

# Sumwise: A Smarter Spreadsheet

Darren Miller, Gary Miller, Luis M. Parrondo  
Sydney, Australia  
darren@sumwise.com

## ABSTRACT

*Sumwise is a new type of spreadsheet which aims to improve model quality and lower the risks associated with errors. Sumwise adopts a more structured and logical approach, compared to traditional spreadsheets. The core features of Sumwise discussed in this paper include: user-defined row and column names, row and column structure, meaningful metadata or tags, and logical cell groups. These features make Sumwise models easier to construct, review, and extend.*

## 1 INTRODUCTION

There is a large amount of research showing that most spreadsheets contain errors. Amongst the many risk factors and reasons given for the existence of spreadsheet errors are:

- Time-consuming and repetitive copy and paste actions [Hellman, 2005];
- Hard coded numbers in place of consistent formulae [Aldridge, 2003];
- Inconsistently copied formulae [Berglas and Hoare, 1998];
- Difficulty in validating A1 syntax [Berglas and Hoare, 1998];
- The lack of a complete listing of formulae [Bewig, 2005]; and
- Frequency and extent of changes and modifications to the spreadsheet [PWC, 2004].

Sumwise is a new type of spreadsheet that directly addresses these issues, by adopting a more structured, logical, and principled approach, compared to traditional spreadsheets. Sumwise aims to improve spreadsheet modelling quality, and has the potential to lower the risks associated with errors, in three key phases of a model's lifecycle:

- Design and construction;
- Review and audit; and
- Modification and extension.

## 2 SUMWISE

Following is an outline of Sumwise's key features and functionality.

### 2.1 User-defined Row and Column Names

Sumwise employs user-defined row and column names, rather than the usual A1 (or R1C1) references found in traditional spreadsheets.

Others have observed that it can be difficult to validate A1 references [Berglas and Hoare, 1998]. For example, the formula to calculate *Operating Profit* in cell AX53 might be =AX51-AX126; but this syntax has no particular meaning on its own. The coded cell references of traditional spreadsheets are a source of errors, as they take time to interpret, and represent a disconnect between the syntax and the underlying meaning.

Some spreadsheet researchers and experts advocate the use of range names to assist in making models more understandable, however, this approach is contentious and may actually make things worse, as it requires the user to maintain integrity between the alias given to the range and the underlying syntactic reference. What was previously a two-step process:

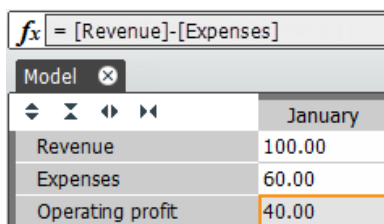
- A1 Syntax → Underlying Meaning;

becomes a three-step process:

- Range Name → A1 Syntax → Underlying Meaning.

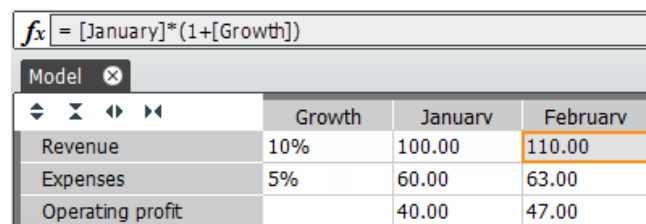
Panko and Ordway reflect this concern, and hold the view that use of range names should be considered “potentially dangerous until research on using range names is done” [Panko and Ordway, 2005]. Their concern seems justified by recent research which indicates that the use of named ranges may “lead to a reduction in debugging performance” [McKeever, McDaid and Bishop, 2009].

Figures 1 and 2 below show a simple Sumwise model. In Figure 1, the selected cell in the *Operating Profit* row contains the formula =*[Revenue]*-*[Expenses]*, and in Figure 2, the selected cell in the *Revenue* row contains the formula =*[January]*\**(1+[Growth])*. In each case, the syntax and underlying meaning are one and the same, and the formula is easy to understand and verify.



