

Integrity Control of Spreadsheets: Organisation & Tools

K. Rajalingham, D. Chadwick

*Information Integrity Research Group,
School of Computing & Mathematical Sciences, University of
Greenwich,
Wellington Street, Woolwich, London SE18 6PF,
United Kingdom
Phone: +44 (0)181 331 8510
Fax: +44 (0)181 3318665
Email: rk09@gre.ac.uk*

Abstract

This paper describes a new approach to the provision of a software engineering discipline for the spreadsheet life-cycle: from requirements, to design, implementation and subsequent maintenance. This approach addresses the widespread problem of spreadsheet errors, which are analysed in detail and the outcome of which is a more comprehensive classification of errors than presented before. It is accompanied and supported by appropriate examples.

This new approach differs from other approaches in that it is based on an analysis of spreadsheet structure. The elements of this approach are explained and illustrated with examples. It's potential for integrity control is also discussed, especially based on the types of errors it will detect, reduce or prevent.

Keywords

Spreadsheets, errors, auditing, end-users, software engineering, modularisation, coupling

1 INTRODUCTION

1.1 The Problem and its Magnitude

Numerous recent publications have indicated the seriousness of spreadsheet errors and their adverse impact or potential impact on businesses.

A recent financial model review by KPMG Management Consulting, London (KPMG, 1997) confirms the frequency and seriousness of spreadsheet errors. Their report states that in 95% of the financial models reviewed, at least 5 errors were found. The review also reveals alarming statistics pertaining to defects in spreadsheet development, addressing the project management, technical and analysis aspects. As a result of these findings, KPMG Management Consulting (London) has engaged in a collaboration with the Information Integrity Research Group of the University of Greenwich, in a research into integrity control in spreadsheets. This paper is part of that research.

An article in *New Scientist* (Ward, 1997) has reported that a study conducted by the British accounting firm Coopers & Lybrand found errors in 90% of the spreadsheets audited. This is an extremely high figure and if the errors went undetected, it could have had a devastating effect on the business. The same source has also indicated that a decade's worth of research findings of Professor Raymond Panko at the University of Hawaii revealed that spreadsheets had a dangerously high rate of errors.

There are also publications from more than a decade ago that clearly state that spreadsheet errors have caused serious disruption of business. Although these cases are not based on formal research, they do show that spreadsheet errors were considered important enough to be reported in the general business and computing press. For instance, according to an article in *Personal Computing* (Ditlea, 1987, January), a Houston consultant with Price Waterhouse had found 128 errors in 4 spreadsheet models that had already been in use for months.

These reports indicate that the occurrence of spreadsheet errors is a major problem for businesses and needs to be addressed urgently.

1.2 The Need for a New Approach to the Discipline for Spreadsheet Development

Findings from research carried out over the last few years show the need for a new approach or discipline for spreadsheet development. This is evident from the call, or implied call, for a new approach in many recent publications. Panko and

Halverson (1996, January) distinctly state the need to adopt strict programming disciplines in dealing with complex spreadsheets.

The journal paper by Panko (Panko, forthcoming) says that there is an obvious need to begin adopting traditional programming disciplines due to the similarity between spreadsheet errors and programming errors. The paper also states that there is far too little knowledge of spreadsheet errors, which implies that much more research has to be undertaken into spreadsheet errors.

It is stated by Hendry and Green (1994, January) that instead of creating the whole spreadsheet first and then checking for errors, errors ought to be checked for at various stages of the development process. This will make it easier to trace and correct errors. This strategy of stage-by-stage component testing is a software engineering principle.

Based on these published reports, we can come to the conclusion that there is a need for the imposition of a programming or software engineering-based discipline in spreadsheet development. This will help address the currently major problem of spreadsheet errors.

2 CLASSIFICATION OF SPREADSHEET ERRORS

A thorough review of literature relevant to spreadsheet development and errors reveals that very little research has been done on studying specific errors that occur in spreadsheets. One such research was undertaken by Chadwick et al (1997, December).

This paper presents a more comprehensive classification of spreadsheet errors as shown in *Figure 1*. This classification is illustrated with examples of specific errors. This is based on work already done as well as original work produced by the authors of this paper.

In the broadest terms, spreadsheet errors can be divided into two major categories, namely *software errors* and *user errors*.

Software errors

Software errors are errors made by the spreadsheet software and therefore their occurrence is generally beyond the control of users, although they can, when aware, take remedial action.

