right-click on an Excel toolbar, you get a list of available toolbars, but the “Audit” toolbar is not to be seen in that list. In order to see it, you have to use the “Customize” command and look for the bar !

Models can be made more auditable quite simply by using names : a formula such as “= Product cost + Margin” is far better than “=B6+K8”. A model is also easier to audit if all the blocks are well isolated : data blocks, intermediate calculations, results, ...

Last, there are also auditing tools, some of them Excel add-ins, which help you better understand a model and automatically notice apparent inconsistencies. Model validation is a definitely more difficult issue, too often overlooked in professional developments. This problem was solved in an interesting fashion for a car-traffic simulation model I created for Aéroports de Paris (AdP), in order to simulate the car traffic in the 20-square km airport area.

During our initial analysis of the complete layout, we selected all the points on the map where we wanted to have car counters. These counters measured car speeds and numbers, on a minute per minute basis. The counters were installed in the many selected positions and took minute per minute measures during all sorts of full days : standard day, weekend, vacation time, ...

In this fashion, many Excel models (one per day, with a row per minute and a column per point) were created, through direct links with the measure apparatus. On a common agreement with AdP, we were provided with half of these data files, AdP keeping the remainder for the validation phase.

I strongly believe in bottom-up development, which I always use in practice. Therefore, I began by developing initially a relatively straightforward model. I then used the first few Excel data files in order to see how the model behaved : this allowed me to enrich the model so that I could obtain more realistic results. When this was done, I opened additional data files and checked my model with these new data, which again led me to add new features to my model and enrich its formulas. Adding data files progressively, and checking the model at each step, I enriched it until it behaved in a satisfactory fashion with all the data files in my possession.

I then gave my model to AdP, and they checked it with the data files they had kept. My model was validated by AdP because the results it predicted were coherent with what their data files had recorded. When they agreed that the model behavior was satisfactory, they ordered the model for a second of the five traffic blocks. I am currently finishing the development of the model for the third block, but the basic model formulas did not need to be modified since the end of the development of the first block.

6 USING CRYSTAL BALL IN COMPLEX MODELS

When Crystal Ball is used in complex models, one problem you often encounter is the limited number of assumptions and forecasts a model may use: it is normally advised not to use more than 1,000 assumptions and forecasts in any sheet. I once built a test model with more than 3,000 assumptions on one sheet, for the sole purpose of testing this limit: it still worked, but I would generally not advise doing this.

When many assumptions are needed, most of them do not really need to be defined as Crystal Ball assumptions, meaning that you do not need them to be included in analyses such as the sensitivity analysis. In this case, you might as well define them directly (if this is possible) using Excel functions. For example, a normal distribution assumption for the size of a man, with an average of 180 cms. and a standard deviation of 10, may be directly defined in Excel with: =NORMINV(RAND(),180,10).

One feature which I often used in complex models was, using the developer kit information, the capacity to use VBA commands to set up and run Crystal Ball models. This allowed me to easily overcome some of the Crystal Ball limits.

One Crystal Ball feature that turns out to be most annoying is the fact that the macros identified in the "Macros" sheet of the "Run Preferences" command are memorized with Crystal Ball and not with the active Excel model: if you close your model and open another Crystal Ball model, that new model will try to run macros which it will of course be totally unable to find! When I have a model where I want to run VBA macros during a simulation run, I have VBA macros which, when the file is opened and closed, set up and reset the run preferences. Linked with the natural ability of Crystal Ball to run VBA macros, the ability to run Crystal Ball operations from VBA gives the user the ability to develop very professional models.

At one time, I was developing for France Telecom a marketing model in which different price strategies were analyzed, the model simulating the customer response in number of telephone calls, average length of calls,... But France Telecom plays a double role: not only are they one of the competitors in the French telephone industry, but they are also the providers of the base network which is also used by their competitors. And, as such, they have the ability to reorganize the network structure, in real time, in order to minimize the cost of answering the demand.