Model	
	January
Revenue	100.00
Expenses	60.00
Operating profit	40.00

Figure 1



Model			
	Growth	January	February
Revenue	10%	100.00	110.00
Expenses	5%	60.00	63.00
Operating profit		40.00	47.00

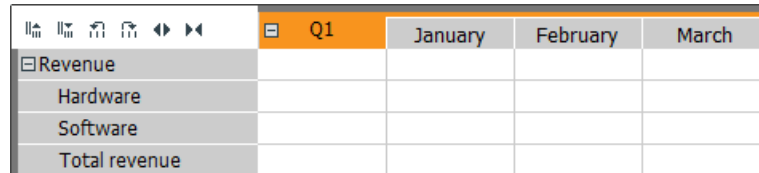
Figure 2

These user-defined row and column labels, and the associated method of referencing cells, make formulae easier to construct and review. And, unlike range names in traditional spreadsheets, these labels, and the formulae that use them, adapt appropriately when modifications are made to the model. For example, changing *Revenue* to be *Sales*, will automatically modify all formulae and other aspects of the model that depended on the label *Revenue*. Furthermore, in contrast to range names in traditional spreadsheets, no guesswork or checking is required to determine whether the labels *Revenue*, *Expenses*, *January*, and *Growth* are in fact pointing to the intended range of cells.

## 2.2 Structure

Sumwise enables rows and columns to be logically structured into hierarchies.

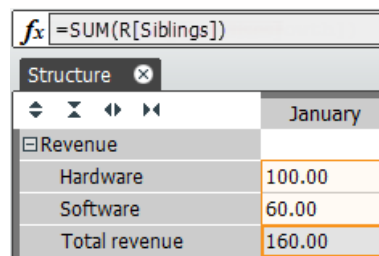
For example, in *Figure 3* below, the *Hardware*, *Software*, and *Total revenue* rows are sub-nodes under *Revenue*; and the *January*, *February*, and *March* columns are sub-nodes under *Q1*.



	Q1	January	February	March
Revenue				
Hardware				
Software				
Total revenue				

*Figure 3*

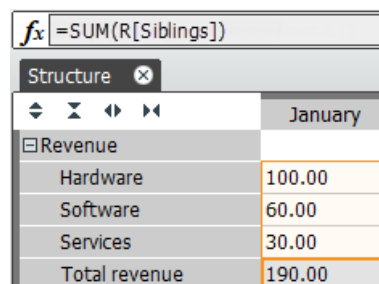
This structure is not just visual, but also reflects the underlying logic of the model, and can be referred to using the keywords: *Parent*, *Children*, *Siblings*, *All*, and *This*. The example in *Figure 4* below shows the use of the *Siblings* keyword to aggregate a list of items. The keyword *Siblings* effectively means all rows in the hierarchy of the context cell, but excluding the context cell itself.



	January
Revenue	
Hardware	100.00
Software	60.00
Total revenue	160.00

*Figure 4*

The benefit of this approach is that formulae can be written in more logical and generic ways. Furthermore, when new rows and columns are inserted, these will be automatically included in Sumwise's structured references based on their structural positions. For example, if a new row called *Services* is added below *Software*, it will be automatically included in the *SUM* calculation because of its status as a sibling row. This is shown in *Figure 5* below.



	January
Revenue	
Hardware	100.00
Software	60.00
Services	30.00
Total revenue	190.00

*Figure 5*

This compares favourably to traditional spreadsheets which generally<sup>1</sup> do not cope well with model extensions, such as inserting new rows and columns, and having references update correctly to include the new cells. Some help is provided by Excel, which deals with certain model extensions (it can cope with the insertion of the *Services* row as per the example above); however, this behaviour is inconsistent and somewhat unpredictable. For example, Excel does not cope with extensions at the top of a range (e.g., inserting a row above *Hardware*).

A further benefit is that the formula  $=SUM(R[Siblings])$  is fairly generic and may be applicable to many cells throughout a typical financial model. It could be used everywhere where the user wants to aggregate the rows at the same level in the hierarchy as the context cell.

## 2.3 Tags

Sumwise provides the ability to add user-defined metadata (or tags) to rows and columns.

For example, in *Figure 6* below, the columns named *January*, *February*, *March*, and *April* have been tagged as *Months*, and the *Q1* and *Q2* columns have been tagged as *Quarters*. These tags can then be conveniently referred to in formulae. If each month is to be based on the prior month multiplied by some growth percentage, this can be expressed as  $=[@Month-1]*(1+[Growth])$ . The same formula can be accurately applied to *April* even though the *Q2* column intervenes between *March* and *April*. This is something that is deceptively complex and risky in traditional spreadsheets.

