14th EuSpRiG Annual Conference
Spreadsheet Risk Management

University of Greenwich, London 4-5 July 2013
European Spreadsheet Risks Interest Group

EuSpRIG 2013 Conference

Spreadsheet Risk Management

Conference Co-Sponsored By:

CIMCON Software
Information Systems Audit & Control Association
(Northern England Chapter)
Cardiff Metropolitan University
University of Greenwich
F1F9 Spreadsheet Engineering
Auditnet.org
Systems Modelling
Operis Plc
Spreadsheet Engineering

Proceedings Editors:

Simon Thorne & Grenville Croll

July 2013
University of Greenwich
EuSpRIG 2013 Conference

“Spreadsheet Risk Management”

EUSPRIG 2013 ORGANISING COMMITTEE

Pat Cleary (Chair), University of Wales Institute Cardiff, UK
Patrick O’Beirne, Systems Modelling, Co. Wexford, Ireland
David Colver, Operis PLC, London, UK
Simon Thorne, Cardiff Metropolitan University, UK
Grenville J. Croll, Spreadsheet Engineering, Bury St Edmunds, UK

EUSPRIG 2013 PROGRAMME COMMITTEE

Simon Thorne (Chair), Cardiff Metropolitan University, UK
Grenville J. Croll, Spreadsheet Engineering, Bury St Edmunds, UK
Jocelyn Ireson-Paine, Spreadsheet Factory
David Ward, Baker Tilly, London, UK
David Colver, Operis PLC, London, UK
# Table of Contents

Practical Challenges with Spreadsheet Auditing Tools  
*Daniel Kulcz, and Jan-Peter Ostberg,*  
*University of Stuttgart, Germany*  

Multidisciplinary Engineering Models: Methodology and  
Case Study in Spreadsheet Analytics  
*David Birch, Helen Liang, Paul H J Kelly, Glen Mullineux,*  
*Tony Field, Joan Ko and Alvise Simondetti,*  
*Imperial College London, UK*  

The misuse of spreadsheets in the nuclear fuel industry  
*Simon Thorne,*  
*Cardiff Metropolitan University, UK.*  

XLSearch: A Search Engine for Spreadsheets  
*Michael Kohlhase, and Corneliu Prodescu,*  
*Jacobs University Bremen, Germany*  

Excel 2013 Spreadsheet Inquire  
*Patrick O’Beirne,*  
*Systems Modelling Ltd, Ireland*  

A Quality Checklist for Spreadsheets  
*Henk Vlootman,*  
*Felienne Hermans*  

Eliminating errors through tables, a new type of  
worksheet technology  
*Ted Hawkins, Morphit, UK, Andrew Lemon and Alec Gibson,*  
*The Edge Software Consultancy Ltd, UK*  

Spatial Modelling Techniques in Microsoft Excel  
*Stephen Allen,*  
*ACBA, UK*
PREFACE

Dear Colleagues,

You are all very welcome to the fourteenth annual conference of the European Spreadsheet Risks Interest Group. Another international assembly, arriving in the wake of scandals in the news about spreadsheet errors; most notoriously, the Reinhart and Rogoff "Austerity spreadsheet" debacle which provoked more questioning of their methodology. We are glad to see that the Anglo-Saxon dominance of spreadsheet research is giving way to a more European and global interest. It's good to see more risk managers, grappling with the practical issues of steering a spreadsheet control project.

I would like to acknowledge with gratitude our sponsors who have supported us financially and professionally. Our logo sponsors this year are CIMCON, and I recommend you to pick up their brochures and learn about their services. We depend also on the support of Cardiff Metropolitan University.

It is my pleasure to once again acknowledge the keen work of our conference and programme organiser Simon Thorn at Cardiff Metropolitan. The committee also depends upon the wise counsel and active support of David Colver of Operis, Grenville Croll of Spreadsheet Engineering, Angela Collins (secretary), Morten Siersted of F1F9, Jocelyn Paine of Spreadsheet Factory and Colin Kerr of Standard Life.

Representatives of these organisations have contributed a great amount of expertise in the organising of this conference, the publicity, the proceedings, and much more committee work in the background.

Unfortunately, Professor Ray Panko is unable to travel to join us this year despite firm plans to do so. We nevertheless continue to tackle the testing questions Ray has tried to raise in every organisation attempting to herd their many spreadsheets: "It's not my job to test software", "Our deadlines don't give us time to test", "How much testing is enough?", "You're not seriously saying we have to check everything every time!", "Our spreadsheets are too big to test", "We don't need to test, we are good at our job, that's what we're paid for", "Our spreadsheets are tested every time by virtue of using them".

So – take part in these debates – that's how we all learn!

Patrick O'Beirne

Chair 2013

Patrick@eusprig.org
Practical Challenges with Spreadsheet Auditing Tools

Daniel Kulesz, Jan-Peter Ostberg
Institute of Software Technology,
University of Stuttgart, Germany
firstname.lastname@informatik.uni-stuttgart.de

ABSTRACT

Just like other software, spreadsheets can contain significant faults. Static analysis is an accepted and well-established technique in software engineering known for its capability to discover faults. In recent years, a growing number of tool vendors started offering tools that allow casual end-users to run various static analyses on spreadsheets as well. We supervised a study where three undergraduate software engineering students examined a selection of 14 spreadsheet auditing tools, trying to give a concrete recommendation for an industry partner. Reflecting on the study’s results, we found that most of these tools do provide useful aids in finding problems in spreadsheets, but we have also spotted several areas where tools had significant issues. Some of these issues could be remedied if spreadsheet auditing tool vendors would pick up some ideas of static analysis tools for traditional software development and adopt some of their solution approaches.

1. INTRODUCTION

Just like other software, spreadsheets are known to contain faults. Spreadsheets are often developed by end-users. Several experiments indicate that spreadsheets are more fault-prone than other software [Powell et al., 2008]. The EuSpRiG has collected evidence of cases where spreadsheet faults caused significant damage. The awareness for these risks has risen in recent years and various new laws like the Sarbanes-Oxley Act 404 are interpreted as demanding more controls and quality assessments on spreadsheets.

It is yet unclear, though, how these risks should be addressed. One reason for this is that it is unclear which spreadsheet characteristics are desirable and which ones should be considered harmful. Research has not provided enough evidence on this yet and practitioners’ recommendations are often conflicting with each other [Kulesz, 2011].

In contrast to spreadsheets, there is more agreement about desirable and undesirable characteristics of traditional software backed up by numerous scientific studies [Beck et. al.,
Furthermore, professional software engineers are well aware of the fact that faulty software can result in significant damage and try to address this risk already in early phases of development. One popular approach to do this is by applying static analysis on source code. This can only be feasibly executed with the support of proper tools. In the following, we will refer to these tools used in non-spreadsheet software development as TSATs (Traditional Static Analysis Tools).

Spreadsheets can be seen as programming languages and products like Microsoft Excel as execution environments for them. Hence, static analysis is principally applicable to spreadsheets as well. Despite the uncertainty about desirable characteristics, a growing number of tool vendors is offering tools which allow casual end-users to run static analyses on spreadsheets. According to [Nixon and O’Hara, 2001] these mostly fully-automated tools claim to help in finding faults before they can cause any damage mostly by:

- Providing aids (i.e. a different visualization) which allow the users to better understand the spreadsheet and its internal structure
- Directly identifying potentially faulty cells (by matching them against “smell” patterns)
- Identifying unique formulas in order to allow the auditor to narrow the scope of the inspection

But is this really enough? How do spreadsheet auditing tools find potential faults? How useful are they for casual spreadsheet users? And are there areas where they can learn from the TSATs?

2. RELATED WORK

Several years ago, Nixon and O’Hara compared five spreadsheet auditing tools by running them on a single spreadsheet which was seeded with 17 faults of different types [Nixon and O’Hara, 2001]. They compared and graded the detection rates of these tools on a four-level scale. Overall, they came to the conclusion that all evaluated tools were indeed useful, although they all had problems with certain types of faults. It is notable that they treated the auditing functions built into Microsoft Excel also as a tool.

In a later study [Howard, 2007], Howard analyzed 16 tools on the market, putting them in three categories: “Auditor’s tools” (5 products), “Control & Compliance tools” (8 products) and “Automation tools” (2 products). According to Howard, sometimes the functionality provided by the five “auditor’s tools” was also contained in some of the “control & compliance tools”, making this separation a bit fuzzy. In the final conclusion, Howard provides a comparison table which lists various capabilities of each tool (i.e. detecting circular references, detecting logic errors or showing precedents) without actually rating their quality.
The study of [Powell et. al., 2007] did a complete audit of more than 50 spreadsheets with two spreadsheet auditing tools and manual inspections. Although the authors observed a high rate of false positives reported by the tools, they were convinced that this approach was still more effective than manual inspections without tool support. Another prior study by [Clermont et. al., 2002] analyzed only their own spreadsheet auditing tool, but on a selection of 78 spreadsheets from industry. The authors came to the conclusion that the tool was helpful for end-users by increasing the understanding of their spreadsheets and showing them potential faults through irregularities in the visualized patterns. In several other studies, Hermans observed similar effects when running self-developed tools on spreadsheets from the industry and interviewing end-users about the tools’ results [Hermans, 2013].

Reflecting on those studies, most of our initial questions still remained open. The two initially mentioned studies reviewed the available tools only from the perspective of “full-time spreadsheet auditors”, claiming that the tools were useful for them. But how useful are the tools for casual end-users? Ratings were given context-independent and using spreadsheets with seeded errors. The applicability of these tools to real-world spreadsheets has not been reviewed. This was done in the three latter studies which incorporated real end-users and real spreadsheets, but they only evaluated one resp. two single tools.

We found no study that compared spreadsheet auditing tools with static analysis tools for traditional software, although they share many characteristics.

3. STATIC ANALYSIS

Static analysis is comparable to the idea of destruction-free quality checks in the manufacturing industry. The analysis should only report possible defects but not alter the analysed object by any means. Research and development in the field of static analysis has been going on for years now and has produced many highly functional attack vectors to conquer issues related to product quality. In the following, we will take a short look at common approaches of static analysis:

Data-flow analysis [Taylor et al. 1980] tracks the possible paths and states the data can take. Its output is a graph which represents every change of a variable in the control flow of a program’s structure by a node. Data-flow analysis is especially good at detecting data anomalies, like lost updates or dead code.

Abstract interpretation [Cousot, 1996] tries to reduce the amount of information in the system to the basic semantics of the analysed object to be able to give answers to questions which would be undecidable otherways. For example a tool can easily detect type mismatches by abstracting from the actual value of a variable to its type.
Clone Detection [van Rysselberghe et al., 2004] searches for repeated occurrences of similar structures, marking these occurrences as clones. Such clones do not only increase the amount of information to be managed, but they also increase the chance that when changing one occurrence, other occurrences are overlooked.

Pattern matching [Beck et. al., 1999] or smell detection is the most popular approach in static analysis. For high-level programming languages like COBOL, C, C# or Java commonly used in traditional software development, feasible patterns have been empirically proven since decades. Just like a virus scanner, the tool tries to find areas which are similar to pre-defined patterns (“code smells”) and tags them with a probability representing the tool’s confidence in the finding.

4. STUDY OF SPREADSHEET AUDITING TOOLS

To approach the questions mentioned at the beginning, the first author has set up a small student research project [Berberich et. al., 2012] in an industrial context. Prior to the study, we had already established contact with a department of an industry partner (we will refer it to as DEPT) which employs around 150 technicians who work in the field of measuring and controlling emissions of industry plants. In this process, they use a dozen spreadsheet templates to capture the emission values, compute violations of allowed emission levels and produce reports about them.

The DEPT employees have used their spreadsheet templates for more than 10 years and have produced around 1500 concrete spreadsheets during this time. The templates are well-established and modifications to single templates do not occur more than twice per quarter. DEPT has an internal quality assurance process which prescribes that each single report sent out to external parties has to undergo an inspection according to the four-eyes-principle. Although DEPT have not struggled often with severe spreadsheet errors in their reports, they find it just time-consuming to inspect their spreadsheets. Their main problem was that they applied the technique “carefully inspect each cell” without any dedicated tool support and without knowing what to look for in particular.

<table>
<thead>
<tr>
<th>Sample</th>
<th># of Cells</th>
<th># of Worksheets</th>
<th># of Formulas</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2714</td>
<td>10</td>
<td>178</td>
</tr>
<tr>
<td>2</td>
<td>3236</td>
<td>10</td>
<td>184</td>
</tr>
<tr>
<td>3</td>
<td>3236</td>
<td>10</td>
<td>186</td>
</tr>
<tr>
<td>4</td>
<td>610</td>
<td>2</td>
<td>117</td>
</tr>
</tbody>
</table>
To help the industry partner in this situation, three undergraduate Software Engineering students were assigned the task to find spreadsheet auditing tools suitable for the needs of DEPT in a student research project scheduled for 4 months. As their first step, the students precisely analyzed the requirements of the industry partner and scanned the market for available spreadsheet auditing tools. They found a total of 14 tools ranging from open source research prototypes to aggressively marketed commercial products. This included tools implemented as spreadsheets themselves, Add-Ins for Microsoft Excel, standalone tools and “scanning services” in the form of SaaS (Software as a Service) on the web. For each of the tools we tried to get an evaluation license to give the students the chance to actually run the tools. Although most vendors were very interested in this study, a few vendors did not reply to our evaluation request or offered their tools only under unacceptable terms, resulting in these tools being removed from the selection. Also, the students developed a list of knockout criteria to shorten the list of tools to be analyzed in detail.

Only three tools made it to the shortlist. The students thoroughly tried them out on 11 concrete spreadsheets supplied by our industry partner (see Figure 1) and evaluated them against eight functional and non-functional requirements gathered earlier from the analysis. Then, the students presented the tools to the industry partner to check if their own findings also matched the perceptions of the industry partner. At the end, the students were able to give a clear recommendation for one of the three tools. Because one of the industry partner’s most limiting knockout criteria were the licensing costs, this was rather a “value for money” recommendation than one for the “best” tool available on the market at the time of the study.

Figure 1: Basic metrics of the spreadsheets supplied by our industry partner
5. PRACTICAL CHALLENGES

Now, three months after completion of this study, we have reflected on its results and experiences with all 14 tools involved. In the following, we share our impressions that are purely based on this reflection. Our aim is not to give concrete recommendations for specific tools but to stress widespread issues and foster a discussion about possible solution approaches.

We have the impression that most of the 14 tools do a satisfactory job in scanning spreadsheets for smell patterns like constants in formulas, references to blank cells or complex formulas. Unlike many tools for software engineers, most spreadsheet auditing tools require only small efforts for installation and the first automated inspection can be initiated easily without the help of specialists.

Many auditing features are already built into modern spreadsheet execution environments, but these features are hidden or hardly accessible for casual users. A good example is the powerful “goto -> special” functionality in Microsoft Excel. Many spreadsheet auditing tools enhance these features, i.e. by comprehensive lists of error conditions like “DIV/0” or discovering unprotected formulas.

However, we have identified several areas where we see practical challenges, since (nearly) all of the reviewed tools had significant problems providing a satisfactory experience for their users in these areas. These areas are presented in this section, ordered by our subjective perception of their severity:

5.1 Presentation of findings

The primary output of a spreadsheet auditing tool which employs fault-detection patterns is a list of findings in the spreadsheets it was instructed to scan. It is essential for this presentation to be useful and understandable to the user. Therefore, the presentation must issue a proper abstraction level (i.e. by grouping findings) and allow users to easily trace each particular finding down to the suspicious cells.

Most spreadsheet auditing tools on the market tend to produce flat lists with findings, referring to various cells. Especially when the findings are output in separate documents the navigation from the finding to the affected cell is often cumbersome. Only few tools allow users to automatically navigate to the finding without locating the cells manually. To make it worse, in many cases such lists are not ordered (i.e. by severity of the findings) or grouped (i.e. by type of the finding).

Many tools color cells or add comments to a spreadsheet, as shown exemplary in Figure 2. We experienced several problems with the approach of coloring cells:
- The “flagging” colors are rarely customizable.
- Colors which are used to show severities (i.e. red=bad, green=good) might seem intuitive, but often colors are used to represent similarities and differences. Here, they often implicitly transport hidden messages like “dark brown is more similar to red than to blue” which are not intended.
- We have seen tools which used more than 32 colors, which is way too much because humans have huge problems in distinguishing them. For instance, if the tool uses “light blue”, “navy blue” and “medium blue” across a huge spreadsheet with several worksheets, this presentation is mostly useless.

![Figure 2: Excerpt of a spreadsheet before (a) and after (b) coloring by an auditing tool](image)

In general, the use of both colors and comments for communicating findings to the user is problematic. Colors and comments are often already used as part of the original spreadsheet. Auditing tools try to address this issue by removing the original colors and comments before running the inspection. Unfortunately, there are spreadsheets which heavily rely on conditional formatting and transport their main information through colors, not numbers. Having them removed or replaced can lead to the effect of spreadsheet users not recognizing their own spreadsheets anymore and, thus, being unable to trace the findings.

Apart from that, many tools only provide a presentation which is useful for “going through each finding” but they lack a management summary that puts the number and severity of findings in relation to other metrics like the size of the spreadsheet. Some of the “control & compliance” tools do a better job here, but the views they provide often lack navigation functions which are important for traceability.

In static analysis for traditional software development there is also a lot of information (fault area, pattern matched, estimated severity) to be presented to the user, which makes an optimal presentation difficult. Commercial vendors of TSATs have invested much effort to
increase the acceptance of their reporting because they realized that only a tool which provides understandable results will sell.

5.2 Handling of false positives

Spreadsheet auditing tools are known to report high numbers of false positives. Researchers report rates between 78% [Cunha et. al., 2012] and 90% [Powell et. al., 2008]. Because the tools don’t work with specifications provided by the users, they have to make their own assumptions which can be wrong. Therefore, false positives are unavoidable. As a consequence, it is important that tools provide proper means to handle false positives. Going through a list of findings and flagging false positives is a tedious and time-consuming work, and users should not have to do it more than once - unless new findings appear. Unfortunately, almost all spreadsheet auditing tools don’t provide any means to this end - on subsequent runs, the user would get the same list of findings, including all findings he or she already flagged as false positives.

The handling of false positives is still an area to be improved in TSATs as well, although TSATs do not produce such high numbers of false positives. Usually, a thorough configuration of the tools helps reducing the amount, but help for marking them and keeping track of them over time is only inadequately implemented in most tools by providing blacklists or working with annotations within the analysed code. These approaches are not perfect and not accepted enthusiastically by the community, but they are still better than the complete lack of support for handling false positives as we observed it in most spreadsheet auditing tools.

5.3 Non-intrusiveness

Many spreadsheet auditing tools alter the original spreadsheet to be inspected during the inspection to render their reports in one of the following ways:

- Cells are colored, indicating “high risk areas” or structural similarities and differences.
- Comments are added to cells.
- Additional worksheets with reports or colored versions of the original worksheets are appended.
- The spreadsheet is enriched by parts of the tool itself (i.e. VBA functions).

Such use can lead to unwanted side-effects, especially when copied cells contain formulas referring to the original sheets. In extreme cases, this could even result in spreadsheets computing wrong values not due to faults in the spreadsheets themselves but due to faults injected by such spreadsheet auditing tools. Some tools try to remedy the problem by creating copies of the complete spreadsheet but this does not provide enough uncoupling in case formulas in these spreadsheets refer to other spreadsheets.
In TSATs this problem does not occur because all additional information is shown on an IDE overlay and is not included in the analysis object. Especially for safety-related software, the risky approach taken by many spreadsheet auditing tools is considered out of the question.

5.4 Understandability of implemented rules

Most spreadsheet auditing tools report matches with smell-patterns as “errors”. But in many cases, end-users see cells or formulas which they regard as being correct flagged as errors and often they don’t understand what the tool is trying to tell them about these cells. In fact, many of the smells are rather related to “higher qualities” like maintainability, but many tools we have seen failed to provide enough hints and explanations to justify their criticism.

TSATs have exactly the same problem. Research in this area such as [Bessey et al., 2010] shows that highly sophisticated analyses do not increase the value of a tool for the user because users tend to ignore warnings or dismiss them as false positives if they do not understand why they were raised.

5.5 Configurability and result inter-comparability

As Zitzelsberger has demonstrated [Zitzelsberger, 2012], many spreadsheet auditing tools report very different numbers of findings even for very simple patterns like “constants in formulas”. The reason for this is that many tools make hidden assumptions and exceptions, i.e. by not treating the numbers 0 and 1 as constants in formulas. In many cases, this behavior is neither configurable nor transparent. The tools only allow users to completely disable particular checks.

In contrast, most of the TSATs are highly configurable and allow to precisely define exceptions and configure the internal thresholds of their scanners. This makes them more understandable, more comparable and also helps to reduce false positives.

5.6 Lack of unification

As previously discussed, spreadsheet auditing tools try to help users with many different techniques. Therefore, it is not surprising that there is no “Swiss army knife” tool which performs great in all disciplines. Instead, specific tools are useful for specific purposes. We do not expect this situation to change in the near future. But for end-users who want to audit their spreadsheets this means that they have to deal with completely different interfaces for configuring and running the tools, as well as with completely different ways to receive the results.