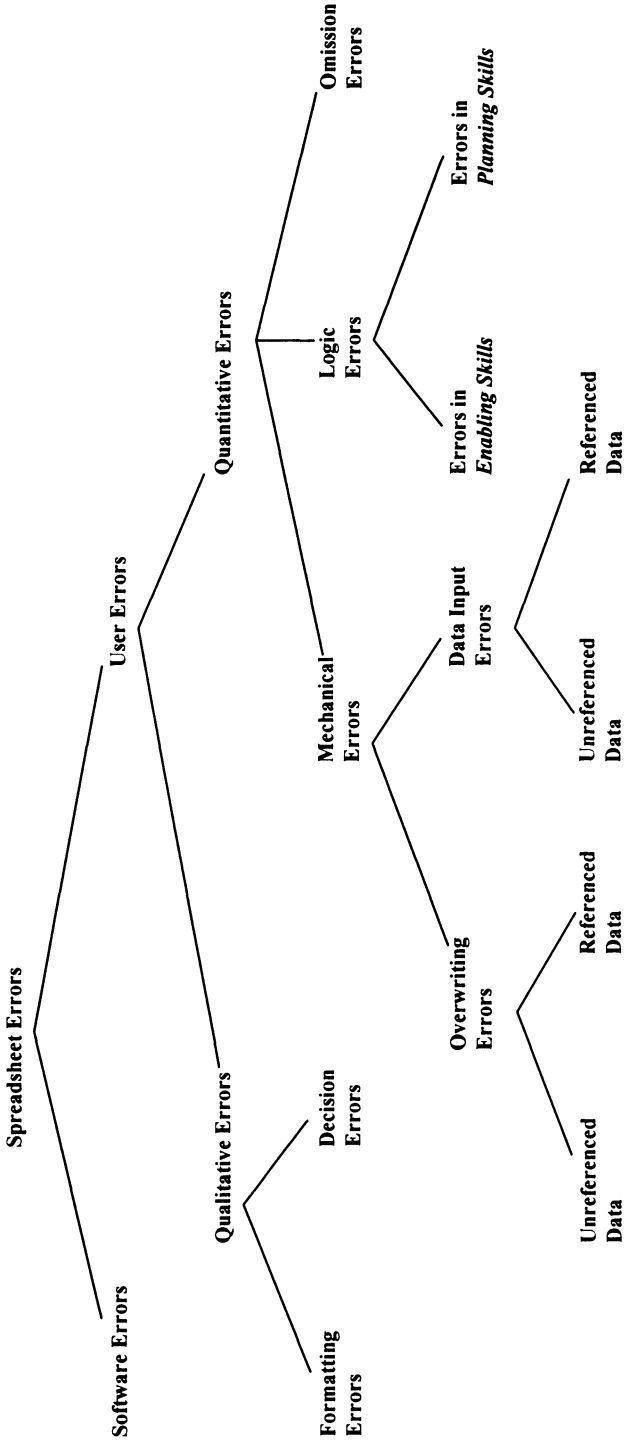


Figure 1 Taxonomy of spreadsheet errors.

Example: Year 2000 Error

In MS Excel 5 for instance, for any entry of a date (without the century) before 01/01/20, the century is assumed to be the 21st century while for any entry of a date (without the century) after 01/01/20, the century is assumed to be the 20th century. This problem, of course, can be avoided if the year is explicitly entered with the century e.g. **09/02/1915**, **03/12/2060**.

User errors

User errors are errors committed by the user and can be prevented, detected and corrected. They can be divided into two major categories at the highest level, namely *qualitative errors* and *quantitative errors*. This is based on Ray Panko's framework for classifying spreadsheet errors (Panko & Halverson, 1996, January).

The spreadsheet model below has been adapted from the one illustrated in the paper by Chadwick et al (Chadwick et al, 1997, December). Most of the errors explained in this section will be based on this model.

	C	D	E	F	G	H	
		Number of Staff	Day Wages £	Night Wages £	Total Wages £	Average Wage £	5
Grade 1		1	17700.50	0.00	=SUM(E6:F6)	=G6/D6	6
Grade 2		3	45540.00	1400.55	=SUM(E7:F7)	=G7/D7	7
Grade 3		9	122340.00	2000.00	=SUM(E8:F8)	=G8/D8	8
Grade 4		12	102350.25	0.00	=SUM(E9:F9)	=G9/D9	9
Grand Total		=SUM(D6:D9)	=SUM(E6:E9)	=SUM(F6:F9)	=SUM(G6:G9)	=G10/D10	10

(Input)



	C	D	E	F	G	H	
		Number of Staff	Day Wages £	Night Wages £	Total Wages £	Average Wage £	5
Grade 1		1	17700.50	0.00	17700.50	17700.50	6
Grade 2		3	45540.00	1400.55	46940.55	15646.85	7
Grade 3		9	122340.00	2000.00	124340.00	13815.56	8
Grade 4		12	102350.25	0.00	102350.25	8529.19	9
Grand Total		25	287930.75	3400.55	291331.30	11653.25	10

(Output)

Figure 2**Staff Budget Costs 1995-1996**

Qualitative errors

Qualitative errors are errors that do not immediately produce incorrect numeric values on the spreadsheet itself. However, they do reduce the quality of the spreadsheet produced as it becomes prone to misinterpretation on the part of the user. As a result, it also becomes more difficult to update and maintain the model. A more detailed investigation into qualitative errors revealed that they can be generally divided into two different types, namely, **formatting errors** and **decision errors**.

Formatting errors

Formatting errors are qualitative errors that occur due to lack of uniformity in the formatting of similar data. This could lead to an incorrect interpretation of their values. An example of a formatting error is the *Money Format: Commas, Standard, Decimalisation error*.