	Quarter	Month	Month	Month	Quarter	Month	
	Growth	Q1	January	February	March	Q2	April
Revenue	10%	331.00	100.00	110.00	121.00	440.56	133.10

*Figure 6*

Another example of a simple Sumwise model that uses tags in a formula is presented in *Figure 7* below. Each row under the *Items* row has been tagged as being of a particular food group, and in the *Summary* section, the formula  $=SUM([Items.Children@Fruit])$  aggregates the quantities for fruit items only.

	Quantity
Items	
Fruit	Apple 1.00
Fruit	Banana 2.00
Vegetables	Cabbage 3.00
Fruit	Date 4.00
Vegetables	Eggplant 5.00
Fish	Flounder 6.00
	Total items 21.00
Summary	
	Fruit 7.00

*Figure 7*

<sup>1</sup> Although techniques such as dynamic ranges, or functions like *INDIRECT* and *OFFSET*, can aid model extension in Excel, these methods have their drawbacks, including: they are advanced techniques and not widely known, they are volatile and do not provide an audit trail, and they rely on setting up range names which have their own set of risks as discussed in 2.1 above.

## 2.4 Cell Groups

Sumwise allows multiple cells to be logically associated with one another into cell groups. Cells in a cell group can share certain properties such as formatting, contents (string, value or formula), validation settings, and the like. Cell groups can be flexibly defined with reference to row and column structure and tags.

For example, in *Figure 8* below, all cells at the intersection of the untagged rows and columns tagged *Forecast* have been grouped together. A cell (the intersection of *[Hardware]* and *[April]*) in this cell group has been selected, and all other cells in this group are highlighted with the same darker grey shading. The formula for this cell group has been defined as  $=[@Month-1]*(1+[Growth])$ , and all cells in this group inherit this formula (and other properties such as formatting).

		Quarter	Month, Actual	Month, Forecast	Month, Forecast	Quarter	Month, Forecast	
		Growth	Q1	January	February	March	Q2	April
Heading	Revenue							
	Hardware	10%	331.00	100.00	110.00	121.00	440.56	133.10
	Software	25%	190.63	50.00	62.50	78.13	372.31	97.66
	Services	50%	95.00	20.00	30.00	45.00	320.63	67.50
Subtotal	Total		616.63	170.00	202.50	244.13	1,133.50	298.26

*Figure 8*

One benefit of this approach is that the number of unique formulae is reduced, relative to the equivalent traditional spreadsheet model. (For example, the above Sumwise model has three unique formulae, whereas the equivalent traditional spreadsheet model has 29.) In addition, the model's logic can be systematically checked by reviewing a list of cell group definitions, including each group's formula, rather than the haphazard approach often adopted with traditional spreadsheets. Some experts recommend printing a list of all formulae as a way to check your model and show others "why you think your spreadsheet is right" [Bewig, 2005]. *Figure 9* below shows Sumwise's group manager panel, which lists the cell groups in the model and displays their properties, including: group definition, formula, and number of cells in the group.

$f_{(x)}$	R[!@Heading] C[@Quarter]	=SUM(C[Children])	6
$f_{(x)}$	R[@Subtotal] C[?@Month]@Quarter]	=SUM(R[Siblings])	8
$f_{(x)}$	R[?untagged] C[@Forecast]	=[@Month-1]*(1+[Growth])	15

*Figure 9*

There are two types of cell groups in Sumwise: formula cell groups, and input cell groups. Cells in a formula cell group all contain the same contents (string, value, or formula), whereas cells in an input cell group may contain different contents. The two types of cell groups are distinguished by different colour shading: grey for formula cell groups, and yellow for input cell groups. This reflects the recommendation by others that constants and formulas should be formatted differently [Raffensperger, 2003].

Once cell groups have been defined, model extension and modification become relatively easy. Additional rows and columns can be added and tagged appropriately, and the new cells will automatically inherit contents and formatting based on the pre-existing cell group definitions. Alternatively, tags for rows and columns can be changed, and cells will leave certain cell groups, and join other cell groups, based on the group logic and priority previously defined. For example, once actual data for *February* become known, you may

want *February* to take on the same structure and logic as *January*—so that actual revenue figures can be inputted. All that is required to achieve this is to select the *February* column and change the tag from *Forecast* to *Actual*.