Because of the many ways to detect potential problems using static analysis, there is also no TSAT which detects everything. But here, this problem is conquered by creating so-called
dashboards which aggregate the results of different tools. Figure 3 shows an example for such a dashboard (taken from “conQAT” [Deissenboeck et. al., 2008]).

5.7 Licensing costs versus risk perception

Our industry partner did not perceive faulty spreadsheets as a major risk. Although this might be explained by the phenomenon of “overconfidence” [Panko 2007], the claims in literature could not stand up to the practical experience of the industry partner having seen few and only insignificant errors in years of spreadsheet use. Therefore, DEPT sees spreadsheet auditing tools rather as a “nice-to-have” and not as a “must-have” and is not willing to spend more than 1000 € for five licenses of such a tool. Most spreadsheet auditing tools cost much more than that. For us, the only explanation for this is that these tools do not target “yet spreadsheet-risk unconvinced” organizations but rather “convinced” organizations or those that are forced (i.e. by law) to audit their spreadsheets.

The question of the cost for risk reduction is of course also an issue for traditional software development, especially in areas like software for medical devices. There is often more at stake and lower overconfidence, so the developers use every help they can get. Furthermore, there are binding standards like the [MISRA] in the automotive industry which forces the
developers to use a tool to show their compliance with the standard, while for spreadsheets there are mostly just proposals like [FAST].

5.8 Compatibility and Portability

Many spreadsheet auditing tools are tied closely to particular spreadsheet execution environments. They often realize their internal core functionality by using the APIs these execution environments provide. While this allows the tools to audit every aspect of a spreadsheet compatible with the particular execution environment, it also makes the tools less portable. As these APIs often change between major releases of spreadsheet execution environments, tool vendors are forced to issue major updates to their tools. For instance, there was a major API change between Microsoft Excel 2003 and 2007, leading to a situation where many spreadsheet auditing tools which were compatible with Excel 2003 had to be rewritten from scratch in order to provide support for Excel 2007 as well. And tools developed for Excel 2007 and above often do not provide backwards compatibility with Excel 2003 and below.

In our opinion, tool vendors should strive to develop the core components of their analysis tools decoupled from particular execution environments. They should audit the spreadsheets based on a data model of their own which was designed for the inspection of spreadsheets from the beginning and is only loosely coupled with the spreadsheets’ internal data format by using independent libraries. Therefore, the auditing core should be fully functional without the execution environment. The latter should only be used to provide a thin user interface which connects the functionality of the auditing core with the spreadsheet execution environment. As far as we know, only spreadsheet auditing tools originating from research like “Smellsheet Detective” [Cunha et. al, 2012] or “Spreadsheet Inspection Framework” [Zitzelsberger, 2012] adopt these principles.

TSATs do not run into similar problems because they only rely on formal definitions of high level programming languages. This makes them independent of APIs which are known to change more frequently than programming languages. This can be easily adopted to spreadsheets because the basic concepts of having cells connected by formulas did not significantly change since VisiCalc introduced it.

5.9 Localization

DEPT’s IT department supplies software to its employees. All software is provided with a German user interface. Therefore, DEPT’s end-users expect to get any new tool - including a spreadsheet auditing tool - with a German user interface. In general, most spreadsheet users at DEPT speak and understand English well, but many of them don’t know the English spreadsheet terminology and, thus, don’t understand findings like “spurious reference”.

ISBN : 978-1-9054045-1-3
Copyright © 2013, European Spreadsheet Risks Interest Group (www.eusprig.org)
Surprisingly, none of the tools evaluated in the study provided any localization for German environments.

6 CONCLUSION

We have seen that spreadsheet auditing tools share many commonalities with static analysis tools for traditional software development. As the spreadsheet auditing tools are relatively young, they have not yet reached the maturity of the tools used in traditional software development. Using spreadsheet auditing tools seems to be more efficient than doing pure manual inspections, but this efficiency could be greatly enhanced if their vendors would apply the lessons learned from the tools used in traditional software development. The latter ones have already developed working solutions in areas where spreadsheet auditing tools are still having a hard time although these solutions appear to be directly transferable.

ACKNOWLEDGEMENTS

This study on which this work reflects would have not been possible without the help of our three students and our industry partner. We would also like to thank all tool vendors who provided us with evaluation versions of their tools. We are also very grateful to Maarten Bessems from spreadsheetsoftware.com who has helped us a lot with his insight and his expertise on both available tools and common issues he experienced in past years when auditing complex spreadsheets in industry. Last but not least, we have to thank Jochen Ludewig, Kornelia Kuhle, Eugen Massini and the two anonymous reviewers for their helpful remarks regarding earlier versions of this paper.
REFERENCES


Mäntylä, M. (2003), “Bad smells in software—a taxonomy and an empirical study”, Helsinki University of Technology

MISRA: http://www.misra.org.uk 04.30pm 02/28/2013


Multidisciplinary Engineering Models: Methodology and Case Study in Spreadsheet Analytics

David Birch*a, Helen Liangb, Paul H J Kellya, Glen Mullineuxb, Tony Fielda, Joan Koc, Alvise Simondettic

a Imperial College London b University of Bath c Arup
* David.Birch@Imperial.ac.uk

Abstract

This paper demonstrates a methodology to help practitioners maximise the utility of complex multidisciplinary engineering models implemented as spreadsheets, an area presenting unique challenges. As motivation we investigate the expanding use of Integrated Resource Management (IRM) models which assess the sustainability of urban masterplan designs. IRM models reflect the inherent complexity of multidisciplinary sustainability analysis by integrating models from many disciplines. This complexity makes their use time-consuming and reduces their adoption.

We present a methodology and toolkit for analysing multidisciplinary engineering models implemented as spreadsheets to alleviate such problems and increase their adoption. For a given output a relevant slice of the model is extracted, visualised and analysed by computing model and interdisciplinary metrics. A sensitivity analysis of the extracted model supports engineers in their optimisation efforts. These methods expose, manage and reduce model complexity and risk whilst giving practitioners insight into multidisciplinary model composition. We report application of the methodology to several generations of an industrial IRM model and detail the insight generated, particularly considering model evolution.

1. Introduction

Many multidisciplinary engineering models are implemented as spreadsheets for ease of construction, modification and portability amongst practitioners. While many benefits are realised by an integrated spreadsheet based model, there are some inherent difficulties common to many engineering models. To demonstrate these challenges, we consider those within the urban masterplanning community.

Urban masterplanning is the process of creating a coherent design for the development of a campus, suburb, city or region. It spans not only architecture but the disciplines involved in the implementation of changes to the built environment such as acoustics and water supply.

Figure 1 Conceptual model of an Integrated Resource Management (IRM) Model [Ayaz08]. Sustainability models from many disciplines are integrated to form a coherent model for assessing urban masterplans [Page08].

Increasing requirements for managing environmental impact have led to demand for interdisciplinary modeling of sustainability metrics such as annual per capita carbon emissions in order to benchmark and improve designs. These drivers have been unified by Integrated Resource Management (IRM) models [Kepran, 2002; Ayaz, 2008; Page, 2008] which integrate models from each discipline into a coherent assessment tool. The challenges encountered in such models
motivate this work and are discussed in the next section. This paper presents the following contributions to address these issues.

- We present a methodology and tool suite for systematic, automatic analysis of large spreadsheet-based models with novel metrics to assess internal communication and integrated sensitivity analysis to aid practitioners in optimisation.
- We apply this methodology with a focus upon multidisciplinary engineering assessment models, a model type not widely studied within literature.
- We demonstrate the methodology's application through practical case studies with an industrial multidisciplinary sustainability model, identifying insight for practitioners and study model evolution over three model generations.

2. Motivation

In this paper we consider Arup's IRM model [Ayaz, 2008; Page, 2008] as an example of a complex spreadsheet based interdisciplinary engineering model. Arup is a global engineering consultancy and their IRM model is used frequently worldwide on a wide-range of projects. We now consider some of the challenges inherent to all such interdisciplinary engineering models.

As shown in Figure 2, Arup's IRM model consists of several different discipline specific sub-models including energy demand, energy supply, passenger transport and land-use. Each discipline has a data input model and an output model which calculates sustainability metrics such as annual energy demand. These input/output model pairs strongly rely, not only, upon each other, but also upon the other disciplines' input and output models. For example, the energy supply sub-model uses inputs from the land-use input sub-model and the outputs of the energy demand model.

This creates a complex interrelated web of models which reflects the physical complexity of sustainability concerns. In the centre of this web is a project-specific sustainability dashboard which calculates summary metrics using information from all disciplines' input and output models. This complexity is a requirement of faithful modeling and is a common feature of many engineering analysis models. This class of models, in contrast to more traditional spreadsheet based models such as tax calculators, face particular challenges:

- **Model Complexity** - Such models are by their nature complex due to the strong coupling between already intricate discipline models which must become facsimiles of real life complexity. This leads to difficulty in gaining an accurate overview of the whole model and to
understanding how a single discipline's model functions; especially outside of a practitioner’s area of expertise.

- **Data Requirements** - Due to this complexity, engineering models frequently contain large data requirements. Our analysis identified 933 (see Section 10) separate design or analysis variables required for the carbon calculation of Arup's IRM model, ranging from the total floor area of residential buildings to the CO2e emissions for disposing of electronic equipment. The time taken to gather, process and enter the required information is a major cost in applying such models.

- **Implicit Knowledge** - Such engineering knowledge is difficult to formalise, being built up as an informal set of good practice over time. Formalising and modifying this implicit knowledge is challenging, particularly when there is limited documentation (e.g. spreadsheet formulas).

- **Interdisciplinary Communication** - Within multidisciplinary models, each discipline has its own nomenclature which must be communicated to the other disciplines involved. Given the limited documentation in many spreadsheets, this may result in the same figure to be included under different names in different units.

- **Project Adaptation** - In contrast with many fixed purpose models implementing a clear specification (e.g. tax law), most engineering models, whilst trying to be as general as possible, often require some tweaking to fit the exact nature of the task at hand. Due to its scope, an IRM model often requires adaptation to each project for the following reasons:
  - **Models too broad** - A model's data requirements are large and can prove broader than the scope of the project, especially during early design stages. This leads to difficulty in fulfilling all the data requirements.
  - **Models too narrow** - A common cause of model adaptation is to meet project specific concerns. For example the inclusion of irrigation and grey water recycling is critical in water stressed areas but is rarer in more temperate climates and so may need to be added to the model.
  - **Cause and Effect unclear** - Project adaptation for these reasons is a difficult activity - the scale of the model makes identification of cause and effect between an input to be modified and the final sustainability metric difficult to determine, especially because of the interrelated nature of multidisciplinary models.

- **Difficulty of Optimisation** - Once an engineering model is applied to a project the most common use is to create a number of design improvement recommendations. This is difficult since it depends on understanding both the overview and the detail of the model. This requires high levels of implicit knowledge in varying assumptions and understanding the flow of cause and effect across multiple discipline models to identify the handful of most advantageous steps that could be taken to improve the design.

- **Implementation** - Whilst spreadsheet based models are common and support ease of use and modification (a survey undertaken by the authors identified around 1,000 engineering analysis models in use within a large engineering firm). There is a growing body of evidence that spreadsheet models in common with other large software products are likely to contain errors at unacceptable rates. A good summary of the current evidence is available in [Panko, 2008].

In summary, there are clear obstacles in the use of all spreadsheet based multidisciplinary engineering models. This paper demonstrates the value of model analysis tools to support practitioners in their information intensive tasks.
3. Methodology

As proposed in [Liang, 2011] with application to the design process, we propose and demonstrate an Extraction and Analysis Methodology (EAM) consisting of a series of techniques to help expose, reduce and manage model complexity. In this paper we explore the impact on multi-disciplinary engineering models. We demonstrate insight into multidisciplinary model composition and show value for designers in quickly focusing efforts into optimisation.

The methodology has the following steps:

1. **Obtain** - Model and project objectives.
2. **Define** - Key Performance Indicators (KPIs) of interest to the project.
3. **Extract** - Slice model to expose and reduce complexity to produce a smaller model computing only the KPIs of interest.
4. **Analyse - Visualise** - Visualise model to aid comprehension and show cause and effect.
5. **Analyse - Metrics** - Compute metrics on calculation model to give insight into model composition.
6. **Optimise** - Set variable ranges to formalise implicit knowledge enabling sensitivity analysis to give insight and focus optimisation effort.

The benefits of this methodology are in the value they provide to the practitioner. Firstly, by reducing the problem size and allowing visualisation to enable interactive exploration of cause and effect. Secondly, by providing metrics and insight into the multidisciplinary composition of the model we show the interaction of various disciplines. Finally, a sensitivity analysis provides further insight and focuses design effort enabling faster optimisation. The methodology also aids model development and evolution as the models are adapted to new projects. Similarly it helps mitigate the risks associated with spreadsheet based modelling particularly during the modification and optimisation stages of use.

4. Related Work

Studies have identified the presence [Panko, 2008; Clermont, 2005] and frequency [Blayney, 2006] of spreadsheet errors. We know that the majority of modellers do not have formal training in spreadsheet based modelling [Panko, 2008]. A body of literature has developed aiming to formalise a taxonomy of spreadsheet modelling bugs [Panko, 2010]. The risks of these errors are commonly underestimated and few users of spreadsheets consider the risks of such errors [Blayney, 2006]. Indeed very few practitioners consider that they need tools for debugging their models. There have been a number of studies into auditing tools for spreadsheets (e.g. [Blayney, 2006] for tax purposes). Historically there has been much interest in deriving visualisations based on the calculation graph of a spreadsheet [Kankuzi, 2008; Shiozawa, 1999]. Several visualisation tools have been proposed to avoid costly errors.

The novelty of our approach is that rather than treating a spreadsheet as simply a software artefact we consider the insight each step and tool in our methodology can generate for the model maintainer with a view to aiding them as they optimise a design. This is particularly a challenge for engineering models as oppose to financial models which have previously been the focus of research. These engineering models through their constant evolution and adaptation to projects present new research challenges. Particularly we propose a life-cycle methodology for the use of such tools by practitioners. We also consider for the first time, the challenges that a multidisciplinary model brings to the challenge of spreadsheet engineering. For example, considering approaches for assessing multidisciplinary communication within models (Sections 7 and 8). We also consider how sensitivity analysis may be performed in large spreadsheet based models. This is enabled through our extraction and analysis methodology and has the potential to generate substantial insight for practitioners as evidenced in Section 10. Finally, we consider the evolution of complex models as they are developed.
and applied to projects. As discussed in Section 11 this is a great source of insight into the model and a future research challenge.

5. Model Extraction

The first stage of the methodology is to extract a slice of the model from Excel. Slicing a model or computer program is a well-known technique [Weiser, 1981] that allows consideration of only the portion of the model involved. In this context, slicing extracts only spreadsheet cells involved in the calculation of particular outputs, reducing the model size and complexity.

We recursively extract cells by starting from the outputs of interest (e.g. annual per capita carbon emissions), read their formula parsing them for references to other cells, recursively extracting these until no more cells are referenced. We used a mathematical expression evaluation library NCalc and modified the grammar to be compatible with Microsoft Excel formulas and implemented a subset of Excel functions allowing internal evaluation of formulas to enable validation of analysis. In contrast with many other approaches [Reichwein, 1999; Shiozawa, 1999; Kankuzi, 2008] this formula parsing approach enables us to gain insight within formulas, for example differentiating cells referenced from arithmetic from table lookup functions which reference hundreds of cells. This enables simplification of the extracted model slice and resultant graph of cells. We also extract cell values and names to aid comprehension of visualisations, metrics and sensitivity analysis. We find model slicing a key contribution to the comprehensibility of the resulting calculation graph as is shown in the next section.

6. Visualisation

Taking inspiration from [Shiozawa, 1999; Kankuzi, 2008; Hermans, 2011], our methodology includes a calculation graph visualisation. We present cells and ranges as nodes in the graph and references between them as edges. We colour the nodes according to which discipline model they originate from, giving insight to discipline communication. Additionally, we support interactive exploration of the calculation graph under several layouts each highlighting different aspects of the graph.

For example Figure 3 highlights the complexity of CO2e emissions per capita per annum for external transport calculation within Arup's IRM model. This model slice contains 255 cells and is visualised with a linlog energy force model highlighting strongly connected sub-graphs. Hence we see ten sub-graphs (calculations) feeding into the metric (the central node in the graph), corresponding to the calculation of carbon emissions for ten modes of transport.
An interesting graph anomaly is that two sub-calculations are connected ("A" and "B"). Upon selecting node "C" linking both calculation clusters ("A" and "B"), a list of the ways this cell is used in the calculation of the metric is generated. Further investigation shows the input value ("C") to represent the CO$_2$e emissions for diesel buses per passenger kilometre. This is used in the calculation of both the bus and coach modes of transport ("A" and "B"). This is unexpected since coaches are normally have around a quarter of the CO$_2$e emissions of buses. This implies the carbon emissions for coaches could be overestimated in the model. This issue was reported to the IRM engineers who agreed the issue was unexpected and had been fixed in later versions of the model but could have been an assumption carried over from a previous project where coaches and buses have similar CO$_2$e emissions on small islands.

This demonstrates the utility of slicing and visualisation tools to aid understanding and examination of complex engineering models.

7. Model Metrics

Having extracted a slice of a multidisciplinary engineering model various graph metrics can be automatically calculated to give insight into the multidisciplinary composition of the calculation model. In contrast with previous approaches we consider the value of worksheet level metrics rather than formula level metrics [Hodnigg, 2008; Hermans, 2012], particularly because of the relationship with the disciplines they represent.

Firstly we can partition the calculation graph by discipline and gain a measure of their complexity via the cell count and number of inputs in their partition. This is shown in Figure 4 which also shows the average valency (average number of cells each cell references and is referenced by). More references show more complexity and interconnectivity which although harder to maintain, may model reality more accurately. Arup's IRM model's carbon calculation has 2,357 nodes with average valency of 2.89. In Figure 4 we see the model's focus upon Energy and Passenger Transport with the Transport input models and the Energy output models containing most complexity and interconnectivity.

<table>
<thead>
<tr>
<th>Discipline Model</th>
<th>Cell Counts</th>
<th>Inputs</th>
<th>% Inputs</th>
<th>Average Valency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Land Use (LU)</td>
<td>38</td>
<td>24</td>
<td>63%</td>
<td>3.24</td>
</tr>
<tr>
<td>Socio Economic (SE)</td>
<td>38</td>
<td>23</td>
<td>61%</td>
<td>1.87</td>
</tr>
<tr>
<td>Passenger Trans (PT)</td>
<td>210</td>
<td>180</td>
<td>86%</td>
<td>1.57</td>
</tr>
<tr>
<td>Pass Trans Coeff (PTCo)</td>
<td>140</td>
<td>99</td>
<td>71%</td>
<td>2.44</td>
</tr>
<tr>
<td>Energy Demands (ED)</td>
<td>477</td>
<td>371</td>
<td>78%</td>
<td>1.89</td>
</tr>
<tr>
<td>Logistics (Lo)</td>
<td>133</td>
<td>111</td>
<td>83%</td>
<td>1.33</td>
</tr>
<tr>
<td>Logistics Coeff (LoCo)</td>
<td>16</td>
<td>16</td>
<td>100%</td>
<td>2.75</td>
</tr>
<tr>
<td>Water (Wa)</td>
<td>111</td>
<td>111</td>
<td>100%</td>
<td>1.00</td>
</tr>
<tr>
<td>Energy Supply (ES)</td>
<td>34</td>
<td>33</td>
<td>97%</td>
<td>1.79</td>
</tr>
<tr>
<td>Energy Sup Coeff (ESCo)</td>
<td>12</td>
<td>12</td>
<td>100%</td>
<td>6.00</td>
</tr>
<tr>
<td>Convert Factors (CF)</td>
<td>2</td>
<td>2</td>
<td>100%</td>
<td>18.00</td>
</tr>
<tr>
<td>Out: Energy Dem (SSEd)</td>
<td>185</td>
<td>12</td>
<td>6%</td>
<td>3.32</td>
</tr>
<tr>
<td>Out: Energy Sup (SSES)</td>
<td>244</td>
<td>48</td>
<td>20%</td>
<td>4.40</td>
</tr>
<tr>
<td>Out: Logistics (SSLo)</td>
<td>67</td>
<td>0</td>
<td>0%</td>
<td>3.99</td>
</tr>
<tr>
<td>Out: Pass Trans (SSPT)</td>
<td>366</td>
<td>0</td>
<td>0%</td>
<td>3.71</td>
</tr>
<tr>
<td>Out: Sodo-Econ (SSSE)</td>
<td>14</td>
<td>0</td>
<td>0%</td>
<td>4.21</td>
</tr>
<tr>
<td>Out: Water (SSW)</td>
<td>264</td>
<td>75</td>
<td>28%</td>
<td>4.08</td>
</tr>
<tr>
<td>Project Outputs (Out)</td>
<td>6</td>
<td>0</td>
<td>0%</td>
<td>14.83</td>
</tr>
</tbody>
</table>

From the number of inputs in each model we gain an indication of each discipline’s data demands. Finally we see that although each discipline has both an input and an output model, this demarcation is not strictly observed in all disciplines. The inputs within output models are of particular concern; though these are sometimes conversion factors or calculation options. Similarly many input models have up to 40% non-input (i.e.
calculation) cells. This is acceptable since they summarise the input data for use in other models (e.g. total land use).