Example 1: Money Format: Commas, Standard, Decimalisation

This error is mentioned with an example by Chadwick et al (1997, December) under *Errors in 'Planning Skills'*. It is a common qualitative error where the cell format is specified as *General* on the spreadsheet. Consequently, the figures have varying decimal places and make it difficult to identify a number that is incorrect by a magnitude of 10, 100 etc.

Decision errors

Decision errors are qualitative errors that occur due to an incorrect decision, choice or assumption that leads to the intentional entry or referencing of an inappropriate piece of data, especially numeric. As far as qualitative errors are concerned, decision errors are far more difficult to detect compared to formatting errors.

Example 2: Qualitative Error Resulting from the Referencing of Non-current Data

This is an example of a qualitative error produced as a result of referencing a piece of data that has become invalid due to time lapse. In the example given below (*Figure 3*), this piece of data is the exchange rate from British Pounds (£) to Ringgit Malaysia (RM) contained in cell D2. If the exchange rate undergoes acute fluctuations and the changes are not reflected in cell D2, the calculation in cell A8 produces a value that is invalid. This is a qualitative error and any decision made based on this value would be unreliable.

A	B	C	D	
	Tea(£)	Milk(£)	Exchange Rate (£ to RM)	1
1994	450	560	7.3	2
1995	904	900		3
1996	872	800		4
1997	123	234		5
				6
Total Sale of Tea & Milk in RM				7
35354				8

Figure 3 Qualitative, decision error.

Example 3: Hard-coding of a Formula

The hard-coding of a formula is another example of a qualitative, decision error. This error decreases the quality of the spreadsheet by making it much less flexible. Referring to *Figure 2*, if the formulae in column H were hard-coded e.g. =G8/9 (in cell H8) instead of =G8/D8, and if any of the values in column D (number of staff) changed, the formula in column H of the same row would have to be re-written. This is just a simple example to illustrate the concept of hard-coding being a source of error.

Quantitative errors

Quantitative errors, in turn, are numerical errors that lead to incorrect bottom-line values. This definition is offered by Panko and Halverson (1996, January) who also state that a simple trichotomy captures the diversity of quantitative errors reasonably well. They are *mechanical*, *logic* and *omission* errors.

Mechanical errors

Mechanical errors are simply typing errors. Though quite frequently occurring, they have a high chance of being spotted and corrected immediately by the person committing the error. Some, however, do go undetected and could lead to incorrect values in other cells. Mechanical errors can be divided into two distinct categories. They are **overwriting errors** and **data input errors**.

Overwriting errors

An overwriting error is said to have occurred when a correct piece of data or formula is unintentionally replaced or overwritten by an incorrect piece of data. This error is also mentioned under *Errors in 'Enabling Skills'* by Chadwick et al (1997, December). There are two types of overwriting errors; **overwriting of unreferenced data** and **overwriting of referenced data**. Each of these can either be a *formula* or a *directly entered value*.

Data input errors

These are errors made by users while entering data into the spreadsheet. Just like overwriting errors, there are two types of data input errors; **error in input of unreferenced data** and **error in input of referenced data**. Unlike the former, the latter produces an incorrect piece of data in a cell which is subsequently referenced by at least one formula. Consequently, the return value sent to the formula cell(s) are also incorrect. Each of these incorrect entries can either be a *formula* or a *directly entered value*. As a result of carelessly entering incorrect cell or range addresses into formulae, the formulae themselves produce incorrect return values, for instance, by referencing *blank cells* or *non-numeric cells*.

Logic errors

Panko and Halverson (1996, January) define logic errors as incorrect formulae due to choosing the wrong algorithm or creating the wrong formulae to implement the algorithm. They involve entering the wrong formula because of a mistake in reasoning. Logic error rates are higher than mechanical error rates (Panko, forthcoming). Although there are several ways of subdividing logic errors, we use the classification of spreadsheet errors into *Errors in 'Enabling Skills'* and *Errors in 'Planning Skills'* as proposed by Chadwick et al (1997, December). However, its use will be confined to quantitative, logic errors only.

Logic errors in 'enabling skills'

Enabling skills are those needed to permit the user full use of the functions and capabilities of the particular spreadsheet package in use, with an understanding of the spreadsheet principles, concepts, constructs, reserved words and syntax. This definition has been adapted from that given by Chadwick et al (1997, December). The two errors in *enabling skills* given below are taken from the same paper.

Example 4: Relative and Absolute Copy Problem

The relative copy causes cell references in a copied formula to alter row and column references relative to the original cell copied. People often make the false assumption that the software will automatically adapt the cell references wherever they happen to copy.

Example 5: Circular References

This error frequently occurs in totals where the formula uses its own value in its calculation. This error will give a run-time error message and so probably occurs infrequently. With reference to *Figure 2*, an example of this error would be the entry of the formula =SUM(D6:D10) into cell D10, instead of =SUM(D6:D9).

Logic errors in 'planning skills'

Planning skills are those required to analyse the business function in order to design the data model which is to be represented electronically by the spreadsheet model. These skills enable the user to identify business functions which are suitable for modelling with a spreadsheet and how this modelling is to be done. This requires thorough knowledge of business functionality and requirements for both the present and the future. This explanation of *planning skills* has been adapted from that given by Chadwick et al (1997, December). The three errors in *planning skills* given below are taken from the same paper.