The cell groups feature of Sumwise can eliminate the repetitive and often error prone ‘copy and paste’ requirement associated with traditional spreadsheets. Copying and pasting causes one piece of logic to be reflected in multiple physically separate cells—and the contents of any of these cells can then be changed independently of the other cells. Bewig highlights the “very common spreadsheet error of changing some but not all of a series of copied formulas” [Bewig, 2005]. In Sumwise, (ideally) each unique formula is written once, and then, using cell groups, automatically applied to other cells in the model that require that same formula. Sumwise then maintains the integrity of formulae across all cells in a formula cell group; if a change is made to the contents of any cell in a formula cell group, this change is made to all cells in the cell group. Thus, if an error is inadvertently made to the contents of one cell, this error is repeated in all cells of the cell group. In our opinion, this is an advantage of Sumwise, as we prefer that models contain big and obvious errors, rather than small and possibly hidden ones. (Actually, we prefer no errors—but big and obvious errors are easier to detect and correct.)

### 3 CONCLUSION

This paper has introduced a new type of spreadsheet known as Sumwise. Sumwise aims to improve model quality and reduce the risks associated with both quantitative and qualitative errors. Sumwise’s features and associated benefits are applicable to, and evident in, three key phases of a model’s lifecycle: design and construction, review and audit, and modification and extension.

The key features and benefits of Sumwise discussed are summarised below.

- User-defined row and columns labels make formulae easier to construct and understand. There is no disconnect between the syntax used and underlying semantics. Formulae say what they mean, and mean what they say.
- Tags can be applied to rows and columns and then used (in formulae) to reference data in a more logical way.
- Error prone copying and pasting of formulae can be avoided through the use of cell groups.
- Sumwise maintains the integrity of formulae in cells that are defined to be part of the same formula group. This aids detection of certain mechanical errors, such as where the contents of one or more cells are changed inadvertently. It also helps prevent certain types of qualitative errors, such as where a formula is overwritten with a hard coded value.
- Sumwise displays a list of cell groups and their formulae, enabling developers and reviewers to systematically check the model’s logic.
- Sumwise facilitates and simplifies common model modification and extension operations, that are difficult to achieve with traditional spreadsheets, including:
  - Inserting new rows and columns, and automatically applying formulae and formatting to the new cells (for more than just trivial data-extension operations); and
  - Changing the logic of rows or columns from one type to another—for example when changing a column from *forecast* to *actual*.

#### 4 REFERENCES

- [Aldridge, 2003] Aldridge, S. (2003), "Fresh Fields", *Financial Management (UK)*, June 2003.
- [Berglas and Hoare, 1998] Berglas, A., Hoare, P. (April 1998), "Error Free? Spreadsheet risks and techniques", *Australian CPA*, pages 42-45.
- [Bewig, 2005] Bewig, P. (July 2005), "How do you know your spreadsheet is right?", [www.eusprig.org/hdykysir.pdf](http://www.eusprig.org/hdykysir.pdf), pages 5 and 8.
- [Hellman, 2005] Hellman, Z. (2005), "Breaking out of the Cell: On the Benefits of a New Spreadsheet User-Interaction Paradigm", *EuSpRIG Conference 2005*.
- [McKeever, McDaid and Bishop, 2009] McKeever, R., McDaid K., Bishop, B. (2009), "An Exploratory Analysis of the Impact of Named Ranges on the Debugging Performance of Novice Users", *EuSpRIG Conference 2009*, page 12.
- [Panko and Ordway, 2005] Panko, R., Ordway, N. (2005), "Sarbanes-Oxley: What About All the Spreadsheets", *European Spreadsheet Risks Int. Grp. (EuSpRIG) 2005 15-47*, at page 28.
- [PWC, 2004] PriceWaterhouseCoopers. (2004), "The Use of Spreadsheets: Considerations for Section 404 of the Sarbanes-Oxley", <http://whitepapers.techrepublic.com.com>, accessed 20 May 2010.
- [Raffensperger, 2003] Raffensperger, John F. (2003) "New Guidelines for Spreadsheets", *International Journal of Business and Economics*, 2, 2, 141-154.