Together with metrics for the most referenced input data and sub-calculations, these multidisciplinary metrics give a key overview of the model focus as well as aiding the maintenance of the model by checking whether design rules are followed. This is particularly important in engineering models where model structure is constantly evolved by practitioners.

8. Discipline Coupling

Since multidisciplinary models contain sub-models from many different disciplines, we consider the interconnections between these disciplines as shown by data dependencies in spreadsheet formulas.

As a concrete example, one hypothesis proposed by the IRM engineers was that the transport model was not connected to the land-use model (since it uses software external to the spreadsheet). In order to test this, a discipline coupling matrix was created (Figure 5). This is calculated by considering all edges in the calculation graph and entering them into the matrix according to which disciplines they are from/to (effectively recording cross worksheet reference in formulas). We see the passenger transport and logistics models (PT, Lo and their Coefficients) are indeed not directly connected to the land-use (LU) model, thus confirming the IRM engineers' hypothesis.

Due to the breakdown of inputs/output models within disciplines we see that the top right quadrant covers output models reading from input models. The bottom left quadrant covers input models reading from output models (which shouldn't and doesn't occur). Much of the model complexity is found in the bottom right quadrant with interconnected calculation models. The diagonal shows sub-model complexity via internal references. We also consider indirect references (reference via another model) and note the primacy of the energy demand and supply models which reference almost all other disciplines.

This is a key technique for considering multidisciplinary engineering models and enables validation that the spreadsheet created matches a conceptual model of communication dataflow.
## 9. Coupling Metrics

Given the number of sub-models comprising the IRM model, there is much similarity between large spreadsheets and large software programs. Considering each discipline’s models as separate code packages we can apply standard software engineering code metrics [Martin, 2006] to the discipline coupling matrix. Such metrics are in a similar vein to [Hermans, 2012] but are calculated at the worksheet level by treating worksheets as packages (inspired by their relationship to disciplines).

<table>
<thead>
<tr>
<th>Afferent Coupling (Responsibility)</th>
<th>Efferent Coupling (Independence)</th>
<th>Instability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Land Use</td>
<td>4</td>
<td>0%</td>
</tr>
<tr>
<td>Socio Economic</td>
<td>2</td>
<td>33%</td>
</tr>
<tr>
<td>Passenger Trans</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Pass Trans Coeff</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Energy Demands</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Logistics</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Logistics Coeff</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Water</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Energy Supply</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Energy Sup Coeff</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>Convert Factors</td>
<td>2</td>
<td>0%</td>
</tr>
<tr>
<td>Out: Energy Dem</td>
<td>2</td>
<td>75%</td>
</tr>
<tr>
<td>Out: Energy Sup</td>
<td>2</td>
<td>60%</td>
</tr>
<tr>
<td>Out: Logistics</td>
<td>1</td>
<td>75%</td>
</tr>
<tr>
<td>Out: Pass Trans</td>
<td>2</td>
<td>50%</td>
</tr>
<tr>
<td>Out: Socio-Econ</td>
<td>2</td>
<td>50%</td>
</tr>
<tr>
<td>Out: Water</td>
<td>1</td>
<td>83%</td>
</tr>
<tr>
<td>Project Outputs</td>
<td>0</td>
<td>100%</td>
</tr>
</tbody>
</table>

Figure 6: Software engineering metrics normally applied to large software projects [Martin06] are applied to multidisciplinary models to gain insight into model maintainability and stability to change.

We calculate measures of a model's responsibility to and independence from other models in terms of the data they provide and consume from other models. We use these to compute instability to model change to identify which models are most likely to cause difficulty for project adaptation.

Firstly, we compute a model's afferent coupling [Martin, 2006] by counting the number of discipline models (worksheets) which reference cells in the given model (worksheet). This gives a measure of the responsibility of a model to other models.

Models with high afferent coupling are less easy to adapt to new projects as changes must avoid breaking its dependant’s expectations.

Secondly, we compute efferent coupling [Martin, 2006] by counting the number of models (worksheets) which cells in a given model (worksheet) reference. This gives a measure of the independence of the model, with lower scores considered more independent. Models with poor independence are likely to be affected by changes in other models.

Finally we compute a measure of a model's instability to change [Martin, 2006] as follows, where 0% is stable and 100% is unstable.

\[
\text{Instability} = \frac{\text{efferent}}{\text{afferent} + \text{efferent}}
\]

Figure 6 shows the results for the IRM model. As expected most discipline input models are highly independent and not likely to be affected by changes to other models. Conversely the output models have varying levels of dependence on other models and so have higher levels of instability. This indicates they are more likely to be affected by model changes, particularly as the model evolves. Instability also coarsely identifies flows of effects from changes in input.

These metrics allow engineers to quantify the difficulty and risks involved in making changes to a given model and how likely these changes are to affect other disciplines' models; frequent reference to
these metrics should create more modular model which are less costly to adapt, in part by identifying erroneous or poorly planned connections between worksheets.

10. Sensitivity Analysis

One common engineering task is to optimise a design for a given KPI, for example annual per capita carbon emissions. This is difficult since the designer must identify all input cells which affect the KPI, consider their ranges and then attempt to find combinations of values which optimise the KPI whilst considering the impacts of doing so.

In support of this we created tools to apply a sensitivity analysis to a slice of the spreadsheet corresponding to all of the cells involved in calculating a KPI of interest. A sensitivity analysis identifies the input factors to which the KPI is most sensitive to changes in. This enables the designer to focus upon the subset of inputs which have the most effect on the KPI, increasing their productivity.

In contrast to many tools we use Design of Experiments techniques to create efficient experiments for interrogating the sensitivity of a model. These techniques take a set of factors (inputs), which affect the output of interest, along with the maximum and minimum value each factor can take (set by the practitioner). A series of model runs is then constructed with varying combinations of factors set at their maximum or minimum levels. These are then run and the results analysed. We use a Plackett-Burman (PB) sensitivity analysis [Plackett, 1946] due to its computational efficiency, which comes at the cost of insight only into the effects of factors and not their interactions.

As an example we consider the annual per capita carbon emissions KPI, extract the corresponding model slice and identify its numeric inputs. For each input the maximum and minimum range of the variable is established with engineers from the appropriate discipline. Note that not all numeric inputs are variable e.g. conversion factors. This produced 933 parameters for a sensitivity analysis; Figure 7. shows the results. This requires 2,563 simulation runs, the Excel-Sensitivity tool runs one run per 0.72 seconds on a Quad Core (Intel i7 720QM) machine, running four experiments concurrently.

Since we can test the sensitivity of more than one KPI to the same set of factors at very little extra cost, we explore side effects on the breakdown of the total per capita CO₂e emissions. This gives insight into the relative importance of each sub-metric to the total and what scope there is for affecting each. For example, different fuel type metrics affect the total CO₂e emissions and the transport KPIs but do not affect the non-domestic buildings sub-metric. Interestingly, we see that district heating has a surprisingly high effect on the carbon efficiency, as do Combined Heat and Power (CHP) systems.
should they be included in the masterplan. These results can be broken down by discipline for more detailed insight.

In conclusion we identify these benefits of a sensitivity analysis on an engineering model:

- **Design Insight** - The designer gains knowledge of the design space, the interactions between the design parameters and the output KPIs of interest allowing a focusing of effort upon only those variables the KPIs is most sensitive to and similarly gaining insight into side effects of changes on other KPIs.

- **Design Space Exploration** - Whilst running the analysis we automatically create and evaluate several thousand designs. Exploration of these allows designers to quickly understand potential configurations and directions for design improvements.

- **Identification of effects of assumptions** - Within most engineering models there are a large number of calculation assumptions. For example, the carbon emissions of buses per passenger kilometre. If included within a sensitivity analysis (the max/min values determining the confidence interval of the assumption) the engineer gains understanding of the relative importance of the assumptions and the respective effects of error margins; enabling focus on refining model uncertainty which will have most impact. This also mitigates risks in analysis accuracy since unexpectedly sensitive inputs are identified.

11. **IRM Evolution**

One interesting use of EAM is its repeated application to a model, particularly as it is adapted to meet the requirements of new projects. We explored the application of the EAM toolkit with three IRM models developed over a number of years from a concept case study to a globally used tool; demonstrating the transferability and scalability of the methodology and tools.

Figure 8 shows such application of EAM to Arup’s IRM model. The size of the model has increased dramatically as more detail and accuracy have been added to the model. This is partly due to the most recent IRM model containing data tables localised to geographical regions. The increase in size also reflects an increase in complexity, as noted by the number of Excel functions called within the model. All figures in the table refer to the slice of the model corresponding to annual per capita carbon emissions. The complexity increase compounds the problems discussed in Section 2, highlighting the need for computational support.

From computing the metrics discussed in Sections 7-9 for each model, we identified the change in focus over time from water modelling through to energy and carbon models by considering the change in complexity and connectivity between the disciplines within the model. These insights demonstrate transferability of the approach. We have been able to apply the process and tools to an Arup vertical transportation model from start to end within one working day reporting valuable insight into the model which was accepted by its expert maintainer.
12.  Further Work

Firstly, we see that high data demands are a barrier to IRM adoption. However, it may be possible for automatic or semi-automatic methods to be applied to the calculation graph to attempt to produce an abstracted version of the model with fewer data requirements. Methods such as sensitivity analysis could be used to identify parts of the model for removal which have limited impact upon the accuracy of the overall model results.

Secondly, whilst a PB sensitivity analysis gives good insight into the model, other forms of sensitivity analysis might be applied (dependent upon their tractability). One of the more interesting methods would be to apply automatic differentiation tools to the overall model formula allowing accurate insight into the multi-variate sensitivities of the model.

Finally, given the formalisation of discipline specific implicit knowledge behind the variable ranges for a sensitivity analysis, it would be interesting to use these as the constraints to an optimisation engine performing constraint based optimisation upon the model. Of course such optimisation would never replace an engineer's insight into which combinations of variable values are practicable but could serve as a valuable decision support tool within IRM models and other engineering models. Particularly the authors see this work and its methodology as being applicable to spreadsheet systems in other domains (e.g. financial) and particularly groups of interlinked spreadsheets.

13.  Conclusions

The case study presented demonstrates the need and the value of computational tools in understanding complex multidisciplinary models. The techniques explored aid practitioners in model comprehension, optimisation and evolution; as evidenced by exploring Arup’s IRM model as a representative model and the aid given to practitioners. Many of whom have no formal programming experience with model development tasks. Model slicing allows reduction of model complexity to show only the salient points. Interactive exploration of the model as a calculation graph valuably enables users to build a mental model of how the calculation works. Model metrics are an interesting and valuable way of gaining detailed insight into the model and its composition. Metrics pertaining to the multidisciplinary nature of the model give higher level insight into interdisciplinary communication. Finally, sensitivity analysis is a valuable technique for understanding the relative importance of hundreds of input variables when seeking to optimise for a given KPI or checking model assumptions. Repeated application of EAM to a model clearly identifies changes in model composition and focus. EAM and its techniques can be applied more widely than IRM models and have been applied to other confidential engineering models. Indeed the approach has been applied to a system of some 500 connected spreadsheet files.

In conclusion:

- We present a methodology and tool suite for systematic, automatic analysis of large spreadsheet-based models with novel metrics to assess internal communication and integrated sensitivity analysis to aid practitioners in optimisation.

- We applied this methodology with a focus upon multidisciplinary engineering assessment models, a model type not widely studied within literature.

- We demonstrated the methodology's application through practical case studies with an industrial multidisciplinary sustainability model, identifying insight for practitioners and model evolution over three generations.

The authors would like to gratefully acknowledge the support of the EPSRC and Arup in funding and supporting this work.
14. References


The Misuse of Spreadsheets in the Nuclear Fuel Industry: The Falsification of Safety Critical Data using Spreadsheets at British Nuclear Fuels Limited (BNFL)

Simon Thorne
Cardiff Metropolitan University
SThorneseCardiffMet.ac.uk

ABSTRACT

This paper considers the management, technological and human factor issues that led to the BNFL fuel rod spreadsheet data falsification incident in 1999. BNFL discovered in 1999 that some data supporting quality assurance and safety processes had been falsified by BNFL workers using spreadsheets. The implication of this finding was that some of the Mixed Oxide Fuel Pellets shipped to customers in Japan for use in nuclear reactors were of an unknown mass and quality. This meant that the use of the MOX pellets fuel produced by BNFL would introduce uncontrolled factors into the safe operation of nuclear reactors. This paper will examine the production of MOX pellets at the Sellafield site, the falsification of data and the report commissioned by HM Nuclear Inspectorate. The paper will then identify a number of managerial and technological failings that led BNFL to use spreadsheets for recording such data. Finally the paper analyses other cases of spreadsheet fraud and explores some strategies for reducing the likelihood and impact of spreadsheet errors and fraud.

1. INTRODUCTION

This paper examines the BNFL data falsification incident in detail and provides some analysis on the context of the failure at the MOX Demonstration Facility (MDF). The paper starts by explaining the nuclear fuel industry and the technical processes of manufacturing nuclear fuel. We then examine BNFL as a company and the production of Mixed Oxide (MOX) fuel pellets at MDF. The data falsification incident is then examined using the Nuclear Installations Inspectorate (NII) report and other sources. The final part of the paper analyses this report and discusses the failings of BNFL management in terms...
of human factors and ergonomics, management practices and technology provision for the MDF facility. The final paper considers what measures could have safeguarded against such a fraud and how these might be practically implemented.

1.1 Nuclear Fuel Industries

The nuclear fuel cycle is a process by which electricity is derived from radioactive materials such as Uranium 235 or Plutonium 239 through controlled nuclear reactions. Nuclear fuels are manufactured to allow precise nuclear reactions under controlled conditions. In a nuclear reactor, such as a light water reactor, fuel rods are placed inside a fuel assembly and inserted into a water filled reaction chamber. The nuclear fuel interacts, giving off ‘slow neutrons’ also known as thermal neutrons. The energy emitted by thermal neutrons is used to heat water, producing steam which in turn drives turbines and produces electricity. The water in the reactor is used both as a coolant and as steam; the means of driving the turbines. This is the most common type of nuclear reactor and is known as a ‘light water reactor’.

The composition of the fuel used in light water reactors can either be uranium based or a mix of plutonium and uranium. Uranium is mined, enriched and manufactured to a specification and then formed into uranium fuel pellets. Depending on the type of reactor, uranium may be mixed with Plutonium 239 which is weapons grade nuclear material to form Mixed Oxide Fuel (MOX).

Before the non-nuclear proliferation treaty of 1970, governments around the world experimented with nuclear weapons and hence needed significant amounts of plutonium. Plutonium is a natural by product of nuclear reactors, but highly enriched weapons grade plutonium is obtained through certain specifications of nuclear reactor. Hence many governments built nuclear reactors that would yield weapons grade plutonium for research. Due to the non-nuclear proliferation treaty, there is now less need for weapons grade plutonium and the focus now rests on how to manage the stockpiles of plutonium now accumulated. This is a non-trivial problem for governments around the world. In 2005 the Parliamentary Office of Science and Technology (POST) reported that the UK has 70 tonnes of civilian plutonium which was predicted to rise by another 10 tonnes by 2015 (POST, 2005). In addition to the danger of nuclear proliferation, plutonium is also difficult and dangerous to store. Not only does plutonium give off radiation which has proven links to cancer but there is also a risk of high energy reactions if the plutonium is placed under certain conditions. These high energy reactions are the same reactions present in nuclear explosions and the operation of nuclear reactors. If the material is not stored and cooled appropriately there is risk of fuel catching fire or melting through thermal neutron interaction. The consequences of plutonium fires are potentially severe since highly dangerous particles can be spread over wide areas transported in smoke. In addition, plutonium 235 has a radioactive ‘half-life’ of 24,100 years meaning that it will not be safe to handle for tens of thousands of years. Hence burying plutonium in concrete
tombs in the ground until it is safe is not viable since the concrete would degrade before
the plutonium, meaning that plutonium could leak into the water table and severely
contaminate the earth.

One option for managing plutonium stockpiles is mixing plutonium and uranium to
produce MOX fuel. The MOX fuel can then be used in civilian nuclear power plants and
helps reduce the amount of plutonium held by the government. However, MOX fuel is a
dangerous substance since it contains weapons grade plutonium and attracts the highest
levels of security. Some suggest (Morichaud, 1997) that the production, specifically
separating plutonium and uranium, and the distribution of MOX fuel is actually worsening
the risk of nuclear proliferation. The major risks being the theft of separated plutonium
and the risk of theft or accident in the transport of MOX fuel from production facility to
reactor.

There are several nations that produce MOX fuel for international trade including France,
Russia, Japan and the United Kingdom. The UK had the first civilian nuclear power
station in the world; Calder Hall at the Sellafield site in Cumbria. The Sellafield site has
since been the centre of the UK nuclear power and fuel processing activities. British
Nuclear Fuels Limited (BNFL) was the organisation responsible for producing electricity
and nuclear fuels in the UK.

1.2 BNFL

British Nuclear Fuels was founded in 1971 as British Nuclear Fuels Limited and was
reorganised and renamed several times until the company was dissolved in 2010 by the
coalition government (BBC, 2011). The activities of BNFL include: generating electricity
via nuclear reactions, decommissioning nuclear power plants, nuclear waste management
and in 1988, the production of Mixed Oxide (MOX) fuel pellets for international sale.
The MOX pellets and fuel assemblies (complete nuclear fuel rods) were manufactured in
the MOX Demonstration Facility (MDF) at Sellafield in Cumbria. The MDF was a small
scale facility designed to produce modest amounts of nuclear fuel for a variety of
customers according to their design. The MDF produced MOX pellets and fuel
assemblies for nuclear power stations in Germany, Switzerland and Japan. In 1997 BNFL
began to manufacture MOX fuel pellets for use in Japanese nuclear power stations.

1.2.1 Production of MOX fuel pellets at MDF

MOX fuel pellets are metal coated cylindrical pellets each weighing approximately 30
grams and standing 2cm tall. The pellets are placed inside an empty zirconium alloy tube
which is known as a fuel rod. The fuel rods can be bundled together in a matrix
framework; these bundles are known as fuel assemblies. The mass and shape of each
pellet varies depending on the specification of reactor the fuel is to be used in, and small
variations in mass and density occur through the manufacturing process. Most MOX fuel is used in light water reactors which use pure water to cool the reactor and to moderate fast neutrons.

MOX fuel pellets are produced by milling assayed quantities of Uranium and Plutonium dioxide powders with some other non radiological components. This forms a MOX powder which is then tumbled until MOX granules form. The MOX granules are sintered (baked in a furnace below melting point) and compressed which produces ceramic MOX pellets. Next, the MOX pellets are carefully ground down to the specification of the customer. The pellets are measured twice by an automated laser micrometer to give the precise mass of each pellet. As a fail safe quality measure, a sample of pellets is measured again using a micrometer by a human engineer who records and compares the measurements. Once the precise mass of a batch of MOX pellets is known, the pellets can then be inserted into empty fuel rods and depending on customer demands, a fuel rod assembly (a bundle of fuel rods).

Accuracy in measurement of MOX pellets is paramount since the exact mass of each pellet varies. In order to have a stable and controllable nuclear reaction, one must know the precise mass of radiological material contained in a nuclear fuel rod or assembly. If there is too much radiological mass, there is a risk that a nuclear reaction could not be slowed sufficiently by using “control rods”. Control rods are made of a material, such as graphite, that absorbs neutrons emitted in nuclear reactions and therefore control the fission reactivity of the fuel rods. When a nuclear reaction needs to be slowed or stopped, control rods are inserted between the fuel rods to absorb the neutrons and prevent the neutrons interacting. As the interaction of neutrons reduces, the amount of thermal energy emitted by the fuel rods reduces which in turn cools the core of the reactor. However, if there is more radiological mass in the fuel rods than has been accounted for, there will not be enough mass in the control rods to absorb all of the neutrons given off by the fuel rods. This means that the neutrons will continue to interact even with the control rods in place. This may result in unexpected heating of the fuel rod cores, which if not sufficiently cooled could breach and exceed the melting point of the fuel rod. This is known informally as a core melt down.

In a core melt down, the pellets and the fuel rod casings reach temperatures in excess of 2,600 F resulting in the complete melting of the fuel rod cladding and pellets. The molten fuel rod material then flows out of the core and into the containment vessel. The containment vessel is a thick concrete pressurised structure enclosing the core. The containment vessels primary purpose is to keep radioactive material inside the reactor and separate from the wider environment in case of a melt down. When the core melts and flows into the containment vessel, it will come into contact with water (used as a coolant) sitting at the bottom of the vessel. This water will rapidly heat and evaporate into steam and there is a danger that the pressure inside the vessel will cause a steam explosion and rupture the containment vessel releasing radiological contamination into the environment.
There is also a risk that the zirconium alloy used in the fuel rod casing will oxidise in the steam and form hydrogen gas. There is a risk that a build up of hydrogen gas will explode inside the vessel, or as the gas is vented in an emergency. This explosion could rupture the containment vessel and release radiological material into the environment. Indeed, it was steam and hydrogen gas explosions that ruptured the containment vessels of MOX reactors at the Fukushima nuclear plant in 2011 (Biello, 2011) although the root cause for this was an earthquake. Even if the vessel remains intact, there is the risk that the molten fuel will continue to react and melt through the thick concrete lining of the containment vessel and release radiological material into the environment. One only needs to examine the disaster at Chernobyl to know how damaging such a release of material is to the environment. Hence engineering nuclear reactions is a precise science and therefore any unknown variables, such as additional radiological mass, are highly undesirable and can result in severe consequences.