Example 6: The TOTALS Problem

The error is said to have occurred when the column total and the row total are dissimilar when they should logically have produced the same result, often due to the lack of cross-checking. For instance, referring to *Figure 2*, the value in cell G10 should be the same whether the formula is entered as =SUM(G6:G9), which is the column total, or =SUM(E10:F10), the row total. This error can be easily picked up if the formula =IF(SUM(G6:G9)>SUM(E10:F10), "Error", SUM(G6:G9)) is entered in cell G10. An error message would then be displayed in the cell if the column and row totals were dissimilar.

Example 7: AVERAGE Problem

This happens when the average function AVERAGE(Rg) is applied incorrectly due to little understanding of its appropriateness. An example of this type of error would be the entry of the formula =AVERAGE(E6:F6) in cell H6, instead of =G6/D6, on the spreadsheet shown in *Figure 2*.

Example 8: Percentage Problem

This error occurs when the formula to calculate percentage is incorrectly written, either due to lack of knowledge of what a percentage is or BODMAS (Brackets, Of, Division, Multiplication, Addition, Subtraction) by which the spreadsheet identifies precedence in calculations e.g. B2/A2*100, B2*100/A2 or B2*A2/100 instead of A2/B2*100 or A2*100/B2. This is based on *Figure 4* below.

A	B	C	
Night Wages £	Total Wages £	Night Wages %	1
1400.00	46940.00		2

Figure 4 Percentage problem.

Omission errors

Omissions are things left out of the model that should be there. They often result from a misinterpretation of the situation. Human factors research has shown that

omission errors are especially dangerous, because they have low detection rates (Panko & Halverson, 1996).

3 EXISTING TOOLS AND METHODS FOR INTEGRITY CONTROL IN SPREADSHEETS

If we apply the *generic information systems development life cycle* to spreadsheets, then referring to *section 2*, it has been found that the different types of errors appear at different stages of the spreadsheet life cycle. There are various existing tools to help with integrity control at these different stages. Among them are the Microsoft Excel audit tool (Microsoft Corporation, 1994), the Spreadsheet Professional audit tool for Microsoft Excel by Spreadsheet Innovations, the DiAntonio method (DiAntonio, 1994) and Chadwick, D. et al's 5-step methodology incorporating the 3A's approach (Chadwick et al, 1997, December). This is illustrated in *Figure 5*.

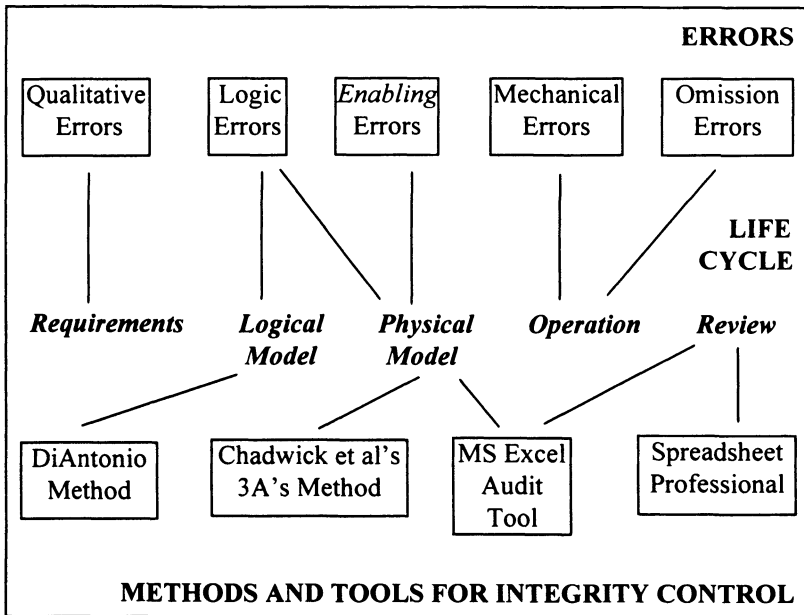


Figure 5 Errors and methods/tools associated with different stages of the spreadsheet life cycle.

Though these tools have to an extent reduced errors in spreadsheets, they have not been entirely successful as the phenomenon still persists.

Microsoft Excel Audit Tool

This tool which is part of the *Microsoft Excel* software enables the user to easily trace the **precedents** or **dependants** of any cell. The precedents of a cell are the cells **referenced** by it while the dependants of a cell are the cells that **reference** it. When tracing the precedents of a cell, an arrowed line is drawn from each precedent cell, pointing to the dependant cell. On the other hand, when tracing the dependants of a cell, an arrowed line is drawn from that cell pointing to each of its dependant cells. An example is given by Chadwick et al (1997, December).

E.g. Referring to *Figure 2*, the precedents of cell **H6** are cells **D6** and **G6**, while the dependants of cell **G6** are cells **H6** and **G10**.

Apart from that, the audit tool also offers a facility for the user to attach a note or description to a cell.