They explained this to me and told me that, after each iteration which created a random demand distribution, in order to calculate their total profit margin, they needed to evaluate the network structure optimized for this random demand. I asked them how this optimization was made, and they told me it was a simple linear programming model. All I needed to do then was to set up the run preferences so that, at the end of each iteration, a macro called the Excel solver.

7 EXCEL GRAPHS ENRICHED

Professional models require professional-looking graphs. There are three ways in which Excel graphs may be enhanced and made more professional :

- A graph should be *elastic*, i.e. automatically adapt to the number of available data. Let us assume you create a graph with 15 data and, later, delete the last five data. Excel will continue to "show" 15 values, the last 5 of which will be 0! Inversely, if you add 5 data for a total of 20 data, the Excel graph will continue to show only the 15 initial data. An elastic graph will show just as many data as you have.
- A graph should be able to have *dynamic titles*, i.e. titles which include information relating to the data, e.g. the average or the maximal value of a series,... Which means you have to use a formula such as "="The maximal value is

"&K23&" units." But Excel does not allow you to enter such a formula as a title text! The trick is to enter the formula in a cell and to define the title as being equal to that cell...

- Graphs with *pull-down menus* for the series selection are very user-friendly and have a professional look. Moreover, they allow you to reduce both memory size and calculation speed since a single such graph will replace many different graphs you would have to build otherwise.

Professional models often use graphs with these three features : elastic graphs with dynamic titles and pull-down menus for series selections. In several of my models, I went one step further by transforming my graphs into movies. What is a movie but a sequence of fixed images represented at regular time intervals? All you have to do if you want to create a movie in Excel is to have a counter which identifies which column (or row) of a table is selected, and to have a macro which will update this index at regular time intervals. A macro movie is a simple VBA loop which updates the index!

8 CONCLUSION

In this paper, we showed that some problems have to be solved if one is to develop professional models with a spreadsheet. Luckily, most of these problems have at least partial solutions, most of them not demanding to resort to Visual Basic. One of the major advantages for using a spreadsheet in the development of professional models is that the development time and cost are much smaller than with the competitive approaches.

Whenever I find myself in a tender offer competition, my spreadsheet-based solution is 3 to 5 times cheaper and faster than traditional solutions with big modeling packages, or based on development languages. When a model is built faster, not only is it cheaper, but it is also much more realistic. The shorter the development time, the less discrepancy there is between the problem which was initially analyzed and the problem at the time the model has become operational: the model is therefore more realistic and thus more efficient. It is a complete win-win situation: faster, cheaper and better.

REFERENCES

- Baker J.E., Sugden S.J. "Spreadsheet In Education – The First 25 Years", *Spreadsheets in Education*, eJSiE 1(1): 18:43.
- Croll G.J., "A Typical Model Audit Approach", *IICIS 2002*, 213-219.
- Grossman T.A., "Spreadsheet Engineering: A Research Framework", *Proceedings of the Third Symposium of the European Spreadsheet Risks Interest Group*, Cardiff, 2002.
- Nixon D., O'Hara M., "Spreadsheet Auditing Software", *Proceedings of the Second Symposium of the European Spreadsheet Risks Interest Group*, Amsterdam, 2001.
- Panko R., "Spreadsheet Errors: What We Know. What We Think We Can Do", *Proceedings of the First Symposium of the European Spreadsheet Risks Interest Group*, Greenwich, UK, 2000.
- Thiriez H., "Improved OR Education Through The Use of Spreadsheet Models", *European Journal of Operations Research*, 135: 461-476.
- Thiriez H., "Spreadsheet-Based Professional Modelling", *INFORMS Transactions on Education*, Volume 4, Number 2, January 2004, on <http://ite.pubs.informs.org/Vol4No2/Thiriez/>.

BIOGRAPHY

Hervé Thiriez has an engineer's diploma in Applied Mathematics from IMAG (Grenoble, 1966) and a Ph.D in Operations Research from MIT (1969), where he taught for one year. He then returned to France for his military service and became a Professor at the HEC School of Management. He has been an Operations Research consultant for 35 years and is the CEO of Logma, a consulting company. He also published 25 books and more than 300 papers on modelling and micro-computing.