2. DATA FALSIFICATION AT MDF

On the 20th of August 1999 a quality control team member noticed similarities between successive lots of secondary quality control data. The secondary quality control data is produced via manual measurement of MOX pellets using a micrometer and a spreadsheet to record the data. On the 10th of September 1999 BNFL notified Her Majesties Nuclear Installations Inspectorate (NII) who launched an investigation into the falsification of safety data. Most of the factual information in this paper is drawn from their report which can be found online (NII, 2000).

The NII examined the case in detail and found some alarming results. In summary, some of the quality inspection workers had copied reams of spreadsheet data from previously completed lots. The NII identified 22 lots of falsified data (1 lot is approximately 200 pellets). The data was falsified by copying completed quality control spreadsheets and then altering certain aspects so that the spreadsheet appeared to relate to the lot and batch of pellets it was assigned to. In addition, the NII identified a further two lots that contained strong evidence of falsification through copy and alteration of previous spreadsheets. The NII also identified another two lots which had significant duplication of data, although this was later disregarded by BNFL.

This confirmed that the total mass of certain pellet lots was unknown. In some cases, the pellets had already been shipped to some of the Japanese clients including the Takahama nuclear power plant (BBC, 2000b). Kansai Electric, the owners of the nuclear power plant were outraged and demanded that Britain take the MOX fuel back to the UK. Eventually the UK government agreed and took the fuel back to the UK at the cost of several million pounds as well as compensating Kansai Electric £40 million (BBC, 2000a).

The NII found that of the five shifts employed to complete secondary measurements, four were implicated in data falsification. The NII were able to identify when the data was falsified since the dates on the spreadsheets lots were accurate. In total three workers were dismissed including one who admitted falsifying data. An undisclosed number of
workers faced disciplinary action as a result of a wider investigation by BNFL. The NII report also indicates that two groups of process workers must have been involved in the falsification. Firstly the workers who measure and record the pellet dimension data and secondly the quality standards workers who sign off the records at the end of the shift. It is evident that the shift managers were either aware of the falsification or grossly negligent not to realise or report the deceit.

2.1 Safety of MOX pellets implicated in the falsification

BNFL always maintained that the fuel implicated in the falsification of data was safe since it had passed through other automated quality control and safety measurements. The NII concluded that the fuel was safe to use but identified that BNFL had breached the conditions of its Nuclear Site License and suspended the MDF facility. However, BNFL issued a statement in the Independent newspaper (Connor, 1999) suggesting that the MOX fuel pellets could possibly be unsafe:

If the pellets are larger than they should be, they can expand and damage the cladding of the fuel rod. If the pellets are too small, they can vibrate and the pellets can rupture

Upon notification of the falsification, Japanese customers immediately ceased dealing with BNFL. The Japanese Minister for International Trade and Industry and the president of Kansai Electric put pressure on the CEO of BNFL to resign. In 2000 the UK government forced John Taylor out of BNFL and appointed a new CEO.

It should also be noted that MOX fuel bound for German reactors also contained irregularities in secondary measurement data. In this case, the data pertaining to secondary measurements was lost through a computer error and BNFL workers attempted to retrieve the data by copying previous lots of measurements. Although there was no safety risk, the German reactor using the pellets was shut down and all the fuel rods were replaced. The German customers then suspended any future shipments of MOX fuel from BNFL.

2.2 Analysis of the conditions leading to the falsification

The NII identify a number of contributing factors that led to falsification of data, and the next section of this paper will explore the human factors, management issues and technology provision at MDF and examine how these factors could have been mitigated. First and foremost the BNFL workers who falsified data using spreadsheets were guilty of committing serious fraud. There were, however, some mitigating factors that led to the incident.
2.2.1 Human factor and ergonomic issues

The NII acknowledge that poor ergonomic considerations played a role in the incident. Although the process of measuring each pellet was “easy”, it did require great dexterity, concentration and patience. The worker would have to place a pellet in the laser micrometer and then copy down the results from the readout on the micrometer onto a spreadsheet.

Manual copying of data is slow, prone to error and in many cases, unnecessary. The nature of the inspection work was highly repetitive and ‘tedious’ according to the NII. Research shows that highly repetitive work is susceptible to lapses in concentration and errors through Base Error Rate (Panko, 2005; Reason, 1990). Panko (2005) shows that in repetitive ‘simple’ tasks such as taking a reading from an instrument and recording it on paper, the error rate of data entry is likely to be approximately 0.5%. The research shows that more logically complex activities attract a higher error rate, approximately 5%. The studies used by Panko (2005) are data entry activities that do not require a great deal of dexterity, so it is possible that the BNFL workers would naturally commit a higher base error rate given the physical demands of the task.

The design of the workstations used by the BNFL process workers at the MDF facility were also criticised by the NII report. Research shows that poor workstation design can have a negative impact on productivity and worker motivation. This poor workstation design issue is exacerbated when the work involved requires precision and dexterity (Yeow & Sen, 2003).

The NII also criticised BNFL over the fact that the quality control process was disconnected from the rest of the manufacturing process. The process workers responsible for the secondary pellet measurements had very little to contribute other than a notional QA stamp on the lots of fuel inspected. Also, the workers didn’t receive any direct praise or recognition, other than payment for a completed shift and this can impact negatively on workers motivation and self esteem. Research shows that low levels of employee recognition can impact on the motivation and compliancy of individuals (Grawitch, Gottschalk, & Munz, 2006).

Grawitch et al. (2006) identify five sets of principles that contribute to an employee's well being and productive attitude in the workplace. One of the five areas identified in the research is recognition through monetary and non monetary means. The research shows that well recognised workers are happier, more productive and generally more amenable to the organisation they work for. Hence BNFL made serious errors in making the secondary quality assurance process separate from the rest of the production process. The quality inspection workers should have job shared with other employees involved in the manufacturing of MOX pellets, since this would have given them greater involvement and appreciation of the manufacturing process. BNFL could have also involved these workers in visiting sites abroad or in the administration of the process. BNFL simply
didn't offer enough reward for the workers undertaking the job to allow them to be truly satisfied with the job. These ergonomic issues do not excuse the falsification of data, but highlight that one possible consequence of difficult working conditions is non-compliant behaviour.

2.2.2 Management failings

The NII identify a number of poor practices that the management of BNFL should have taken control of. The first is the lack of adequate supervision and control which the NII describe as ‘virtually non-existent’. At the MDF facility, managers spent more time in the MOX pellet production area rather than the fuel rod fabrication area. Checks on the secondary measurement process are described as ad-hoc and supervision of secondary measurements were more geared towards checking progress than ensuring accuracy.

The NII report also states that the shift managers viewed the secondary fuel pellet inspection as a ‘low risk’ job not requiring supervision. This clearly gives out the wrong message to the quality assurance engineers who took the secondary pellet measurements. From the lack of management interest, it may appear that the secondary pellet inspection is a trivial formality and not an essential safety check. The shift managers were also relied upon too heavily, having responsibility for overseeing secondary measurements and ensuring that the plant runs smoothly. One shift manager said that they spent far more time problem solving plant related issues. Other issues, such as supervision duties were allowed to slip.

Indeed when the NII consulted the managers and quality assurance engineers over the importance of the secondary fuel pellet measurements, there was a clear disparity. The management thought that the process workers clearly understood the purpose and consequently the importance of the task assigned to them. The process workers were unaware of the importance and safety implications of the secondary pellet checks. The management therefore had failed to deliver even the most basic of awareness training to the process workers. The shift managers were asked about the ‘secondary pellet measurement instruction manual’ which details the process of secondary measurement. Two of the three shift managers interviewed as part of the investigation had never read the document and another had not seen it before the NII showed it to them. Evidently there was a lax safety culture at MDF.

The lack of management presence at MDF was identified as a major problem. The NII suggest that managers did not spend enough time at the plant and did not have a questioning attitude towards the activities in the plant. During the NII investigation of the data falsification, an NII worker observed conditions whilst inspecting the MDF facility which should ‘not be allowed for a plant manufacturing nuclear fuel’. These conditions
were pointed out to the BNFL management, who quickly rectified the situation. It is disturbing that the BNFL management did not themselves recognise and rectify the problem; they were seemingly unaware of the issue.

These management problems were typical of the poor setup of the MDF facility. In addition, the provision of technology to support the secondary measurement processes was also deemed inadequate by the NII. Although the NII do criticise BNFL for the poor quality computer systems in use, they do not recognise that the use of spreadsheets is a serious oversight in the planning of such a facility.

2.2.3 Technology failings

The NII identify a number of computer security issues such as the lack of password and cell protection on the spreadsheets. However, the NII do not acknowledge that spreadsheets should never have been used for such a task. The choice of spreadsheets as a tool for recording this data was probably a non-decision by the management. It seems likely that the BNFL management didn't consider or fully understand the data recording demands of the quality control process and hence didn't have a more suitable computer system in place to record the data. It seems likely that the use of spreadsheets in this instance was probably reactive, i.e. the need to record data was identified late and spreadsheets were chosen since they were available.

The secondary quality control and safety procedures should have had a dedicated bespoke computer system that automatically input data into a database. The role of the quality control workers would then be one of corroborating the measurements given by the micrometer. The database could also record useful information such as the engineer responsible for signing batches of pellets off. The data could also be made available to the shift supervisors to assist them in corroborating the integrity of the data presented and be backed up. The engineers wouldn't need to enter any data and the risk of falsification or loss of information is greatly reduced. Indeed, it is evident that other than informal checks, there was no process at BNFL to detect fraudulent activity.

The use of spreadsheets at BNFL for the secondary measurement control represents a significant failure to plan the computer system necessary to support the manufacturing process of nuclear fuel. One has to question the ability and thoroughness of the IT/Technology management at BNFL. The nuclear industry is not a stranger to complex, well designed and well controlled computer systems that are governed by standards and best practices. Indeed, there are extensive guidelines published by the UK Health and Safety Commission (HSC, 1998) on the use of computer systems in safety critical settings such as the nuclear industry. The scope of the HSC document is extensive, the report details standards, practices and risks associated with computer and human-computer control systems. There are detailed guidelines to the use of programming languages, development methodologies and testing strategies.
The HSC report does not cover the use of spreadsheets, which are incorrectly labelled as ‘ubiquitous office applications’. In practice, as demonstrated by BNFL, spreadsheets are used well beyond the ‘office’ setting and in this case were fundamental to subverting the safety of nuclear fuel production. The attitude towards spreadsheets demonstrated by BNFL and the omission of serious consideration of spreadsheets in the HSC document highlight a new problem. It would seem that UK nuclear installations find spreadsheets to be ‘throw away’ and operationally unimportant to the organisation. It appears that BNFL, and perhaps the wider UK nuclear industry, use spreadsheets in much the same way that non-nuclear organisations do. Spreadsheet use in organisation tends to be ad-hoc; typically spreadsheets will ‘fill the gap’ that bespoke and corporate systems miss. We know that spreadsheets found in business are error prone (Panko, 1999) and that human factors are important to these errors (Panko, 2008). Occasionally, these errors cause financial loss through relatively simple mistakes. For example, Trans Atlanta corporation lost $24 Million from a copy and paste error (French, 2003). It would seem that the UK nuclear industry is exposing itself to significant risk if spreadsheets are used as commonly as they are in other organisations. Anecdotal evidence suggests that the UK nuclear industry makes extensive use of spreadsheets for data modelling activities similar to proposals outlined by Massachusetts Institute of Technology for the future of nuclear power (M.I.T., 2003).

Since this evidence is anecdotal, more could be learnt from an extensive in-depth study of the use of spreadsheets in the UK nuclear industry. In addition the UK nuclear industry should seek to protect themselves and the larger population by formulating strategies to deal with spreadsheet errors and fraud.

2.3 Cases of spreadsheet fraud

Cases of fraud committed using spreadsheets are not well documented but anecdotally common. The following section explores some examples and discusses the features of spreadsheet fraud.

Butler (Butler, 2002) describes his experiences working for Her Majesties Customs and Excise (HMCE), a public body responsible for collecting taxes in the UK. Butler presents an analysis of sample tax return spreadsheets submitted by members of the UK public. The paper considers seven spreadsheets which were audited using HMCEs specialist spreadsheet auditing software tool, Spreadsheet Auditing Compliance and Examination (SpACE). Although in this case it is difficult to separate innocent mistakes from intentional fraud, Butler noted that the 19 mistakes worth £1.4 million resulted in additional taxes owing and not refunds.

John Rusnak was an investment banker working for Allied Irish Bank (AIB) in international currency exchanges. Between 1997 and 2001 Rusnak lost $691.2 million in bad trades (Butler, 2002). During this period Rusnak devised a system of hiding the losses by exploiting internal validation processes, convincing supervisors that his paperwork need not be checked and subverting corporate risk calculations by manipulating data in spreadsheets. Rusnak managed to continue this fraudulent activity unnoticed for four
years. He was only found out when it was noticed that one of the supervisors had not been reconciling Rumsak’s trades. Upon examining the paperwork associated with the trades, it quickly became apparent that Rumsak was committing fraud and the elaborate lie unravelled.

Perhaps the most famous recent case of fraud comes from Bernie Madoff and his collaborators who managed to mask a Ponzi scheme between 1990 and 2009. A Ponzi scheme is where a bogus trader takes monies from investors under the pretence that the money will be invested in a high yield hedge fund. In actual fact, the money is never invested; Madoff kept all monies from investors in a Chase Manhattan bank account. The focus of the scheme is to recruit more and more innocent investors to pay into the scheme whilst balancing any requests from current investors for cash. When an investor requested monies from Madoff, he would promptly comply using the cash accumulated in the Chase Morgan account so that the operation appeared genuine. As the financial crises hit in 2008, more investors sought to withdraw cash from investments, including Madoff’s hedge fund. Eventually the monies accumulated from investors were exhausted, Madoff’s hedge fund became insolvent and the true nature of the scheme was revealed.

During the 19 years of the schemes operation, Madoff had been convincing investors, other traders and regulators that his operation was genuine. The mechanics of the fraud were based on manipulating historical trading data to backward engineer gains in trades. For instance if Madoff decided he needed to make a 1% profit in the first quarter, he would look at the data leading up to that period and work out which stocks he would need to have bought to arrive at a 1% profit. The spurious data would then be entered into a spreadsheet and Madoff could generate a paper trail for his 2300 clients. Investigators said that upon close inspection of the paper trail, some trades were obviously fraudulent but because Madoff had surrounded himself with co-conspirators and insisted on using an archaic computer system that had to be updated manually, he managed to avoid detection by the authorities. The exact amount of money fraudulently handled in this case may never be known, some estimate the estimate figure as $65 billion (Arvedlund, 2009). Some suggest that the fraud started in the 1970’s and hence the total amount of money involved is likely to be higher (Rushe, 2011). Spreadsheets in this case were used to hide the true nature of the fraud. Spreadsheets provided the paper trail needed and since there was no suspicion of the hedge fund, no further questions were asked. Bernie Madoff was tried for securities fraud and given the maximum jail sentence of 150 years in prison in 2009. A detailed account of the mechanics of the fraud is given by Arvedlund (Arvedlund, 2009).

2.3.1 Detecting spreadsheet fraud

In all the cases of fraud discussed in this paper, spreadsheets were used to mask and legitimise the fraudulent activities of individuals. In Rumsak’s case, spreadsheets were used to hide the losses he made. Madoff used spreadsheets to generate apparently legitimate paper trails to fool regulators and investors. Secondary quality and safety
control process workers altered a few details so that cloned spreadsheet data appeared legitimate. Although we cannot be sure the discrepancies in HMRC tax returns are fraud but if they were, they are a result of simple manipulation of data or calculations in the spreadsheet. We can conclude that spreadsheets are easy to manipulate and that hiding manipulation in the spreadsheet is not difficult. Another interesting feature of the cases considered is the timescales involved. Madoff and Rusnak managed to hide the nature of their activities for significant periods of time, 19 and 4 years respectively. One possible reason for this comes from spreadsheet error research. Research tells us that the most effective approaches to finding spreadsheet errors yield only a 60-80% detection rate. Logically, detecting fraudulent activities in spreadsheets must attract roughly the same success rate, although since the perpetrators are attempting to avoid detection, the success rate may be lower. Clearly, detecting fraudulent activities in spreadsheets is difficult; hence the significant amount of time between the start and detection of these frauds. Research suggests that spreadsheets are particularly exposed to fraudulent activities because the code and data of the spreadsheet are intermixed (Mittermeier, Clermont, & Hodnig, 2005). This intermixing makes auditing the spreadsheet more difficult which in turn makes detecting fraud difficult too. Spreadsheets also tend to suffer from an endemic lack of documentation (Taylor, Meynihan, & Wood-Harper, 1998). Documentation could be used to check the validity of calculations and assumptions and may be useful in detecting fraud. Data types in spreadsheet application software can be manipulated to hide parts of a calculation or exclude values. For instance, cells can be set to data type ‘text’ which displays a ‘text’ number that is not taken into account for calculations since it is non-numerical (Panko, 2006). One can therefore exclude certain cells of the spreadsheet in calculations, whilst appearing inclusive. Values or calculations can also be hidden by utilising cells outside the users monitor display. For example in a sufficiently complex spreadsheet, one could hide a calculation or value on row 1 million, which isn’t visible on screen. This could allow someone to have dummy information in the visible portion of the spreadsheet whilst doing all calculations on another non-visible area.

Perhaps the most important feature of any fraud is trust. The fraudsters were trusted by their managers, supervisors and regulators that their activities were appropriate. The role of spreadsheets was to reassure the trust. As long as spreadsheets appear to show business as usual the trust placed in the individuals is warranted and no suspicion arises. Clearly, more stringent regulation and a more critical appraisal approach is needed from regulators and internal auditors in order to detect fraud in the future. In the same vein, ignorance or collusion of management is another important point to consider. In the case of BNFL, the managers failed to check that the secondary measurements were legitimate and that the process workers were actually doing the job. In Rusnak’s case, he convinced his supervisor not to reconcile his trades and in Madoff’s case he convinced regulators that his operation was legitimate based on his track record as a trader and fraudulent data he produced in spreadsheets. In all of these cases if the regulators managers or supervisors had carried out their role appropriately and scrutinised spreadsheet artefacts properly, it seems likely that the perpetrators would have been caught earlier.
2.3.2 Preventing spreadsheet fraud

Organisations heavily relying on spreadsheets should consider adopting a control system similar to pharmaceutical companies in the US. Pharmaceutical companies have long used spreadsheets for critical activities such as drug testing and have specific legislation relating to use of spreadsheets that reduces the likelihood and impact of errors and fraud. Title 21 CFR Part 11 protects against security violations and ensures the reliability of electronic records (FDA, 2011). Spreadsheets are explicitly mentioned in this legislation which demands companies provide evidence of: audits, validation, electronic signatures and documentation for any software artefact including spreadsheets. The legislation also dictates that electronic artefacts such as spreadsheets be stored in a secure server so that once the spreadsheet has been created and audited, it cannot be changed without authorisation. This should discourage fraud since the process of auditing, validating and documentation would most likely uncover any mistakes or fraud in spreadsheets. Such a system would prevent fraudulent alteration of spreadsheets, since access is limited by ‘lock and key’. Authorisation would be needed to implement any changes with subsequent auditing, validation and documentation. Such a system would make the prospect of committing fraud fraught with difficulties, higher risk and therefore a less attractive option. In addition if all organisations relying on spreadsheets were to use a system that complies with Title 21 CFR Part 11, regulators could easily check compliance.

If one considers the case of BNFL, such a control system would have either discouraged the original fraud or detected the fraud earlier. It is likely that such a system would have either discouraged or caught out both Madoff and Rusnak too. Either the fraud would either have been seen as too risky in the first place or their activities would have been detected earlier. There are other control systems and legislation relevant to spreadsheets (Panko, 2006) but none of these are as comprehensive towards spreadsheets as the FDA regulations.

The evidence suggests that spreadsheet fraud is generally high stakes. In the cases examined either a large amount of money or significant safety risks are implied in the fraud. Organisations should take the initiative to protect themselves and the public and adopt control frameworks similar to those necessary to comply with Title 21 CFR Part 11. Alternatively governments should realise the risk of spreadsheet error and fraud in critical industries such as finance and nuclear fuel production and impose regulations designed to discourage and detect fraud.

3. CONCLUSIONS

The falsification of data at the MDF facility was a worldwide revelation (Wald, 2000) that had serious consequences for BNFL and was the result of poor management and planning. The shift and plant managers were guilty of complacency and ignorance over the secondary pellet inspection processes. Some of the shift managers failed to read the instruction manuals issued on the secondary measurement process. It is clear that the
senior management didn’t check if the manuals had been read or indeed have any sort of best practice on training. As identified by the NII, the lax attitude of MDF workers and BNFL towards health and safety is alarming.