Spreadsheet Professional for Microsoft Excel by Spreadsheet Innovations

This tool, which is an add-on to *Microsoft Excel*, has served as a rather useful auditing tool. It has various functions to help detect errors in the spreadsheet model. Among the significant functions are the *calculation checker* and the *cell translation* facility.

Calculation Checker

This function enables the user to view the contents, potential error and precedents of a formula cell. Based on the spreadsheet model shown in *Figure 2*, if a 'calculation check' is done on cell H10 and the 'Show Precedents' option is subsequently selected, the precedents of cell H10 are displayed in the following style (note that this, however, does not show the relationship between them in the same display i.e. =G10/D10):

Ref	Translation	Value
G10	Grade Total	291331.30
D10	Grade Total	25

The translation for each precedent (cell or range) is given in terms of location, corresponding row heading and values.

Cell Translation

The result of a 'Cell Translation' for cell E10 would produce information as given below:

Sheet1!E10	= SUM (E6:E9)
Grade Total	= SUM (Grade 1:Grade 4)
287930.75	= SUM (17700.50:102350.25)

The formula in the cell is first translated into a form where each cell address is replaced with its corresponding row heading and then presented in a form where each cell address is replaced with its corresponding value.

DiAntonio's Method for Spreadsheet Development

DiAntonio has proposed a structured method consisting of six distinct steps for the construction of spreadsheets.

- Step 1:** The problem is understood and defined.
- Step 2:** Isolation of facts is done by splitting the spreadsheet into two parts, one for the *facts* and one for the *solution*.
- Step 3:** The solution is formatted or designed and it uses data from the *facts* part of the spreadsheet.
- Step 4:** The program is tested with sample data.
- Step 5:** The program is evaluated in terms of functionality, headings, labels and format.
- Step 6:** The program is documented either on the spreadsheet itself or in hard copy.

Chadwick et al's 5-step Methodology Incorporating the 3A's Approach

Chadwick et al (1997, December) propose a five-step methodology for spreadsheet auditing, that incorporates the 3A's (appropriateness, accuracy, about-right) Approach. A brief outline of this methodology is given below.

- Step 1:** Checking the appropriateness of the formula applied, from a logical point of view, based on the underlying business model.
- Step 2:** Checking the accuracy of the formula entered based on a correct interpretation of the data model.
- Step 3:** Checking if the resulting numeric value of the cell is about right.
- Step 4:** Validating a formula copy to a cell or a range.
- Step 5:** Modularising the spreadsheet by breaking it down into separate logical areas (modules).

4 A MODULAR APPROACH TO REDUCING ERRORS IN SPREADSHEETS

4.1 Rationale for a Modular Approach

This approach applies the concept of *modularisation* to the design and structuring of spreadsheets. Based on *section 1.2*, it is evident that there is a need for the introduction of strict development disciplines in spreadsheeting, as well as the adoption of traditional programming or software engineering-based principles, concept and practice. The modular approach to spreadsheet development is a fundamental principle of software engineering. Support for the modular approach comes from DiAntonio (1986) and Chadwick et al (1997, December) but is weakly defined in both these sources (refer to *section 3* for a brief outline of both these methods).

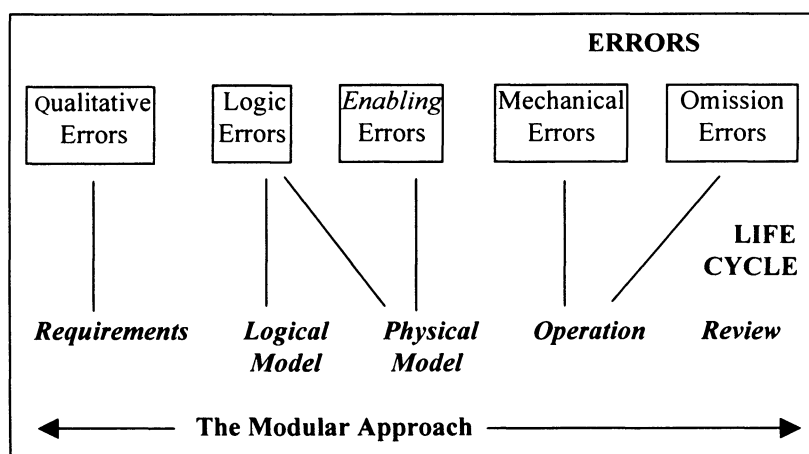


Figure 6 The modular approach covers all stages of the spreadsheet life cycle.

Within the context of spreadsheeting, *modularisation* refers to the structuring of the spreadsheet model into distinct blocks or modules with data being passed between them. The elements of this approach are discussed in detail in the next section (4.2).

Another fundamental principle of this modular approach is the concept of *coupling*. This is also a software engineering concept. Coupling may be defined simply as the link between modules resulting from the passing of data between them. This is also discussed in *section 4.2*.

4.2 A Modular Structure for Spreadsheets

The modular approach dictates the division of the physical model (spreadsheet data) into distinct modules, with each cell value having a column heading and row heading associated with it.