One might be tempted to blame the individuals who falsified the secondary measurement data solely for this fraud. Indeed, falsifying records on a nuclear site cannot be excused, but as this paper has shown, this falsification was enabled through bad management. The working conditions and processes at MDF were a result of significant management failings. These failings result from the poor ergonomic workstations design, lack of awareness of the importance of the secondary measurements, lack of management presence and supervision and a laissez faire attitude to health and safety. However, the greatest failing at the MDF facility came in the planning stage. The use of spreadsheets highlights the inadequacies of the provision of computer systems at the site, especially considering the safety critical nature of the secondary measurement process. Spreadsheets should never have been used for such a task. This suggests that the management were either oblivious to the data recording demands of such a task or categorically failed to provide an adequate solution. If BNFL had properly planned the data capture needs of the process and provided an adequate bespoke computer system for recording secondary pellet data, data falsification would have been impossible. But because BNFL relied on an inadequate technology for recording information, it was easily subverted by willing individuals.

Despite all of the difficulties experienced in the production of MOX pellets at MDF, the plant was scaled up to the Sellafield MOX Plant (SMP) in 2001 at a cost of £660 million (DEFRA, 2001). After the Fukushima disaster in 2011, the Japanese government decided to move away from nuclear power as the risks to public safety were seen as too great. This in turn meant that the SMP facility had lost its only remaining customer and the UK coalition government decided to close SMP permanently (BBC, 2011; NEI, 2011). However, in late 2011 the UK government committed to investing heavily in nuclear energy and announced 8 new nuclear power stations. Several of these new nuclear power stations will use MOX fuel and consequently a new MOX fabrication facility will be built on the Sellafield site to supply these new reactors (Mcalister, 2011). It seems likely that spreadsheets will be involved in the planning and operation of this new plant. In this new plant the management should implement a policy and control framework to manage the use of spreadsheets. This policy should minimise the risk of errors and fraud and should safeguard against a similar data falsification incident in the future.

The use of spreadsheets in BNFL is similar to other organisations that make use of the technology. Spreadsheets were used to fill the gaps left by the poorly planned inadequate bespoke system of the MDF facility as previously highlighted. This is typical of organisations in the business world; spreadsheets silently become part of the IS infrastructure without anyone realising. Anecdotal evidence suggests that spreadsheets are routinely used in data modelling exercises such as fuel degradation, nuclear reactions and plant longevity. There is a danger that the UK nuclear industry does not understand the
risky nature of spreadsheets and as a consequence are exposing themselves to significant risks and dangers. BNFL were fortunate that nothing more untoward happened as a result of subverting safety process at MDF. One cannot ignore that spreadsheets are particularly open to fraud and any future use of the technology should be regulated and controlled. The role of spreadsheets in this fraud was similar to other cases of fraud. Spreadsheets were used to hide the true nature of the activities and to reassure the trust placed in individuals.
References


DEFRA. (2001). SELLAFIELD MOX PLANT - MANUFACTURE OF MOX FUEL JUSTIFIED. Department for the Environment Farming and Agriculture. UK Government publication number 165/01


XLSearch: A Search Engine for Spreadsheets

Michael Kohlhase, Corneliu Prodescu
Jacobs University Bremen
http://kwarc.info

Christian Liguda DFKI Bremen
http://www.dfki.de/cps/staff/liguda

Abstract

Spreadsheets are end-user programs and domain models that are heavily employed in administration, financial forecasting, education, and science because of their intuitive, flexible, and direct approach to computation. As a result, institutions are swamped by millions of spreadsheets that are becoming increasingly difficult to manage, access, and control.

This note presents the XLSearch system, a novel search engine for spreadsheets. It indexes spreadsheet formulae and efficiently answers formula queries via unification (a complex query language that allows metavariables in both the query as well as the index). But a web-based search engine is only one application of the underlying technology: Spreadsheet formula export to web standards like MathML combined with formula indexing can be used to find similar spreadsheets or common formula errors.

1. Introduction

Spreadsheets are end-user programs and domain models that are heavily employed in administration, financial forecasting, education, and science because of their intuitive, flexible, and direct approach to computation. It has been estimated that each year tens of millions of professionals and managers create hundreds of millions of spreadsheets [Pan00]. But we have hardly any tools to mine this immense body of reified knowledge, models, and programmatic experience.

Existing tools center around risk management for spreadsheets via spreadsheet audits that create spreadsheet inventories for an organization, estimate risks of individual spreadsheets, and introduce best practices for risk control (see e.g. [Bur08; NO01]), code reviews that semi-automatically detect risky parts and practices in spreadsheets and try to ameliorate them, and test methodologies that semi-automatically generate test cases for spreadsheets, see e.g. [Rot+01]. Except for the first step in spreadsheet audits, all of these tools are local – i.e. apply to single spreadsheets. A notable exception is the EUSES spreadsheet corpus and the statistics gathered for it in [FR05]. However, even this corpus only contains ca. 4.500 spreadsheets, a number which is multiple orders of magnitude smaller than the spreadsheet inventories of large organizations or what is known to search engines: A spreadsheet auditor reported $10^7$ spreadsheets in a single fortune-50 company at EuSpRIG 2010 and a Google search for filetype:xls reports $1.5 \times 10^7$ hits.
For **global services on spreadsheets** we need tools that scale to very large corpora. In practice, this means two things: *i*) standardized, web-scalable representation formats and *ii*) sub-linear processing algorithms. In this paper, we provide both for the case of spreadsheet formulae, and apply this to a concrete application: the XLSearch engine, which allows to efficiently find spreadsheets by querying for their formulae.

**Organization** In the next section, we will present a machine-understandable vocabulary for the ca. 360 functions, constants, and references used in current spreadsheet programs; this acts as the basis for representing spreadsheet formulae as content MathML expressions. This representation allows us to utilize a pre-existing retrieval engine for mathematical formulae (the MathWebSearch system), which we will describe in section 3 to make this paper self-contained. Section 4 presents an application that harvests formulae and result fragments from a spreadsheet for indexing in MathWebSearch. In Section 5, we describe the XLSearch system, a novel search engine for spreadsheets as one possible application we can build with these components. Section 6 concludes the paper and discusses other applications of the combination of MathML representations and indexing/querying.

**Running Example** To make the technical exposition more coherent, we will use the following situation as a running example:

*Semantex Inc*, a successful financial consulting company has just changed its financial forecasting policy from linear extrapolation to second-order Lagrange extrapolation and is now faced with changing the spreadsheets it is using for forecasting. This change impacts everything from the reporting spreadsheets to tables embedded into powerpoint presentations. Fortunately, *Semantex Inc* has recently carried out a spreadsheet audit and thus has a good overview over all documents that contain spreadsheet tables.

In such a situation, a spreadsheet formula search engine like XLSearch can help, since it can search for variants of the linear extrapolation formula

\[
f(x) \sim f(a) + \frac{x-a}{b-a}(f(b) - f(a))
\]

for a function \(f\) from its values at \(a\) to \(b\). Even though this example was chosen more for expository qualities than for business realism, it already reveals many qualities of the solution.

## 2. SPREADSHEET FORMULAE IN MATHML: SPSHP ONTOLOGY

MathML [Aus+10] is a W3C standard for the representation of mathematical formulae. It contains two sub-languages: *i*) “presentation MathML” for the layout of mathematical formulae – this supports the high-quality presentation of mathematical formulae in browsers and XML-based publishing workflows, and *ii*) “content MathML” for the representation of the functional structure of formulae – this supports interoperability between mathematical software systems. The latter is relevant for our purposes in this paper. Content MathML represents formulae as operator trees consisting of applications of functions to arguments (the apply elements in Figure 1), variables, numbers (mn elements), strings, and symbols. The latter are represented by csymbol
elements; the meaning of a symbol is specified by referencing a content dictionary (CD), which provides information about properties of the functions, definitions, notation definitions and types, identifying the concept in the CD by name (the text content of the csymbol element).

![MathML example](image)

**Figure 1**: SUM(A5:A8)*2 in content MathML

So the main task in defining a MathML representation for spreadsheet formulae lies in providing a set of CDs that specify the underlying vocabulary.

We provide a set of content dictionaries [SPSHP] for the formula translation. We jointly call them the SPSHP ontology. See Figure 2 for a depiction of the theory graph (a modular graph of theories that provide vocabularies of concepts and axiomatizations of the properties of their objects connected by theory morphisms – meaning-preserving transformations; see [Koh06; RK13]).

![Ontology diagram](image)

**Figure 2**: SPSHP: An Ontology for Spreadsheet Functions
The SPSHP Content Dictionaries

The starting points of the SPSHP ontology in Figure 2 are the theories types (spreadsheets naturally induce a type system with flexary functions, optional arguments, and subtypes) and values which introduces the concept spreadsheet values. These are specialized into the subtypes for numbers (theory num with integers, floating point and complex numbers), strings (theory strings), and truth values (bool). The theory spsht provides the basic building blocks of spreadsheets (cells, rows, columns, tables) and their types. Theory error provides representations of typesheet errors raised by spreadsheet programs. Theory arith provides representations of the elementary arithmetic operations, which are not represented by spreadsheet functions but by the operators $+$, $-$, and $\ast$, etc.

From all this material, theory spshform introduces the concepts of “value expressions” (expressions constructed from cell/range references, functions, strings, and numbers; they evaluate to spreadsheet values or errors) and value expression lists. Together with the flexary function types, the latter induce natural types of spreadsheet functions like SUM, which take arbitrarily many arguments that can be interpreted as lists of cell values. For instance, in the formula $\text{SUM}(A5:A8, 7, 3)$ the range description $A5:A8$ induces a set of values in the spreadsheet computation. Theories stats to text provide the symbol declarations of the ca. 360 spreadsheet functions themselves; they follow the grouping found in spreadsheet applications. Finally, the theory spshp collects all the SPSHP sub-theories by importing them for convenience.

Interoperability

So far, we have been able to keep the CDs in the SPSHP ontology independent of the particular spreadsheet application (MS Excel, OpenOffice Calc, Apple Numbers, Google Spreadsheet, etc.), as the formula languages of the applications have been standardized for interoperability. But there are functions whose implementations differ between applications, e.g. the COUNTIF function to count the number of cells which contain a certain value. If the cells A1 and A2 contain the value TRUE, then the formula $\text{COUNTIF}(A1:A2;1)$ evaluates to 0 in Excel and to 2 in OpenOffice Calc.

In this case, we extend the SPSHP theory graph with application-specific theories as indicated in the picture on the right. Here, oo-stats and xls-stats are theories that specify functions whose semantics differ and that therefore cannot be specified in the application-independent theory stats. The theories ooc (for OpenOffice Calc) and xls (for MS Excel) are convenience theories, which collect the application-specific theories – note that by inheritance the theories *-stats contain all the functions from stats – just like spshp does in the application independent case. Intuitively, these theories represent the sub-ontologies for specific applications and are used for concrete translation projects. Note that e.g. ooc and xls share the majority of the specification and thus constitute a good basis for spreadsheet system interoperability (without translation) at the semantic level. However, even the application-specific functions are often aligned and very similar, thus we can specify views between the application-specific theories. OMDoc views\(^1\) map concepts of the source theory

\[^1\]represented as dashed arrows in Figure 3; the label $j: \psi$ specifies the name $i$ and the translation $\phi$.
to expressions in the target theory. For the view $j : \psi$, we have to implement the COUNTIF function from theory xls-stats in terms of the function COUNTIF from theory oo-stats, e.g. by removing truth values from the value formula lists in the arguments (the dual view $i : \square$ can be defined similarly). Furthermore, the views between the application-specific component theories induce top-level views between theories oo and xls that can (eventually) be used for semantic interoperation between spreadsheet applications, since they allow meaning-preserving translations of spreadsheet formulae.

3. INDEXING AND QUERYING FORMULAE BY UNIFICATION

MathWebSearch is an unification-based search engine for the efficient retrieval of mathematical formulae [MWS; KMP12]. The system consists of the three main components pictured in Figure 4. The crawler subsystem collects data from the corpora\(^2\). It transforms the mathematical formulae in the corpus into MWS Harvest\(^s\) (XML files that contain formula-UR1reference pairs) and feeds them into the core system. The core system (the MathWebSearch daemon mwsd) builds the search index and processes search queries: it accepts the MathWebSearch input formats (MWS Harvest and MWS Query; see [KP]) and generates the MathWebSearch output format (MWS Answer Set). These are communicated through the RESTful interface restd which provides a public HTTP API conforming to the REST paradigm. The system supports two main workflows:

1. The crawler sends an MWS Harvest to mwsd. The XML is parsed and an internal representation is generated. This is used to update the Substitution Indexing Tree and consequently the database.
2. The user sends an MWS Query to mwsd. The XML is parsed, an internal query is generated. Using an efficient traversal of the index tree, formulas matching the search term are retrieved and aggregated into a result. This is translated to an MWS Answer Set and sent back to the user.

The system has been tested on large sets of formulae. Memory usage is linear (on average, 40 Mb for 1 Million formulae), while query times are fairly constant with respect to index size\(^3\), averaging at 40 ms per query.

\(^2\)Note that we envision essentially one crawler per corpus. The crawlers are specialized to the respective formula representation, the organization and access methods to the corpus, etc.

\(^3\)However, they do depend on the complexity of the query.
4. HARVESTING FORMULAE FROM SPREADSHEETS

In this section we describe the process of parsing spreadsheets and generating formula harvests that can be used by MathWebSearch. We are not only interested in the formulae, but also in the context they are used in. Therefore, we describe in Subsection 4.1 the context information we extract and the structure of the resulting harvest. The formula parser and converter is described in Subsection 4.2, the structure detection module for finding the context information is presented in Subsection 4.3, while the harvest generator is described in Subsection 4.4. We describe the process of generating the harvest by using a slight modification of the Winograd spreadsheet from [KK09] (see Figure 5(a)). Our spreadsheet uses linear extrapolation for calculating the revenues and expenses in the projected years (see Section 1).

4.1 The Harvest Structure

Following [KK09], we use the term legend for those non-empty cells that do not contain input or computed values, but contain text strings that give auxiliary information on the cells that do. We call a grid region a functional block (FB), if that region could be interpreted as a function which maps elements from a legend to values. As the function is meant to be an intended function of the spreadsheet creator, it is immaterial whether the values are calculated or inputted. For example, the region B13:F13 of Figure 5(a) could be interpreted as a function, which maps years to the total expenses in that year, and the region B4:F4 as a function that maps a year to the revenues of that year. We call a functional block computed if all formulae are cp-similar, i.e. if they only differ in their cell references like $\text{SUM}(B4:B13)$ and $\text{SUM}(C4:C13)$. Because all expenses for the projected years are calculated by linear extrapolation the area E7:F11 in Figure 5(a) is a computed FB. A formal model which defines functional blocks and legends as mathematical objects is introduced in [Lig12].

To compute a harvest we need to find all computed functional blocks in a spreadsheet together with the parts of the legends surround them. For each computed FB, we

<table>
<thead>
<tr>
<th>MathML Formula</th>
<th>Position information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>Excel formula</td>
</tr>
<tr>
<td>XHTML Snippet</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Harvest for an FB

Figure 5: A spreadsheet and a cutout of a computed functional block
create a harvest of the structure shown in Table 6. The contents of the surrounding
legend cells are used as keywords which can be used to curtail the formula search.
Because all formulae in a computed FB are cp-similar, we create one location-
independent MathML representation per FB with the parser from Section 4.2. For
representing a functional block as search result to a user, an XHTML snippet
containing the FB and the surrounding legends is generated, like the one shown in
Figure 5(b) for the FB E7:F11. Furthermore, the concrete formula of the upper-left
cell from to FB is saved for search result representation. At last, the position
information which locates the spreadsheet and the region in which the FB was found is
also stored.

4.2 Formula Parsing

We used the open source parser generator Antlr [Par13] to create a parser that
transforms an Excel\footnote{As formulae in other spreadsheet programs have nearly the same syntax as Excel, our parser can be tailored for those with minimal adjustments.} formula into an abstract syntax tree (AST). Figure 7 shows
the resulting AST for the formula \( C7 + (E$3 - C$3)/(D$3 - C$3) \ast (D7 - C7) \) from cell
E7. The parser is aware of different operator priorities, nested formulae and cross
worksheet references, and transforms cell references like A5 to an integer based row and
column pair. Creating MathML from ASTs is an easy programming exercise given a
vocabulary of spreadsheet symbols that act as counterparts of the AST nodes. The
SPSHP presented in Section 2 fills this requirement.

4.3 Structure Detection in Spreadsheets

To find functional blocks and their legends, we use a simplification of our structure
detection unit (SDU, see [Lig13]), which classifies each cell as “legend”, “FB”, “empty”
or “hidden” and then aggregates regions into computed FB with legends.

\[\text{Figure 7: Abstract Syntax Tree of } C7 + (E$3 - C$3)/(D$3 - C$3) \ast (D7 - C7)\]
Cell Classification  SDU uses a simple heuristics to classify some cells: formula cells are always “FB” and “nonempty”, non-formula cells that contain at least 75% letters are classified as “legend”. This heuristics are appropriate, because a misclassification of a non-formula functional block cell as legend is unproblematic, as it will just be integrated into the context of a computed FB (see “Area Detection” below). In particular, this heuristic correctly classifies the cells of Figure 5(a), except B3:F3, B4:D4 and B7:D11. Afterwards, hidden cells (like the cells C1:F1, C2, D2 and F2 in Figure 5(a)) are set to the type of the cell that hides the other ones (e.g. C1:F1 are set to the type “legend” of cell B1 in Figure 5(a)).

Area Detection After classifying cells, SDU marks regions with cp-similar formulae as a functional block. In our example (see Figure 5(a)), we obtain the blocks E4:F4, B13:F13, B15:F15, and E7:F11, which SDU searches for the legends of each functional block. It starts in the first row of the FB and iterates upwards until it finds a row which contains at least one legend cell and no functional block cell in those cells that are right above the functional block. Then it iterates further upwards to the last row that is not empty and does not contain a functional block cell. The region between those rows which is right above the FB is taken as a legend region for the functional block. SDU repeats that search on the left side of the functional block and iterates through the columns instead of the rows. In our example in Figure 5(a) SDU finds a legend area in E1:F3 and A7:A11 for the functional block E7:F11.

4.4 Harvest Generation

For the generation of a XHTML snippet (see the one in Figure 5(b)) from the results of the area detection, we use the Apache POI API [POI] to get the relevant data from a spreadsheet. Therefore, we create a document representation of the original spreadsheet, and delete all worksheets except the one that contains the functional block. From the remaining sheet, we delete all rows and columns which do not contain a cell that is part of the functional block or surrounding legend. Afterwards, we use the HTML exporter from Apache POI to create an HTML document which is then transformed to XHTML by using JTidy [Jti].

For transforming a spreadsheet to a snippet, merged cells need some special attention. In our example, the cell B1 in Figure 5(a) contains the header "Year" that is also relevant for the functional block E7:F11. Therefore, we move the content of merged cells that are partially inside and partially outside of a relevant legend region from the outside (e.g. from B1) to the inside part (e.g. to E1). As the HTML converter is not aware of merged regions, we delete all of them afterwards to avoid confusion.

5. XLSearch, A SEARCH ENGINE

We will now assemble a spreadsheet search engine from the components introduced above. Like any web search engine, XLSearch consists of a crawler, the core indexing/query engine (see Section 3), and a front-end that accepts queries and displays results.

Crawler As we imagine that the XLSearch engine will usually be deployed in institutional settings, which – after a spreadsheet audit – have created a spread-

---

These cells can classified by other heuristics or via decision trees (see [Lig13]).
sheet inventory, we have restricted ourselves to a simple crawler that maps the MathML converter from Section 4.2 over a list of URIs of spreadsheets and generates MathWebSearch harvests from that are passed on to mwsd for indexing. But for the application in the search, we do not want concrete cell references in the index, since they are meaningless outside spreadsheet context. Therefore our parser variablizes cell and range references to MathWebSearch meta-variables (q:qvar in Figure 8), which can be instantiated in the search. In our example, the formula

\[
C7+ (E$3 - C$3)/(D$3 - C$3) \times (D7 - C7)
\]

becomes the MathML expression in Figure 8.

![Figure 8: Index Entry for C7+ (E$3 - C$3)/(D$3 - C$3) * (D7 - C7)](image.png)

**Front End** For simplicity, we use a web-based front-end that resembles web search engines for XLSearch; other front-ends, which e.g. embed XLSearch functionality into the spreadsheet program itself are imaginable, but are left to future research. Figure 9 shows a typical situation: the user has entered the query in the text box at the top. The query interface

1. accepts spreadsheet formulae in native syntax extended with query variables (names prefixed by `?`)
2. converts them to MathML by the parser from Section 4.2 extended by a rule that transforms `?foo` to `<q:qvar name="foo"/>`, and
3. sends that to mwsd via its RESTful interface via a HTTP POST request.

In our example we see the formula `?fa+ (\(?x - ?a)/(?b - ?a) \times (?fb - ?fa)`, which queries the index for linear extrapolation formulae.

mwsd returns a list of hits, all representing indexed formulae which unify with the query. Each hit carries a harvest datum as in Table 6 and keywords extracted from the containing FB, providing further information to the user. In Figure 9, the
mwsd has found three hits. For each of these, the raw Excel formula, keywords and the URI reference (the URI of the spreadsheet and the FB identifier) are displayed. By clicking the second hit, a result snippet is revealed, in the form of the functional block with legends.