Most, if not all, models can, with minor adjustments, be made to conform to this structure. The fact that the spreadsheet is separated into separate blocks or modules suggests that a modular approach is being taken, based on an analysis of spreadsheet structure.

The term we have given to a distinct module of the business spreadsheet is an **extent**. An extent can be defined as a matrix representing a logical area or module of the spreadsheet. An extent is a range with special properties. It has various special characteristics.

The minimum size of an extent is a 2 by 2 range (4 cells). The first column of an extent contains the row headings while the first row of an extent bears the column headings. Every cell within a particular column (except the first column) is associated with the same column heading, which occupies the top cell of that column. Similarly, Every cell within a particular row (except the first row) is associated with the same row heading, which occupies the left-most cell of that row.

Column headings and row headings of an extent must be defined by the user. No two cells can have exactly the same combination of column heading and row heading as there cannot be two or more column headings or row headings with the same name, although a column heading can share the same name with a row heading.

The following steps must be taken in defining an extent and its boundary:

Step 1:

All adjacent data cells within a column, that logically have the same column heading, are given a column heading which occupies the cell immediately above the first data cell of the column. It is possible for there to be only one data cell in the column.

Step 2:

All related columns of data are then identified and placed next to the first data column, to form a table. The top cell of each of these columns bears the column heading. The columns are only considered to be related if they are of the same

height and when put next to each other, every cell within a particular row must logically have the same row heading.

Step 3:

The row heading is entered into the cell to the left of the first cell of each row within the table. The entire matrix now is called an extent and represents a module or logical area of the spreadsheet. The spreadsheet model shown in *Figure 2* is an example of an extent. The resulting generic structure of an extent, after these three steps are performed, is as shown in *Figure 7* below.

q	q+1	q+2	l	
q,r						r
						r+1
						r+2
						.
						.
						.
					l,m	m

Figure 7 Generic structure of an extent.

With reference to *Figure 7*,

- The **boundary** of the extent is defined in terms of its top-left cell (column q, row r) and bottom-right cell (column l, row m).
- **Column headings** are contained in cells in the first row (row r) of columns q+1 to column l.
- **Row headings** are contained in cells in the first column (column q) of rows r+1 to row m.
- **Data values** are contained in all the other cells except the top-left cell (column q, row r)

The concept of *coupling* was introduced in the previous section. Within the context of this modular approach, there is coupling between two modules when a formula in one references data in the other.

For a particular instance of coupling between two modules, the referencing module is the *dependant* and the referenced module is the *precedent*. Coupling in this

paradigm is the precedent-dependant link between cells in different modules. This is illustrated in *Figure 8*. For further clarification of these terms, refer to *section 3 (Microsoft Excel Audit Tool)*.

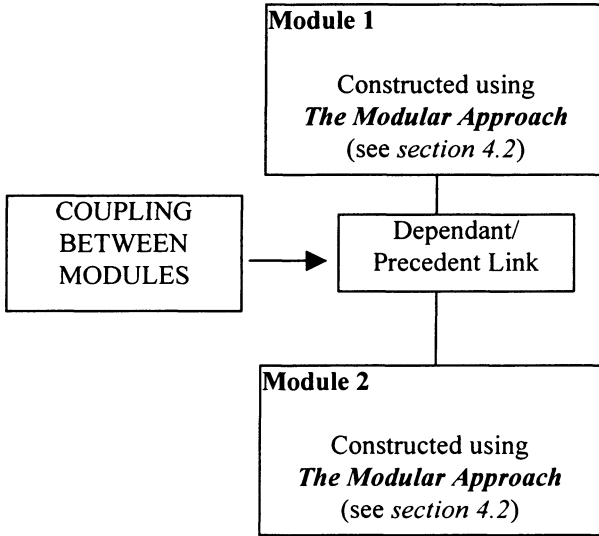


Figure 8 Precedent-dependant link/coupling between modules.

4.3 Visual Representation of Elements of a Formula in Natural Language Form

As a by-product of the modular approach, it is possible to visually represent the elements of a spreadsheet formula. This is aimed at minimising errors in spreadsheets.

Most of the errors that occur in spreadsheets are those concerning formulae. These are also the sort of errors that have the greatest impact on the business and its operations. Numerous types of errors can be made when entering formulae into cells, one of the most significant of which is the entry of incorrect cell addresses or references.

When such errors are committed, it is often difficult to detect and correct them based on the original structure of the formulae that appears in the formula bar of the spreadsheet screen. This is primarily due to the use of cell addresses in the formulae to refer to data.

This problem can therefore be overcome if formulae were represented in a more visual, English-like and comprehensible form. This will certainly facilitate the validation and audit of spreadsheet formulae. The proposed technique for visually representing spreadsheet formulae will be able to present formulae in such a form. Any software tool used to implement this technique will be able to convert a formula written by a user in conventional form, expressed in terms of cell addresses, into a form that is more readable and visual. This is done mainly by displaying the corresponding column and row headings of each cell referenced by a formula. This makes every spreadsheet cell value meaningful and also ensures that the user understands this meaning when creating and using the spreadsheet. Several methods of presenting such formulae have been developed in this form. These different methods are given below based on the spreadsheet model in *Figure 2*.

C	D	E	F	G	H	
	Number of Staff	Day Wages £	Night Wages £	Total Wages £	Average Wage £	5
Grade 1	1	17700.50	0.00	17700.50	17700.50	6
Grade 2	3	45540.00	1400.55	46940.55	15646.85	7
Grade 3	9	122340.00	2000.00	124340.00	13815.56	8
Grade 4	12	102350.25	0.00	102350.25	8529.19	9
Grand Total	25	287930.75	3400.55	291331.30	11653.25	10

Segment of *Figure 2*

Referring to *Figure 2*, formulae are present in the following cells:
G6 to G10, H6 to H10, and D10 to F10.

The formulae selected to illustrate the various methods are given below:

CELL	FORMULA
F10	=SUM(F6:F9)
H10	=G10/D10

Method 1: *Algebraic English*

Cell Representation of the Formula

F10 =SUM(Night Wages £_Grade1:Night Wages £_Grade4)

H10 =Total Wages £_Grand Total/No of Staff_Grand Total

This method simply converts each cell address to its corresponding column and row headings but retains the binary operators.

Method 2: Fully English

Cell Representation of the Formula

F10 *Night Wages £_Grand Total* = SUM (*Night Wages £_Grade1* (to) *Night Wages £_Grade4*)

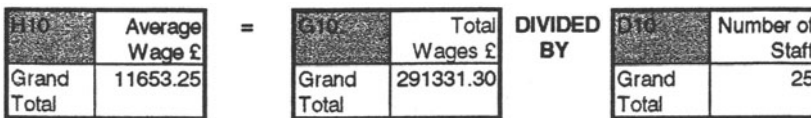
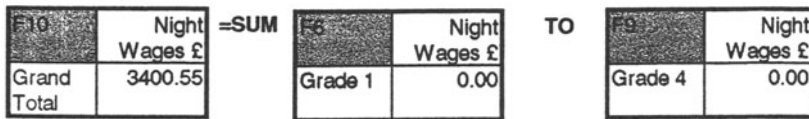
H10 *Average Wage £_Grand Total* = *Total Wages £_Grand Total* (divided by) *No of Staff_Grand Total*

This method converts each cell address to its corresponding column and row headings as well as each binary operator from symbol to natural language.

Method 3: Graphic Display

Cell: F10

Representation of the Formula:



This is the most visual of the three methods. It is also the method preferred and recommended by the authors of this paper. Apart from associating each cell address with its column and row headings, this method also displays the value contained in the particular cell. In each display, a different colour is used for each different column heading and row heading. For instance, in the first example, all three cells shown have column heading (*Night Wages £*) in the same colour (*blue*) text to indicate they are all in the same column. However, the row headings (*Grand Total*, *Grade 1* and *Grade 4*) are in different colours (*red*, *green* and *purple*) indicating that they are in different rows.

This approach may be successful in eliminating many of the errors referred to in *section 2*, particularly the following:

- i) **Qualitative errors**, especially **formatting errors**.
- ii) **Quantitative errors**, especially **logic errors**. A large number of **mechanical errors** can also be picked up through the use of *method 4* (*section 4.1*).

4.4. Survey

A survey was carried out to determine the preference of students to the visual methods described in *section 4.3*. The students were presented with four choices: the normal MS Excel formula style and the three mentioned above. They were asked to rank them in order of clarity and ease of understanding. There were 63 respondents to the questionnaire. 46 respondents (73%) indicated preference for the visual methods. 21 of them (34%) chose **method 3** (the graphic display) as the most clear and easiest to understand. It was the most appealing of the four choices, with the normal formula style (26%), the algebraic English (18%) and the totally English (22%).

With the use of colour in the questionnaire and a better understanding of the benefits of method 3, it is possible that a lot more students would have selected it as their first choice. The findings, however, do endorse research of the visual technique, especially the *graphic display*.

5 CONCLUSION

This paper has covered the magnitude of the problem of spreadsheet errors, a classification of these errors and an appreciation of where errors occur in the spreadsheet life cycle. A study of existing tools and methods for integrity control revealed that the whole approach was piece-meal. From this, developed the need to investigate a new approach to the discipline of spreadsheet building based on software engineering principles.

A modular approach encompassing the concepts of modularisation and coupling was discussed as a possible method of integrity control in spreadsheets. This in turn, gave rise to a visual technique to represent elements of spreadsheet formulae. All of these findings may now be integrated into a coherent spreadsheet development methodology.