**Deployment & Demo** We have deployed an instance of XLSearch at http://search.mathweb.org/xl/ which indexes the EUSES corpus [FR05] with ca. 4.5 thousand spreadsheets.

6. **CONCLUSION**

We have presented a novel search engine that allows finding and accessing spreadsheets by their formulae. Such a search engine has multiple applications: it can be used to spot problematic formulae (e.g. known errors) in large spreadsheet corpora, or find re-usable tables (code blocks) in legacy spreadsheets leading to cost savings.

The main algorithmic core of the XLSearch engine is the pre-existing MathWebSearch formula search engine, which has been under constant development in our group for half a decade. For the application in the spreadsheet domain, we have developed a standardized vocabulary (the SPSHP ontology) that allows to transform spreadsheet formulae into content MathML, which is the core of the input/query format of MathWebSearch.

**Further Applications** As the average query time is in range of 10-50 milliseconds, searches can even be utilized for very interactive applications. For instance, a variation of Netspeak [NSpk] for spreadsheet formulae. While Netspeak is able to find the most common word that is used in a phrasal context, our search engine finds the most common subformulae in a formula context. This can be very helpful for finding a very long and complex formula which can be just partially remembered by a user.
Alternatively, the spreadsheet system could monitor the number of similar formulae by sending off unification queries every time delimiters balance. As formulae in an organization are bound to be similar, an unexpected drop in the similar formula number could indicate a typo or error; and the author can be alerted in real time.

The SPSHP ontology supports applications in its own right: via the standardized format, formulae can be exported to other applications, e.g. via the clipboard (which supports MathML). Furthermore, formulae can be simplified or partially evaluated by standard symbolic computation systems, which can also also be used for query expansion, i.e., by searching for the variant \( \text{SUM}(C7; (E$3-C$3)/(D$3-C$3) \ast (D7-C7)) \) of the linear interpolation formula.

**Future Work** Currently, the search engine hits are ranked by alphabetically sorting the file URIs. We expect that — as in Web search — ranking will be a crucial factor in the efficacy of search, and we want to explore this aspect further. We conjecture that for spreadsheets, where pagerank-like algorithms are hardly applicable, application-specific traits will have to be taken into account: [Sha+12] finds “Studies suggest that location, file type, time, keywords, and associated events are the attributes best remembered”; we are currently thinking about organizing search results by a file system tree widget with folding and unfolding interactions, if the corpus is organized this way.

Finally, we are thinking about including cognitive cues like the user-selected names for cells and ranges (see e.g. [Bew03; Spr]) into the search process as additional keywords.

**Acknowledgements** Work on the concepts presented here has been partially supported by the German Research Foundation (DFG) under grant KO 2428/10-1 and HU 737/6-1 and the Leibniz association under grant SAW-2012-FIZ KA-2. The authors are indebted to the SiSsI group for discussions and insights on abstract spreadsheets.
7. REFERENCES


ABSTRACT

Excel 2013 (version 15) includes an add-in 'Inquire' for auditing spreadsheets. We describe the evolution of such tools in the third-party marketplace and assess the usefulness of Microsoft's own add-in in this context. We compare in detail the features of Inquire with similar products and make suggestions for how it could be enhanced. We offer a free helper add-in that in our opinion corrects one major shortcoming of Inquire.

INTRODUCTION

From the earliest days of spreadsheets, auditors and reviewers have been seeking ways of gaining an understanding of their underlying structure of formulas. Once Excel arrived and large worksheets became common, it rapidly became obvious that simple formula listings were too tedious to work with, and spreadsheet auditing tools were developed that provided two indispensable services: a visual overview of the structure of a spreadsheet, and a detailed listing of features ('bad smells') which from the experience of the auditors may indicate an actual error or a weakness that could easily materialise as a defect in use.

One of the first of these, and still in use, is the Spreadsheet Detective which created maps using single-letter codes such as F for Formula, and symbols such as > to indicate a formula was copied to its right. With a small column width and a black-and-white printer, one could quickly see how a spreadsheet had been structured. Later, colouring was added tools and auditing software like SpACE added many colourful schemes and detailed lists of suspect features. The authors of SpACE also published one of the first frameworks for spreadsheet audit, as presented by Ray Butler in a pioneering EuSpRIG 2000 paper "Risk Assessment For Spreadsheet Developments: Choosing Which Models to Audit". For a later view of how financial service companies do it, see "A Typical Model Audit Approach: Spreadsheet Audit Methodologies in the City of London" by Grenville Croll.

Many more such products have been created in what was largely a cottage industry. The author has his own product, XLTest. Some added features to assist with building and debugging the formulas in certain types of spreadsheets, eg financial modelling, such as Operis OAK. Some, with the impetus of the FDA's 21 CFR Part 11, focused on locking down spreadsheets under control, such as ABB's DACs. More recently, enterprise spreadsheet management systems such as Liquidity, ClusterSeven and Cimcon have provided a visualisation of the use and development over time of systems of spreadsheets, and have added monitoring and control features. For example, see the white papers by Clusterseven or the EuSpRIG 2007 paper "Managing Critical Spreadsheets in a Compliant Environment by Soheil Saadat". A list of these tools is maintained at http://www.sysmod.com/ssllinks.htm

These tools are like standard static software analysis. While they considerably increase the efficiency of the auditor in getting through large amounts of routine analysis, they are less useful as a predictor of the cost of an audit. This is discussed by David Colver in his EuSpRIG 2011 paper "Drivers of the Cost of Spreadsheet Audit". Neither will they...
detect errors in logic. In their EuSpRIG 2010 paper "The Detection of Human Spreadsheet Errors by Humans versus Inspection (Auditing) Software", Salvatore Aurigemma and Raymond R. Panko reported "[tools] were almost useless for correctly flagging natural (human) errors in this study."

These tools are excellent in raising questions; the remedial actions are however a matter of skill and judgment, as discussed by Louise Pryor in her Eusprig 2008 paper "Correctness is not enough".

**THE INQUIRE ADD-IN**

Excel version 15 has an Add-in that came from Microsoft’s purchase of the Prodiance SpreadsheetIQ product, which adds an "INQUIRE" tab to the Ribbon.

To activate the Add-in, click File > Options > Add-Ins, in the Manage list, select COM Add-ins, click the Go button, and check the Inquire Add-in.

**OVERVIEW OF THE COMMANDS**

The Workbook Analysis Report creates an interactive report showing detailed information about the workbook and its structure, formulas, cells, ranges, and warnings.

The Workbook Relationship Diagram creates an interactive, graphical map of workbook dependencies created by connections (links) between files. The types of links in the diagram can include other workbooks, Access databases, text files, HTML pages, SQL Server databases, and other data sources.

The Worksheet Relationship Diagram creates an interactive, graphical map of connections (links) between worksheets both in the same workbook and in other workbooks.

The Cell Relationship Diagram creates a detailed, interactive diagram of all links from a selected cell to cells in other worksheets or even other workbooks. These relationships with other cells can exist in formulas, or references to named ranges. The diagram can cross worksheets and workbooks.

The Compare Files command lets you see the differences, cell by cell, between two open workbooks. Results are colour coded by the kind of content, such as entered values, formulas, named ranges, and formats. A window shows VBA code changes line by line.

The Excel 2013 preview had an Interactive Diagnostics command that was withdrawn from the released version.

The Clean Excess Cell Formatting command removes excess formatting and can greatly reduce file size. This helps you avoid "spreadsheet bloat," which improves Excel's speed.
The Workbook Passwords command maintains a password list, which will be saved on your computer encrypted and only accessible by you. The password list is needed so that Inquire can open its saved copy of your workbook.

The Help command gives details and examples of these commands.

In the following section we describe these commands in detail, give some of their uses and show their limitations.

**WORKBOOK ANALYSIS REPORT**

If the workbook has unsaved changes, you will be asked "Only changes that have been saved will be included in this analysis. Would you like to save your changes now?" Click Yes to save the workbook.

The Summary sheet shows the basic properties: Workbook name, Creation Date, Modified Date, File Size (bytes), Title, and Author; and then the count of each of these detailed findings which are listed in separate subsequent sheets. It would be useful for the Summary sheet to be hyperlinked to the sheets listed.

<table>
<thead>
<tr>
<th>Linked Workbooks</th>
<th>DDE Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Connections</td>
<td>Visible Sheets</td>
</tr>
<tr>
<td>Hidden Sheets</td>
<td>Very Hidden Sheets</td>
</tr>
<tr>
<td>Formulas</td>
<td>Array Formulas</td>
</tr>
<tr>
<td>Formulas With Errors</td>
<td>Formulas With Logical Values</td>
</tr>
<tr>
<td>Formulas With Numeric Values</td>
<td>Formulas With Date/Time Values</td>
</tr>
<tr>
<td>Formulas With Textual Values</td>
<td>Formulas With Numeric Constants</td>
</tr>
<tr>
<td>Formulas With Textual Constants</td>
<td>Formulas With Nested IF Statements</td>
</tr>
<tr>
<td>Formulas Without Cell References</td>
<td>Formulas Referencing Blank Cells</td>
</tr>
<tr>
<td>Formulas Referencing Hidden Cells</td>
<td>Formulas Referencing Text Cells</td>
</tr>
<tr>
<td>Formulas Referencing External Workbooks</td>
<td>Formulas Formatted As Text</td>
</tr>
<tr>
<td>Positive Formulas</td>
<td>Negative Formulas</td>
</tr>
<tr>
<td>Unique Formulas</td>
<td>Duplicate Formulas</td>
</tr>
<tr>
<td>Inconsistent Formulas</td>
<td>Cells With Dependents</td>
</tr>
<tr>
<td>Cells With Textual Constants</td>
<td>Cells With Numeric Constants</td>
</tr>
<tr>
<td>Cells With Comments</td>
<td>Cells With Validation Criteria</td>
</tr>
<tr>
<td>Cells With Conditional Formatting</td>
<td>Cells With Numerics Stored As Text</td>
</tr>
<tr>
<td>Invisible Cells</td>
<td>Used Input Cells</td>
</tr>
<tr>
<td>Unused Input Cells</td>
<td>Occupied Cells</td>
</tr>
<tr>
<td>Merged Cells</td>
<td>Blank Cells</td>
</tr>
<tr>
<td>Blank Referenced Cells</td>
<td>Unlocked Cells</td>
</tr>
<tr>
<td>Hidden Rows and Columns</td>
<td>Named Items</td>
</tr>
<tr>
<td>Named Items With Errors</td>
<td>Warnings</td>
</tr>
</tbody>
</table>

**HOW TO USE THE SUMMARY REPORT**
You would scan this sheet and note items of interest, such as items that you would not expect to find, or to find in different numbers to what is reported. What you notice is going to be influenced by your previous experience and expectations of this spreadsheet, or similar spreadsheets from this user or business unit, and your knowledge of Excel in general. If the meaning of any item is not clear, you may need to refer to some training materials, or ask questions on online Excel forums, to obtain further information. It is beyond the scope of this paper to provide such explanation.

WHAT IS MISSING FROM THE SUMMARY REPORT

The Summary sheet lists for the properties only the Title and Author. Other tools show the name of the previous author, the last print date, and custom document properties which may indicate company-specific information such as contact people or tracking numbers.

It does not report whether the workbook was saved with Automatic or Manual calculation, nor the existence of Circular references or Iteration.

It does not report Styles – not even a count, which would indicate workbooks with excessive styles; nor unusual number formats that may hide data.

It does not report VBA – not even the presence of macros is shown.

Neither does it show the use of Data Lists, Pivot Tables, Consolidation Sources, Views or What-if features such as Scenarios or Goal Seek.

The detailed list sheets all begin with three rows of two merged cells at the top which means that if you copy and paste either the heading in A1 or the filename in A2 into a Word document, or into a worksheet as values, you get two copies.

THE REMAINING SHEETS IN THE WORKBOOK ANALYSIS REPORT

The lists of links (Linked Workbooks, DDE Links, Data Connections) give only the bare name or link reference, not where they were found, and do not indicate whether the link source file actually exists, nor whether the linked content is up-to-date, nor for queries where they are used nor the CommandText.

There are separate sheets to list the names of Visible Sheets, Hidden Sheets, and Very Hidden Sheets, but it does not indicate the presence of Chart sheets, nor of Macro or Dialog sheets which may exist in very old Excel files. It does not give the following information which other tools report such as excessive used range; the Print Area; Page Header and Footers; Objects such as embedded charts or graphics or Form or ActiveX buttons; whether a worksheet has the old Lotus evaluation rules set; whether the sheet windows are not displaying the Formula Bar, Scroll Bars, Row & Column headings; Zero values; and user-definable search strings such as "password".

The 'All Formulas' sheet as one per line in A1 style, with no concept of how the same formula in relative terms has been copied over areas. This is to me the single biggest failing. The vast majority of spreadsheets contain rows, columns, or blocks of repeated formulas, also known as clones or copies of a unique formula. One of the most important tests of the structural soundness of a spreadsheet is to verify the integrity of such areas, that they are neither too small nor too large relative to similar areas, and that no inconsistent entries exist inside them. This has been a feature of nearly every other spreadsheet formula audit tool. For a detailed discussion of the mechanics see the EuSpRIG 2008 paper by Markus Clermont "A Spreadsheet Auditing Tool Evaluated in an Industrial Context"15.
I find it hard to believe that such a list, with an implication that each formula is different and has to be individually checked off in a list, would have been accepted as useful in a commercial product. Their list of “unique formulas” is therefore exactly the same as the complete formula list, except for any formulas which are duplicated in A1 style. Therefore we make available an IQHelper add-in to group similar formulas together to make checking easier. It is available from http://sysmod.wordpress.com/2013/03/06/excel-2013-inquire-addin-improved-iq/

The 'Array Formulas' sheet is interesting as a separate list, but I would prefer them to be simply distinguished in the 'All Formulas' sheet by a separate indicator that the formula is an array. Currently they are recognisable by being enclosed in braces {}.

The 'Error Formulas' sheet lists those cells with #Error values. An overflow error, such as a negative or excessive date value, is not reported in this sheet but in 'All Formulas' as a real date (eg 14/04/1791 01:17:02) rather than the ###### error value.

I'm not clear what value there is in listing Logical Formulas, Numeric Formulas, Date/Time Formulas, and Textual Formulas; but somebody probably wants it. My preference is for sheets to be given a map with a specific colour for each data type.

'Numeric Constant Formulas', or 'Textual Constant Formulas' that is, cells with constants embedded ('hardcoded') in the formulas, are a useful list to check for (eg) adjustment values that should be moved to their own cells.

'Nested Ifs' shows formulas with more than one level of IF nesting. This is a useful indicator of a common cause of difficulty in understanding formulas with multiple logic paths. It would be even nicer if the depth of nesting was shown. It would be even better if the complexity measure was applied to All Formulas, rather than just nested Ifs.

'No Cell Refs' is a list of calculations like =1+2+3 but they can also be system functions such as =NOW().

'Blank Cell Refs' is one of these unavoidable reports where most of the information is not useful but buried in there are some possible problem formulas. Excel's error checking has that option turned off by default. Again, I prefer to see these highlighted in context by a mapping tool. Like all auditing tools, all that can be done is to raise the question and leave the judgment to the reviewer.

'Hidden Cells Refs' is potentially of interest, and again a judgement call as to whether the reason for hiding the source cells is for readability (hiding detail not normally needed), analysis (eg grouping or filtering) or obscurity (potential deception).

'Text Cells Refs' can be simply formulas that copy or transform text values, but this report may also pick up the problem where a sum range includes text cells by mistake, or cells accidentally formatted as text.

'External Workbook Refs' is a useful place to collect all the external references. It is clever enough to include external links embedded in defined names. For some reason it also reports some internal cross-sheet references; for example: ="Nov"06!'G5 is reported, but ="Oct 06!'F4 is not reported.
'Formatted As Text' can be useful to pick up formulas that are not calculating because they were entered into cells formatted as text. Sometimes people temporarily remove formulas by prefixing them with an apostrophe and then forget to reinstate them.

'Positive Formulas' eg =+B1 may be a sign of old Lotus 1-2-3 habits, where users begin a formula entry with a plus sign rather than an equals sign but are not normally seen as a problem.

'Negative Formulas' eg =-B1 could be an entry made with an initial minus sign or potentially be a keyboard 'fat finger' error because the minus key is beside the equals key on the keyboard. It's not one I've ever seen but there's always a first time.

'Unique Formulas' is probably the most useless listing there is, because all the formulas are regarded as unique in A1 style. I think they simply didn't understand how the term 'unique formula' is normally used in auditing. It is used to describe formulas which are unique in R1C1 style. Other terms for these are 'root formulas' which are copied in blocks, or 'schemas'.

'Duplicate Formulas' shows formulas which are identical on the same sheet in A1 style. This is actually a correct definition of 'duplicate' in this context. It may be useful in order to show several references to the same cell, typically a parameter or title.

'Inconsistent Formulas' are those reported by Excel in its Error Checking rules. It includes those that the user has chosen to ignore and so suppress the green triangle indicator. The following table has the complete list of Excel's rules:

<table>
<thead>
<tr>
<th>Excel Error Checking rule</th>
<th>Where reported in Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cells containing formulas that result in an error</td>
<td>'Formulas With Errors'</td>
</tr>
<tr>
<td>Inconsistent calculated column formula in tables</td>
<td>n/a</td>
</tr>
<tr>
<td>Cells containing years represented as 2 digits</td>
<td>Not reported</td>
</tr>
<tr>
<td>Numbers formatted as text or preceded by an apostrophe</td>
<td>'Numerics As Text'</td>
</tr>
<tr>
<td>Formulas inconsistent with other formulas in the region</td>
<td>'Inconsistent Formulas'</td>
</tr>
<tr>
<td>Formulas which omit cells in a region</td>
<td>n/a</td>
</tr>
<tr>
<td>Unlocked cells containing formulas</td>
<td>n/a</td>
</tr>
<tr>
<td>Formulas referring to empty cells</td>
<td>'Blank Cell Refs'</td>
</tr>
<tr>
<td>Data entered in a table is invalid</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Other cell checks that I would like to see are:
- Overflow error
- Formula too long
- Formula with double minus
- Numeric text right aligned
- Range_Lookup missing the fourth parameter
- Formula Hidden
- Formula Unlocked

'Cells With Dependents' lists every cell that is further referenced by other cells. I don't see the value in this. I prefer a colour map that gives a specific colour to each number of
dependents, so that if you see a cell that has five dependents in a block where the surrounding cells have four or six, you have something to check.

'Textual Constants' and 'Numeric Constants' list every input value in the workbook. This may have a value for decision models with a few inputs and many calculations, to guide in knowing what cells could be grouped together in an input area.

The 'Comments' sheet lists the sheet name, cell address, formula, value, comment author and text. It may be useful to indicate what people thought useful as meta-data, as information about the spreadsheet formulas and observations on the results. It may reveal previous users and obsolete comments too.

'Validation Criteria' and 'Conditional Formatting' are useful lists as they make obvious structure which is normally under the surface. However, it does not list the actual conditional formatting rules, merely the number of formats applied to each cell.

'Numerics As Text' is also one of Excel's error-checking rules. It is particularly useful when reviewing imported data

'Invisible Cells' are those in hidden rows or columns. It does not report cell contents hidden by formatting codes such as ";;;;" or white text, which other auditing tools pick up as bad smells.

'Used Input Cells' are all those which are referred to by formulas. 'Unused Input Cells' are those not referenced by formulas. It does not recognise cells referred to indirectly by OFFSET or INDIRECT.

I can't see the value in a simple list. A colouring map in place would indicate more usefully areas or cells in or beside other used areas, that ought (or ought not) to be referenced.

In this map from XLTest, most cells in the table have 2 dependents (coloured yellow) but two rows have a different colour which means they have only one dependent. This is because the formula to their right omits those cells.

'Merged Cells' may be useful because references to merged areas, or a copy/paste of them, are known to be problematical.

'Occupied Cells' is a list of every value and formula in the workbook. Can anyone think of a use for this? Comparing two versions maybe? Its complement is 'Blank Cells', all the unused cells. What on earth is that for?

'Blank Referenced Cells' may be useful to indicate missing values but again I'd much prefer to see that in context rather than a list.

'Unlocked Cells' indicates that the user has thought of input cells and protection. But the Inquire add-in does not show when sheets are protected. And it does not list separately unprotected cells with formulas, which is one of Excel's error-checking rules. Neither does it list formula cells with the 'Hidden' protection checked. 'Hidden Rows and

<table>
<thead>
<tr>
<th>Number of plays per week:</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>130</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>229</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>132</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>120</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>130</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>229</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>132</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>120</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>130</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>229</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>132</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>120</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Royal Royalty €465.35
Columns’ is a useful list, although curiously it is sorted by range address regardless of sheet name. It reports rows or columns fully hidden but not very narrow ones which are visually hidden.

‘Named Items’ only lists the visible names, not the hidden ones. And it does not highlight which names are duplicated with local and global scope, although if they are visible it reports in the Warnings sheet that they exist (‘Workbook contains duplicate named ranges.’)

‘Named Items With Errors’ is fair enough, although I’d simply show that as a filter on the full Named Items list.