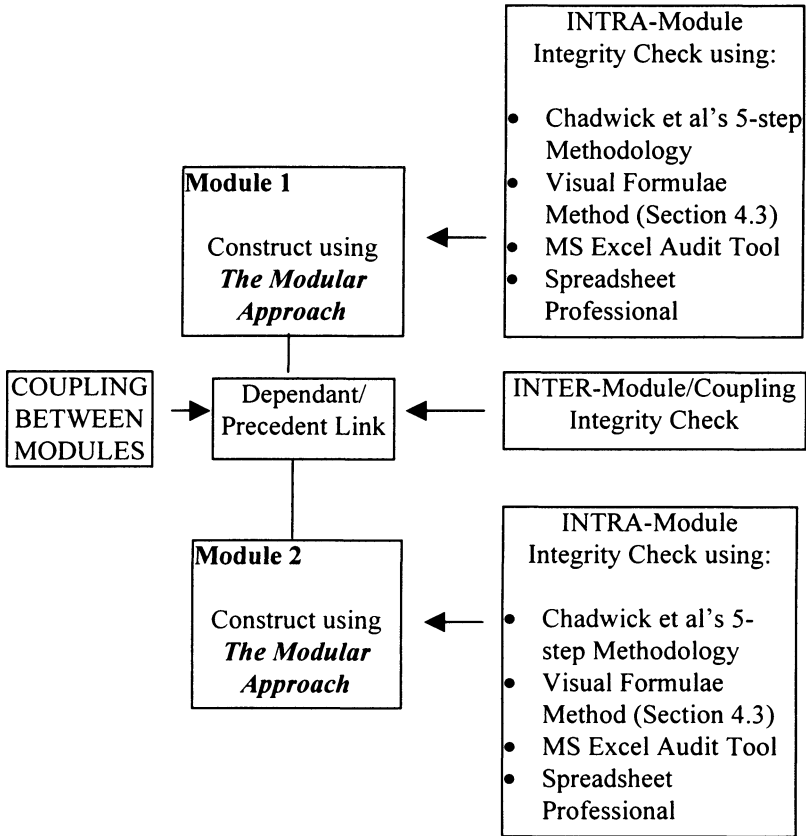


Figure 9 Spreadsheet development methodology.

A Spreadsheet Building Methodology could therefore be as follows:

- Step 1:** Create a logical model
- Step 2:** Create physical modules
- Step 3:** Perform INTRA-module integrity checks
- Step 4:** Perform INTER-module/coupling integrity checks

The principal advantages of this approach are as follows:

- i. It may be useful in all phases of the spreadsheet life-cycle, as shown in *Figure 10*.
- ii. The industry trend is towards Graphical User Interface (GUI), visual languages, etc.

- iii. The concept of modularisation is based on tried and tested techniques in use in software engineering and can be used to structure spreadsheets.
- iv. The visual technique could greatly help in the process of spreadsheet auditing, which is a key concern in many companies.
- v. It requires little training for users due to its simplicity.

Further work needs to be done on:

- i. A methodology/tool for capturing requirements.
- ii. A methodology/tool to aid modular structuring of the logical model.
- iii. Creation of an add-in software tool that facilitates modular structuring of the physical. model which will aid intra-module and inter-module integrity checking.
- iv. A methodology/tool for reducing *mechanical errors*.
- v. A methodology/tool for auditing completed spreadsheets.

6 REFERENCES

- Chadwick, D., Knight, J. and Clipsham, P. (1997, December) Information Integrity In End-user Systems. *Proceedings of the IFIP TC-11 Working Group 11.5 First Working Conference on Integrity and Internal Control in Information Systems*, Zurich, Switzerland.
- DiAntonio, A. E. (1986) *Spreadsheet Applications*. Prentice-Hall.
- Ditlea, S. (1987, January) Spreadsheets Can be Hazardous to Your Health. *Personal Computing* Vol 11, p60-69.
- Hendry, D. G. and Green, T. R. G. (1994, January) Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model. *International Journal of Human-Computer Studies* Vol 40, p1033-1065.
- KPMG (1997) Executive Summary: Financial Model Review Survey. KPMG Management Consulting, London.
- Microsoft Corporation (1994). *Microsoft EXCEL users Guide*.
- Panko, R. R. and Halverson, R. P., Jr. (1996, January) Spreadsheets on Trial: A Framework for Research on Spreadsheet Risks. *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, Maui, Hawaii.

Panko, R. R. (forthcoming) What We Know About Spreadsheet Errors. Conditionally Accepted for Publication in the *Journal of End User Computing's* Special issue on Scaling Up End User Development, Spring 1998.

Ward, M. (1997) Fatal Addition. *New Scientist* 16th August.

7 BIOGRAPHY

Kamalasen Rajalingham

I am currently pursuing a doctorate (PhD) in Computing & Information Technology at the University of Greenwich, United Kingdom. My specific research area involves studying spreadsheet errors and developing a framework for the quality and integrity control of spreadsheets. I am working within the Information Integrity Research Group of the School of Computing & Mathematical Sciences. I am also a part-time lecturer at the University. I teach Introduction to Databases, Database Management and Human Factors, and conduct tutorials for Mathematics, Statistics, Information Systems Security and several other units.

David Chadwick

I am a Senior Lecturer at the University of Greenwich. I teach Audit Security & Law, Information Systems Security and several other units. I am also the Head of the University's Information Integrity Research Group, whose primary function at present is to address the issue of spreadsheet errors. I presented a paper entitled *Information Integrity In End-user Systems* at the *IFIP TC-11 WG 11.5 First Working Conference on Integrity and Internal Control in Information Systems* held in Zurich, Switzerland in December, 1997. Apart from that, I have also written several articles on spreadsheet errors.