The final ‘Warnings’ sheet gives miscellaneous other findings, although there is no documented list of what it is capable of reporting. When I asked that on the Microsoft forums, I was asked "Can you write a justification for this idea? Why document the message list is necessary." They found it hard to believe I actually wanted to know what this add-in could do! As far as I can tell from internal evidence, the possible warnings are:

1. Workbook is setup to change results to same precision as display.
2. Workbook contains formulas with errors.
3. Workbook contains hidden rows or columns.
5. Workbook contains invisible cells.
6. Workbook contains unlocked cells.
7. Workbook contains duplicate named ranges.
8. Workbook Title has not been set.
9. Workbook Author has not been set.
10. Workbook is setup for R1C1 reference style.
11. Workbook contains sheet names with leading and/or trailing blanks. The sheet names are: . . .

### COMPARISON OF REPRESENTATIONS OF REPEATED FORMULAS

Inquire:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sheet Name</td>
<td>Cell Address</td>
<td>Formula</td>
<td>Value</td>
</tr>
<tr>
<td>----</td>
<td>-------------</td>
<td>-------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>46</td>
<td>Oct 06</td>
<td>G11</td>
<td>=SUM(C11:F11)</td>
<td>936</td>
</tr>
<tr>
<td>46</td>
<td>Oct 06</td>
<td>G12</td>
<td>=SUM(C12:F12)</td>
<td>928</td>
</tr>
<tr>
<td>47</td>
<td>Oct 06</td>
<td>G20</td>
<td>=SUM(C20:F20)</td>
<td>883</td>
</tr>
<tr>
<td>48</td>
<td>Oct 06</td>
<td>G14</td>
<td>=SUM(C14:F14)</td>
<td>793</td>
</tr>
<tr>
<td>49</td>
<td>Oct 06</td>
<td>G15</td>
<td>=SUM(C15:F15)</td>
<td>691</td>
</tr>
<tr>
<td>50</td>
<td>Oct 06</td>
<td>G16</td>
<td>=SUM(C16:F16)</td>
<td>639</td>
</tr>
<tr>
<td>51</td>
<td>Oct 06</td>
<td>G17</td>
<td>=SUM(C17:F17)</td>
<td>593</td>
</tr>
<tr>
<td>52</td>
<td>Oct 06</td>
<td>G18</td>
<td>=SUM(C18:F18)</td>
<td>523</td>
</tr>
<tr>
<td>53</td>
<td>Oct 06</td>
<td>G19</td>
<td>=SUM(C19:F19)</td>
<td>469</td>
</tr>
<tr>
<td>54</td>
<td>Oct 06</td>
<td>G20</td>
<td>=SUM(C20:F20)</td>
<td>460</td>
</tr>
<tr>
<td>55</td>
<td>Oct 06</td>
<td>G21</td>
<td>=SUM(C21:F21)</td>
<td>409</td>
</tr>
<tr>
<td>56</td>
<td>Oct 06</td>
<td>G22</td>
<td>=SUM(C22:F22)-80</td>
<td>300</td>
</tr>
<tr>
<td>57</td>
<td>Oct 06</td>
<td>G23</td>
<td>=SUM(C23:F23)</td>
<td>292</td>
</tr>
<tr>
<td>58</td>
<td>Oct 06</td>
<td>G24</td>
<td>=SUM(C24:F24)</td>
<td>289</td>
</tr>
<tr>
<td>59</td>
<td>Oct 06</td>
<td>G25</td>
<td>=SUM(C25:F25)</td>
<td>217</td>
</tr>
<tr>
<td>60</td>
<td>Oct 06</td>
<td>G26</td>
<td>=SUM(C26:F26)</td>
<td>231</td>
</tr>
<tr>
<td>61</td>
<td>Oct 06</td>
<td>G27</td>
<td>=SUM(C27:F27)</td>
<td>174</td>
</tr>
<tr>
<td>62</td>
<td>Oct 06</td>
<td>G28</td>
<td>=SUM(C28:F28)</td>
<td>126</td>
</tr>
<tr>
<td>63</td>
<td>Oct 06</td>
<td>G29</td>
<td>=SUM(C29:F29)</td>
<td>125</td>
</tr>
<tr>
<td>64</td>
<td>Oct 06</td>
<td>G30</td>
<td>=SUM(C30:F30)</td>
<td>89</td>
</tr>
<tr>
<td>65</td>
<td>Oct 06</td>
<td>G31</td>
<td>=SUM(C31:F31)</td>
<td>81</td>
</tr>
</tbody>
</table>
This is easy enough for twenty formulas, less so for hundreds or thousands.

Compare that with the Operis Analysis Kit (OAK) Review listing and map:

The map shows in red the beginning of each new block of a distinct formula.

Another kind of map is illustrated by the XLTest Distinct Formula listing:

Which gives the colour key for the XLTest Distinct Formula map:

---

ISBN : 978-1-9054045-1-3
Copyright © 2013, European Spreadsheet Risks Interest Group (www.eusprig.org)
COMPARE THE LISTING OF DEFINED NAMES

Inquire misses the hidden names (in this case 'Oct 06'!Royalty, see later):

OAK shows the hidden names and gives useful information on the dimensions of the ranges referred to, to help in consistency checking. Another sheet reports names with overlapping ranges.

XLTest shows the visibility and current value:
WORKBOOK RELATIONSHIP

This is a useful picture of the dependencies in systems of linked workbooks.

This illustration shows the dead links in the Names. It would be useful if clicking on those linked files gave more information, such as where the links were found.

WORKSHEET RELATIONSHIP DIAGRAM
For any moderately complex workbook, this does not provide much enlightenment. I don’t see the value of showing the workbook on the left hand side and all the links going to that, as we already know that they are all in that workbook. It would be clearer to show for example “Intro” on its own unlinked to anything. Also, I would like the arrow to be annotated with information such as how many links there are for one arrow.

**CELL RELATIONSHIP**

This shows the links between individual cells. Because it does not have any concept of ranges of similar formulas, the diagrams rapidly sprawl as the number of copied formulas increases. Here is an image from a simple tax calculation spreadsheet:

![Cell Relationship Diagram](image)

It is therefore probably best suited to simple spreadsheets.
It will try to expand the tree for all the precedent cells in a LOOKUP formula, which could be thousands. This can cause Excel to hang (the window goes frosty and the title bar says "(Not Responding)" for several minutes. Task Manager can show you how it using memory and CPU time:
Eventually it will give you a graph with thousands of nodes:

COMPARE FILES
The Compare Files command lets you see the differences, cell by cell, between two workbooks. Results are colour-coded by the kind of content, such as entered values, formulas, named ranges, and formats. A window shows VBA code changes line by line. Differences between cells are shown in a grid layout, like this:
The "Show Details" button applies to VBA differences and shows a WinDiff output of the code. Clicking "Help" just gets you "We're sorry, but there is no help available for Spreadsheet Compare in this market."

There is a rather obscure term 'SysGen …' in the options that is not explained anywhere in the Help. It only makes a difference to the report after you tick the option 'Include system generated changes in result' in the Options dialog which appears when you start a Compare; so you have to do at least one comparison, even if you do not use the output, to get at the following dialog: (As you can unselect the four SysGen checkboxes in the Options, I don't know why they don't always process it by default as you can always tick or untick the display later.)

By 'system generated changes' they mean formulas that have been automatically changed by Excel as a result of an insertion or deletion. For example, say you have a formula =SUM(E6:E9) and you delete row 9 so the formula now shows =SUM(E6:E8).

With 'SysGen Formulas' unticked, no difference is reported:

When ticked, the difference is highlighted:

'SysGen Names' refers to Excel's internal names such as Print_Area.

**BUG IN THE COMPARE OPTIONS LIST**

If you click the Option heading in the Option checkboxes at the bottom left, the table is sorted alphabetically by Option. But the first checkbox is still "Select All"; that is, the action of the checkboxes does not change from the unsorted list.
The 'Entered Values' comparison is case sensitive, and there does not appear to be an option to make it treat upper and lower case as being the same.

The following page shows a complete picture of the Compare windows. If exported to Excel, the comparison is shown as a table like this:

<table>
<thead>
<tr>
<th>Sheet</th>
<th>Range</th>
<th>Old Value</th>
<th>New Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Staffing Plan</td>
<td></td>
<td></td>
<td></td>
<td>Added Column K.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td></td>
<td></td>
<td></td>
<td>Deleted Row 9.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td></td>
<td></td>
<td></td>
<td>Deleted Row 11.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>I19</td>
<td>=ROUNDC21/I19,-3)/4/1000</td>
<td>=ROUNDC$19/I17/1000,0)/4</td>
<td>Formula Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>J19</td>
<td>=ROUNDC21/J19,-3)/4/1000</td>
<td>=ROUNDC$19/J17/1000,0)/4</td>
<td>Formula Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>C19</td>
<td>350,000 (350000)</td>
<td>400,000 (400000)</td>
<td>Entered Value Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>E19</td>
<td>10 (9.75)</td>
<td>11 (11)</td>
<td>Calculated Value Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>F19</td>
<td>7 (6.75)</td>
<td>8 (7.75)</td>
<td>Calculated Value Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>G19</td>
<td>5 (5.25)</td>
<td>6 (6)</td>
<td>Calculated Value Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>I19</td>
<td>4 (3.5)</td>
<td>4 (4)</td>
<td>Calculated Value Changed.</td>
</tr>
<tr>
<td>Staffing Plan</td>
<td>J19</td>
<td>3 (3)</td>
<td>4 (3.5)</td>
<td>Calculated Value Changed.</td>
</tr>
</tbody>
</table>
For display purposes OAK can insert rows or columns to line up the sheets to match:

<table>
<thead>
<tr>
<th>Range</th>
<th>Cell contents on 1st Sheet</th>
<th>Cell contents on 2nd Sheet</th>
</tr>
</thead>
<tbody>
<tr>
<td>From Row = 9</td>
<td>1 rows inserted</td>
<td>1 rows inserted</td>
</tr>
<tr>
<td>From Row = 11</td>
<td></td>
<td>1 columns inserted</td>
</tr>
<tr>
<td>From Column = K</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K1</td>
<td>Staffing</td>
<td></td>
</tr>
<tr>
<td>K2</td>
<td>Q3</td>
<td></td>
</tr>
<tr>
<td>K3</td>
<td>Year 2</td>
<td></td>
</tr>
<tr>
<td>K6</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>K7</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>K8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B9</td>
<td>Other</td>
<td></td>
</tr>
<tr>
<td>C9</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>E9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>G9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>H9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>J9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>L9</td>
<td>80000</td>
<td></td>
</tr>
<tr>
<td>E10 =SUM(E6:E9)</td>
<td>=SUM(E6:E8)</td>
<td></td>
</tr>
<tr>
<td>F10 =SUM(F6:F9)</td>
<td>=SUM(F6:F8)</td>
<td></td>
</tr>
<tr>
<td>G10 =SUM(G6:G9)</td>
<td>=SUM(G6:G8)</td>
<td></td>
</tr>
<tr>
<td>H10 =SUM(H6:H9)</td>
<td>=SUM(H6:H8)</td>
<td></td>
</tr>
<tr>
<td>I10 =SUM(I6:I9)</td>
<td>=SUM(I6:I8)</td>
<td></td>
</tr>
<tr>
<td>J10 =SUM(J6:J9)</td>
<td>=SUM(J6:J8)</td>
<td></td>
</tr>
<tr>
<td>K10</td>
<td>=SUM(K6:K8)</td>
<td></td>
</tr>
<tr>
<td>K13</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>K14</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>K15</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>K16</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>K17 =SUM(K13:K16)</td>
<td>=K10+K17</td>
<td></td>
</tr>
<tr>
<td>K19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C21</td>
<td>350000</td>
<td>400000</td>
</tr>
<tr>
<td>I21 =ROUND($C$21/I19, -3)/1000</td>
<td>=ROUND($C$21/I19/1000,0,0)/4</td>
<td></td>
</tr>
<tr>
<td>J21 =ROUND($C$21/J19, -3)/1000</td>
<td>=ROUND($C$21/J19/1000,0,0)/4</td>
<td></td>
</tr>
<tr>
<td>K21 =ROUND($C$21/K19/1000,0,0)/4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CLEAN EXCESS CELL Formatting

This says it clears excessive formatting but does not tell you beforehand what will be cleared, nor afterwards what was cleared.

Interactive analysis

This was removed from the final release of the Inquire add-in for Excel 2013, because of stability problems.

By comparison, OAK has a menu to report various items of interest:

A strong feature of OAK is its facility to understand the reconstruct the components of a formula and to progressively simplify it, for example by pruning inactive paths.

Again to compare with another tool, XLTest produces an analysis either interactively or as a batch process:
XLTEST provides a detailed worksheet analysis:

**Worksheet Errors**

No Named ranges created for checks

<table>
<thead>
<tr>
<th>No.</th>
<th>Error Value</th>
<th>Count</th>
<th>Areas</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>#DIV/0!</td>
<td>1</td>
<td>1</td>
<td>B22</td>
</tr>
<tr>
<td>2042</td>
<td>#N/A</td>
<td>1</td>
<td>1</td>
<td>B16</td>
</tr>
<tr>
<td>2029</td>
<td>#NAME?</td>
<td>2</td>
<td>2</td>
<td>B13</td>
</tr>
<tr>
<td>2000</td>
<td>#NULL!</td>
<td>1</td>
<td>1</td>
<td>B25</td>
</tr>
<tr>
<td>2036</td>
<td>#NUM!</td>
<td>2</td>
<td>2</td>
<td>C5</td>
</tr>
<tr>
<td>2023</td>
<td>#REF!</td>
<td>3</td>
<td>3</td>
<td>B17</td>
</tr>
<tr>
<td>2015</td>
<td>#VALUE!</td>
<td>2</td>
<td>2</td>
<td>B15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>Error Check</th>
<th>Count</th>
<th>Areas</th>
<th>Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Error value</td>
<td>12</td>
<td>8</td>
<td>C5</td>
</tr>
<tr>
<td>2</td>
<td>Text two digit year</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Number stored as text</td>
<td>1</td>
<td>1</td>
<td>F20</td>
</tr>
<tr>
<td>4</td>
<td>Inconsistent formula</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Omits cells in region</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Unlocked formula cell</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Refers to empty cell</td>
<td>4</td>
<td>4</td>
<td>B16</td>
</tr>
<tr>
<td>8</td>
<td>List validation error</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Inconsistent list</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Fails data validation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Overflow error</td>
<td>2</td>
<td>2</td>
<td>B5</td>
</tr>
<tr>
<td>12</td>
<td>Number in formula</td>
<td>6</td>
<td>4</td>
<td>E5</td>
</tr>
<tr>
<td>13</td>
<td>Format hides value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Format Font Fill colour</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Conditional Format Font</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Formula too long</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Formula starts with minus</td>
<td>1</td>
<td>1</td>
<td>B5</td>
</tr>
<tr>
<td>18</td>
<td>Formula with double minus</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Numeric text right aligned</td>
<td>1</td>
<td>1</td>
<td>F20</td>
</tr>
<tr>
<td>20</td>
<td>Range Lookup check params</td>
<td>3</td>
<td>3</td>
<td>B15:B17</td>
</tr>
</tbody>
</table>

**Other Checks**

<table>
<thead>
<tr>
<th>Count</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
<td>A1:F30</td>
</tr>
<tr>
<td>64</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>E20</td>
</tr>
<tr>
<td></td>
<td>='F:\DOCS\SCC3\Ex1Demo[EX1DEMO.XLS]Budget08'!N70+'F</td>
</tr>
<tr>
<td>No.</td>
<td>Function</td>
</tr>
<tr>
<td>-----</td>
<td>------------</td>
</tr>
<tr>
<td>1</td>
<td>NOW</td>
</tr>
<tr>
<td>2</td>
<td>DATE</td>
</tr>
<tr>
<td>3</td>
<td>IF</td>
</tr>
<tr>
<td>4</td>
<td>ISERROR</td>
</tr>
<tr>
<td>5</td>
<td>ISERR</td>
</tr>
<tr>
<td>6</td>
<td>Total</td>
</tr>
<tr>
<td>7</td>
<td>VLOOKUP</td>
</tr>
<tr>
<td>8</td>
<td>SUM</td>
</tr>
<tr>
<td>9</td>
<td>LOG</td>
</tr>
<tr>
<td>10</td>
<td>INDIRECT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>Number Formats used</th>
<th>Count</th>
<th>First use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>General</td>
<td>203</td>
<td>A1</td>
</tr>
<tr>
<td>2</td>
<td>m/d/yyyy h:mm</td>
<td>1</td>
<td>B5</td>
</tr>
<tr>
<td>3</td>
<td>m/d/yyyy</td>
<td>3</td>
<td>C5</td>
</tr>
<tr>
<td>4</td>
<td>0.00</td>
<td>2</td>
<td>B7</td>
</tr>
<tr>
<td>5</td>
<td>@</td>
<td>1</td>
<td>D13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>Custom Styles used</th>
<th>Count</th>
<th>First use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>207</td>
<td>A1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comment</th>
<th>Cell note text</th>
<th>Cell Value</th>
<th>Cell Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>B27</td>
<td>POB:</td>
<td></td>
<td>=SUM(IF(ISERROR(B1:B25),1,0))</td>
</tr>
</tbody>
</table>

This array function was entered using Ctrl+Shift+Enter.
CONCLUSION

The Inquire add-in has the great advantage of being included with Excel, rather than a separate purchase. It provides some minimum standard of documentation, although the simplistic formula listing is not useful for large workbooks. Commercial auditing software offers much more and for a modest cost, typically less than 400 US dollars. Some, like OAK, go beyond review facilities and offer development assistance as well.

Patrick O’Beirne
Systems Modelling Ltd, Ireland
Landline +353 5394 22294 mobile:+353 86 835 2233
Email: pob@sysmod.com
Systems Modelling website: http://www.sysmod.com
Blog: http://sysmod.wordpress.com
LinkedIn profile: http://ie.linkedin.com/in/patrickobeirne

REFERENCES

2 http://www.spreadsheetdetective.com/
3 http://www.auditware.co.uk/
4 http://arxiv.org/abs/0805.4236
5 http://arxiv.org/abs/0712.2591
6 http://www.sysmod.com/xltest
7 http://www.operis.com
8 http://www.abb.com/consulting
9 http://www.liquidity.com
10 http://www.clusterseven.com
11 http://www.sarbox-solutions.com
12 http://arxiv.org/abs/0805.4211
13 http://arxiv.org/abs/1111.5002
14 http://arxiv.org/abs/0808.2045
15 http://arxiv.org/abs/0805.1741
A Maintainability Checklist for Spreadsheets

Henk Vlootman & Felienne Hermans
Vlootman Consultancy and Delft University of Technology
info@vlootman.nl - f.f.j.hermans@tudelft.nl

ABSTRACT

Spreadsheets are widely used in industry, because they are flexible and easy to use. Often, they are even used for business-critical applications. It is however difficult for spreadsheet users to correctly assess the maintainability of spreadsheets. Maintainability of spreadsheets is important, since spreadsheets often have a long lifespan, during which they are used by several users. In this paper, we present a checklist aimed at measuring the maintainability of a spreadsheet. This is achieved via asking several questions, and can indicate whether the spreadsheet is safe to use now and in the future. We demonstrate the applicability of our approach on 11 spreadsheets from the EUSES corpus.

1. INTRODUCTION

Spreadsheets are used extensively in business, for all sorts of tasks and purposes. While other assets of companies---like software products and processes---are strongly guarded, spreadsheets are usually not structurally checked. This lack of control contrasts their impact, which can be widespread, as previous studies have shown. For instance, Hall [Hall, 1996] interviewed 106 spreadsheet developers and found that only 7% of the spreadsheets were of low importance and that as much as 39% were of high importance. In a more recent study we found similar results [Hermans, 2011].

However, these spreadsheets are often of dubious quality, as there are no widely accepted spreadsheet design policies and they are hardly ever enforced or checked. While in the past, research has been conducted on spreadsheet metrics [Bregar, 2004] [Hodnigg, 2008], it remains hard to assess the maintainability of a spreadsheet. In this paper we present a checklist that can be used in industry by spreadsheet professionals to measure a single spreadsheet and to compare different spreadsheets with each other in a fair way. This checklist has been developed by the first author of this paper and has been used to assess over 150 models over the past years.

The checklist aims to measure the maintainability of a given spreadsheet, according to ISO/IEC 9126, this includes ease of understanding, adapting and testing the spreadsheet. Note that we only focus on the maintainability of formulas, other constructs like charts, VBA and pivot tables are out of scope.

In this paper, we describe the rationale behind the checklist and put it to the test, by applying it to 11 randomly selected spreadsheets from a well-known test set and having this analysis
done by two different assessors; both authors of this paper.

With this analysis, we want to investigate:

- Whether this checklist is capable of categorizing spreadsheet maintainability
- To what extent the two different assessors obtain similar results
- How well spreadsheets score on the checklist

2. THE RATIONALE BEHIND THE CHECKLIST

The checklist consists of several categories, that each contributes to the checklist in a different way. The categories we use are based on existing work in spreadsheets [Read, 1999], [Prior, 2006] combined with on our personal experiences working with spreadsheets in the field.

It is important to note that the current version of the checklist is neither perfect nor complete. In our opinion a checklist is an appropriate way to measure spreadsheet maintainability and the current version is merely a first attempt that has proven valuable in practice.

2.1 Documentation

For maintenance and evolution purposes, it is very important that documentation exists [Prior, 2006] Documentation matters for several reasons. The most important is the continuity of the model. If the creator of the model is not present anymore, for whatever reason, problems will occur. And as we saw in previous work, models tend to stay in use for several years. Often it happen that there is just one employee can work with a spreadsheet, and problems occur when this employee is on a holiday.

We distinguish two different types of documentation: technical documentation and user documentation.

*Technical documentation* helps to understand the structure of a model. For instance, it explains why certain design elements were chosen and how chains of formulas are connected. The lack of technical documentation is a serious problem.

*User documentation* helps the user to work with the model. Although not as important as technical documentation, missing or faulty in user documentation can cause the model’s users to misuse the model.

2.2 Structure

Previous research [Hermans, 2011], [Hermans, 2012a] shows that a clear structure for an Excel model is essential for the understandability of the model. Therefore, spreadsheet structure it is a central category of the checklist. We distinguish three elements of structure: the categorization of worksheets, the naming of worksheets and the separation of calculations and input.
2.3 Management

Within control, there are two elements for the checklist. The first one is changeability, defined by ISO/IEC9126 as the ease with which we can modify a spreadsheet. Therefore we look at whether a model is easy to understand and whether modifications might influence the results.

The second aspect of the management category is ease of use. When variables and formulas are easy to find, it will be easier to maintain the model.

These two aspects result in questions on the location and visibility of the input cells and the use of named ranges.

2.4 Safety

Safety is defined as the robustness of the model: can small changes have large consequences or have control measures against this been implemented? For this category we look at the normalization of the variables. With this we mean that if how constants in the model (i.e. Value Added Tax) how are stored on the spreadsheet and how to formulas utilizing this constant reference to it. Furthermore, we look at the use of user selection choices. Are control elements used and are inputs validated?

2.5 Formatting

Good formatting can support the user, this is the reason that we added this as a category is our check list. In this category, we look at how cells are formatted and whether the creator of the model added comment boxes or explanation.

2.6 Skills

In the final category, we aim to measure the skills of the creator of the spreadsheet. The more complex Excel elements are used, the higher the skills of the user is ranked. This method is based on the previous work of [Hole, 2009]

3. THE CHECKLIST

This section contains the questions of the checklist. This checklist can also be found online: http://www.vlootman.nl/index.php?option=com_weblinks&view=category&id=12:bestanden&Itemid=30&lang=en

3.1 Weighting the questions

In Section 3.2, the weightings for each category are given in brackets. This weighing has been chosen based on personal experience of working with the checklist in industry. However, we do not consider the specific weights given to a question as a part of the checklist model. Rather, we open up the possibility for users to calibrate these weights to represent their own assertion of the value of each question. As long as the weights are kept
constant while measuring multiple spreadsheets, the checklist is suited to compare different
spreadsheets.

3.2 The questions

Documentation

1) Is there any technical description available? 20
2) Is there any user description available? 15

Structure

3) Are the sheets grouped by function? 20
4) Is the naming of the worksheets understandable? 5
5) Are calculations separated from input? 15

Management

6) Are all variables placed together? 10
7) Is there a clear distinction between input and output? 10
8) The output is compact and clear? 10
9) Are valid Excel ranges used? 10
10) Are the input cells logically grouped? 10

Safety

11) Is normalization used on the variables? 20
12) In which way will the user selection be processed? 15

Figure 1 A screenshot of the check list in Excel
**Formatting**

13) Are the input cells formatted consequently? 10  
14) Are the output cells formatted consequently? 10  
15) Are the other cells formatted consequently? 10  
16) In which way will the user be supported? 15  

**Skills**

17) Is array functionality used in the model? 20  
18) Does the model support windows for the output? 15  
19) Are names used in the model? 15  
20) Are names separated in categories? 10  
21) Are the names consistently composed? 5  
22) Are the names consistently used? 5  
23) Are (complex) single sided functions used in the model? 10  
24) Does the model have nested functions? 15  
25) Does the model have links towards other cells? 10  
26) Does the model have absolute links or names towards other cells? 5  

**4 EVALUATION**

**4.1 Goal and setup**

To evaluate our checklist, we performed an empirical study in which both authors checked 11 spreadsheets from a well-known test set of spreadsheets.

With these experiments we want to validate that

- The checklist distinguishes spreadsheets from each other  
- The results can be used to assess a spreadsheet  
- The results can be used to improve a spreadsheet  

**4.2 Data**

As a source for the experimental phase we used spreadsheets from the EUSES Spreadsheet Corpus [Fisher, 2005]. This corpus consists of around 5000 spreadsheets, divided over 11 categories---ranging from educational to financial---and has been used by several researchers to evaluate algorithms on spreadsheets, among which [Hermans, 2010] and [Abraham, 2006]. We randomly selected one spreadsheet from each of the 11 categories. All 11 spreadsheets can be downloaded from our research page (http://swerl.tudelft.nl/bin/view/FelienneHermans/).
### 4.3 Results

Table 1 Results of the check list on the EUSES corpus by the first author. Highlighted cells are questions where authors disagreed

<table>
<thead>
<tr>
<th>Q</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Documentation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>User sheets</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Structure</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>N/A</td>
<td>Yes</td>
<td>N/A</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Safety</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Controls</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Formatting</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>14</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>15</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>16</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>In cells</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
</tr>
<tr>
<td><strong>Skills</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>18</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>19</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>21</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>22</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>23</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>24</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>25</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>26</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 2 Scores corresponding to the answers found in Table 1, as calculated by the current weighting in the checklist

<table>
<thead>
<tr>
<th>Model</th>
<th>Doc.</th>
<th>Structure</th>
<th>Man.</th>
<th>Safety</th>
<th>Format</th>
<th>Skills</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Posey_Q</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
<td>1.3</td>
</tr>
<tr>
<td>FinalBudget</td>
<td>0</td>
<td>1.3</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
<td>1.4</td>
</tr>
<tr>
<td>CHOFAS</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>3.6</td>
<td>4.1</td>
</tr>
<tr>
<td>FinFun</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>3.6</td>
<td>2.5</td>
</tr>
<tr>
<td>karen-cs101gradebookSp98</td>
<td>0</td>
<td>1.3</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>3.6</td>
<td>2.1</td>
</tr>
<tr>
<td>9-Grade</td>
<td>0</td>
<td>1.3</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
<td>1.7</td>
</tr>
<tr>
<td>Solutions_week_3</td>
<td>0</td>
<td>1.3</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>2.3</td>
<td>1.9</td>
</tr>
<tr>
<td>grain_inventory_market#A85C5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>6.7</td>
<td>0.9</td>
<td>2.2</td>
</tr>
<tr>
<td>Equity2</td>
<td>0</td>
<td>1.3</td>
<td>10</td>
<td>4.3</td>
<td>0</td>
<td>5.5</td>
<td>4.1</td>
</tr>
<tr>
<td>occupancy_schedules_m#A82E1</td>
<td>4.3</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>1.4</td>
<td>2.2</td>
</tr>
<tr>
<td>BTVSCCG Inventory</td>
<td>0</td>
<td>1.3</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
<td>1.7</td>
</tr>
</tbody>
</table>

Difference between assessors

The first observation that can be made in Table 1, is the fact that the results of both authors are not entirely consistent. Even in the more technical categories, like skills, we see differences, like on the issue of nested functions (Q24) Here the discrepancy is whether a formula F14*(1-F16) is or is not a nested function. Author 1 only considers built-in functions (like SUM) to count as nested, where author 2 also counted infix operations like as functions. These results indicate that more guidance in answering the questions would improve consistent scoring.

Documentation

The second fact that stands out is the fact that all but 1 spreadsheets miss both technical and users documentation. This might be because for these spreadsheets documentation exists outside of the spreadsheets. This would not have been gathered by the creators of the EUSES, which only searches for spreadsheets. However, we do not believe this is the case for all of those spreadsheets, we think that these results show that the general state of documentation is poor in spreadsheets.

Structure

Within the structure category, it is interesting to see that in one of the cases, the assessor differ in their opinion on clear naming of worksheets. In this case (file 8) there are some worksheets with standard names (sheet 13, sheet 14, etc.) and some other where a name has been changed by the user. In the majority of the cases (7 spreadsheets) clear names are used. However, worksheets are never clearly designated for input of output, in all models this is mixed over worksheets.
Management

Management is the category that overall scores the highest of all, with scores from 4 to the maximum value of 10. This is mainly due to the fact that the spreadsheets clearly separate input and output quite well, that output is grouped and that valid ranges are used. Probably, the management category scores high, as spreadsheet users have to be able to find their output easy, so arranging this wisely is something they do, as opposed to building in safety constraints, which is not needed for their daily tasks.

Safety

In the safety category too, the scores are low. This means that often, hardcoded values are used. In only one of the spreadsheets, user controls were introduced to prevent erroneous and validations were not used in one single test spreadsheet.

Formatting

In the formatting category too, there is only one spreadsheet scoring over 0, this is file 7, where input and output cells are consistently colored and the user gets help with their input too. In this category too, there are some differences between the assessors, mainly on the question whether non-input and output cells, i.e. labels, are consistently formatted.

Skills

The skills category is a diverse category, with scores ranging from 0.9 to 5.5, as can be seen in Table 2. As mentioned before, there are differences in judgement of skill between authors as the nature of this work is highly technical. Further, the results might be biased since 4 questions in this category concern naming. A model lacking any names will score very low on skills because of this. In retrospect, this focus might be a bit heavy.

4.4 Conclusion

The aim of this study was to validate that a spreadsheet checklist can help users to compare spreadsheets in an objective way.

Based on the results we have presented above, we can conclude that randomly chosen spreadsheets from the EUSES corpus differ on many aspects of the checklist, confirming the idea that this checklist helps to distinguish between spreadsheets. Furthermore we observe that although there are some differences between the two assessors, overall the results are consistent enough to conclude that the checklist can be used to compare spreadsheets in a fair way, even if assessed by different people. Finally, we see that the spreadsheets under observation in this study do not score high, which is consistent with experiences from the first author while using this checklist in practice: many spreadsheets could be greatly improved.
5. DISCUSSION

The current version of the checklist is able to measure the maintainability of a spreadsheet, in other words, to measure how easy to understand and change the spreadsheet will be for future users. However, there are some limitations. For instance, in the current version of the checklist only focuses on formulas and hence graphs, charts and pivot tables are not taken into account. The following subsections describe other avenues for improvement.

5.1 Automating the checklist

Some of the categories of the check list, such as the skills category can be easily automated. This would reduce the work load for the assessor. We do however prefer a partly automated system over a fully automated, as some categories like documentation can never be automated and also, using a fully automated system might make it easy for spreadsheet users to game the metrics.

5.2 Taking smells into account

In previous research [Hermans, 2012a; Hermans, 2012b] we have worked on the detection of smells in spreadsheets. In that work we have found the ‘smelly’ formulas can diminish maintainability of a spreadsheet. Hence, it would be interesting to add a category specific for ‘smells’. Also, spreadsheet complexity metrics, like the ones suggested in [Bregar, 2004] or [Hodnigg, 2008] could make our assessment method more precise.

5.3 Suggesting improvements

Many of the questions of the check list are actually guidelines in disguise, such as the use of naming or a clear difference in lay-out between input and output cells. Having a low mark on one of these questions, might be caused by the fact that a spreadsheet user in unaware of these underlying guidelines. Explaining these guidelines and even giving concrete help on how to improve-such as ‘move these and these formula cells to worksheet ‘Sheet1’- could help users improve their score.

5.4 Making ranges continuous

In the current version of the checklist, many questions are dichotomous yes/no and in when using the checklist, we found that sometimes an option in between would make assessing more fair. Another possibility would be to add an option for ‘not applicable’.

6 CONCLUDING REMARKS

The aim of this paper is to develop a set of questions that indicate the maintainability of a given spreadsheet. To that end we have suggested such a checklist and applied in on 11 spreadsheets from the EUSES corpus. We conclude that, while this paper represents a first attempt, our checklist is capable of differentiating between different spreadsheets and identifying areas for improvement.
The key contributions of this paper are as follows:

- A comprehensive checklist to assess spreadsheet maintainability
- An empirical evaluation investigating the usefulness of that checklist

The current research gives rise to several directions for future work. Firstly, it would be very interesting to perform a more thorough empirical evaluation of this checklist to validate whether it can prevent errors or problems in practice. Furthermore, it would be interesting to couple low scores to concrete actions, which help users to improve their spreadsheets. This could be combined with the above proposed automation of the checklist.

7 REFERENCES


Introducing Morphit, a new type of spreadsheet technology

Ted Hawkins Ph.D., Andrew Lemon Ph.D., Alec Gibson B.Sc. (Hons) (Dunelm)
The Edge Software Consultancy Ltd.,
77 Walnut Tree Close, Guildford, Surrey. GU1 4UH UK
thawkins@edge-ka.com

ABSTRACT

This paper describes a new type of spreadsheet which mitigates the errors caused by incorrect range referencing in formulae. This spreadsheet is composed of structured worksheets called tables which contain a hierarchical organization of fields. Formulae are defined at the field-level removing the need for positional references. In addition, relationships can be defined between fields in tables, allowing data to be modeled rather than simply processed and providing a re-usable framework for authoring spreadsheets. We shall describe the key features of tables with an emphasis on error detection and avoidance.

1. INTRODUCTION

Despite their utility it is clear that in many cases current spreadsheets are not fit for purpose in today’s business environment [Saadat 2008]. Inadvertent errors are too easily made [Chadwick 2008a, Dunn 2010, Panko 2008a], and auditing spreadsheets is so difficult that they have been identified as a significant factor in the financial crash of 2008 [Croll 2009] as well as a number of fraud cases [Butler 2009, Mittermeir et al. 2008, Panko and Ordway 2008, Saadat 2008]. Cell error rates have been estimated at between 2% and 5%, with approximately 94% of spreadsheets containing errors [Panko 2008a, Stephen G. Powell, Kenneth R. Baker 2009].

The reliance on explicit positional references in formulae is the root cause of two major sources of error - physical area related errors [Ayalew and Mittermeir 2008] and semantic and extendibility errors caused by poor layout [Przasnyski et al. 2011]. Consider the simple formula:

\[ \text{SUM(A1:B2;A12:B13)} \]

This formula is intrinsically difficult to read and understand which data the ranges are referring to, even more so were the references to go across worksheets. Positional formulae are often fragile with respect to worksheet modifications, errors can easily be introduced as new cells are inserted between the cells in a range, or formulae are incorrectly over-written as a result of copying absolute cell references. Auditing spreadsheets is a difficult and time consuming process since cell-by cell inspection is the only guaranteed method of ensuring compliance [Panko 2008b]. While we acknowledge that there are tools available to assist with the auditing process, it is our experience from the pharmaceutical industry, that very few people use them. Rather than trying to detect
errors every time a spreadsheet is modified, we believe it is better to address the underlying flaws in the spreadsheet concept that allow these errors to be introduced in the first place.

Many of the problems with spreadsheets arise from the attempt to represent structured data in what is effectively an unstructured environment. Any structure to the data in a spreadsheet is implicit from a combination of user defined layout and presentation, rather than being inherent in the design.

For these reasons we have designed Morphit [Edge 2013] (www.edge-ka.com/products/morphit), a standalone application that, while retaining many of the features of traditional spreadsheets, incorporates the idea of explicitly defined data structure. In this paper we shall illustrate how this works in practice.

2. TABLES

Morphit contains two principal page types – sheets and tables. Sheets are almost identical to a traditional spreadsheet and used for presentation of static reports, however most work is achieved using tables. The quickest way to understand tables is to look at an example.

The screenshot above shows a Morphit table containing sales data for a pet shop. The table is defined as a hierarchy, with the top level relating to years, the second level relating to months, and the bottom level containing sales by code. New rows can be added at any level to expand the years, months, and sales codes.

Columns in a table are called fields. Of particular interest in this table are the two fields ‘Monthly Total’ and ‘Yearly Total’. Both of these fields are calculated using exactly the same formula – ‘=SUM(Total)’. Morphit will calculate the result of this formula using only the values of ‘Total’ which are beneath the formula-containing cell in the hierarchy.
This already helps remove some of the common causes of errors in spreadsheets. The formula is only entered once for the ‘Monthly Total’ and ‘Yearly Total’ fields and it is propagated to any new cells in those fields. This removes the requirement to enter formulae in each cell, so eliminating many physical area related errors caused by transcription [Saadat 2008]. In addition, should any new rows be added at the Sales level, they will automatically be included in the calculations. This helps overcome the ‘physical area mix up problem’ [Ayalew and Mittermeir 2008].

The overall effect of this is that while the formulae should still initially be validated, the number of formulae to check has been reduced to an achievable level. In addition there is no longer any need to revalidate the table when adding new rows.

**Using Multiple Tables**

It is rare to find a problem simple enough that it can be modeled using a single table, or even a set of unconnected tables. Morphit contains powerful techniques that allow you to join several tables together in order to model complex business problems.

**Field Borrowing**

The first technique is called Borrowing. This is illustrated using the following example.

In this example we have decided that we want to take the sales data from the previous example and summarize it by sales code and year, visualizing the results on a line graph.

This table is constructed entirely from data sourced from the previous sales table. First, we borrowed in the ‘Sales Code’ field. This brings across all unique values in the ‘Sales
Code’ field in the Sales table. Next, we borrowed in the ‘Year’ field. This brings in all years where there are sales data matching the sales code, for example in 2011 no wide-mouthed frogs were sold, and so that year is not present.

The ‘Total’ field is then calculated as in the previous example. The formula is almost identical, however this time it references the field value explicitly from the ‘Sales’ table – ‘=SUM(Sales!Total)’. The ‘Total’ formula only makes use of those rows from the Sales table where both the ‘Sales Code’ and ‘Year’ values match reducing the scope of the formula only to related data.

If a new sales code is added to the Sales table, this table will automatically update adding a new row (and a new line in the chart). Conversely, removing all the sales data for a given sales code will remove all related data in the ‘Sales Summary’ table and chart. None of these operations require re-validation of the spreadsheet.

The technique of borrowing maximizes the use of data, removing the requirement for redundant data entry, a major source of qualitative errors [Stephen G. Powell, Kenneth R. Baker 2009].

**Linking Fields Between Tables**

Borrowing allows you to ‘pull’ entire fields across from one table to another. However, sometimes you might just want to lookup values from another table based on the value in a particular field. This is illustrated by the following example with two tables.
The first of these two tables contains a list of prices and applicable VAT rates for the animals in stock. The second table contains details of sales invoices. The ‘Item’ field in the ‘Invoices’ table is a drop down list containing all the animals in the ‘Animals’ table. We have then created a link between the ‘Item’ field in this table and the ‘Animal’ field in the ‘Animals’ table. This reduces the scope of any references to the fields in the ‘Animals’ table to only those rows where the ‘Item’ field value matches the ‘Animal’ field value. This is referred to as linking through keys. Therefore the first line of the ‘Invoices’ table identifies only one goldfish. In the ‘Net’ field the price is calculated using a simple formula ‘=Quantity*Animals!Price’. The ‘VAT’ field is also calculated using a similar process referencing the correct VAT rate for the selected animal.

A table may be linked to more than one field in another table, and to more than one other table. While in this example we have linked a field containing a drop down list, we could just as well have linked a simple text field.

Linking through the use of keys removes the issues of positional cell references. Formulae are scoped by the records matching the key, in a similar way to the scoping by groups. Linking is used to match imported data to existing data in tables removing the requirement to control the order of data and mitigating data structure errors [Stephen G. Powell, Kenneth R. Baker 2009].

3. CONCLUSIONS

The use of tables and field level formulae can significantly minimize physical area related errors, and enable rigorous formula auditing. The use of hierarchical tables and techniques such as linking and borrowing facilitate an object-orientated approach to spreadsheet design. Data can be modeled rather than just processed as happens with traditional spreadsheets. The class of object is defined by a table, fields represent the properties of the class and each row represents an object or instance of the class. Relationships between objects can be described that access data without introducing de-normalization (replicating data redundantly across tables). Taking this modeling approach delivers three major benefits:

1. Improved comprehension and readability.
2. Increased re-use.
3. Reduced errors.
Table-based spreadsheets are far more re-usable than traditional spreadsheets, operating under a wide variety of conditions without resorting to complex redundancy in order to incorporate all potential variations.

The robustness and inherent flexibility of spreadsheets written in this technology has already been recognized within the pharmaceutical industry, reducing the effort required to accommodate the high degree of variability within biological systems.

Whilst tables cannot address all spreadsheet problems, such technology should reduce the risk of error by reducing complexity and enabling rigorous auditing.


REFERENCES


Spatial Modelling Techniques in Microsoft Excel

Stephen Allen
ACBA (UK) LTD
steve@acba.co.uk

ABSTRACT

We begin by considering the expectations of the creators of VisiCalc, the first spreadsheet. The emphasis is on the nature of the spreadsheet grid. The grid is taken as a presentational method for showing a solution to a Sudoku puzzle. We consider methods or approaches for the solution.

We look at the relationship between this model and academic papers on the methods for describing and categorising end-user models generally. We consider whether the type of model described here should be categorised separately. The complexity of the model is reviewed in the context of commendations to minimise overly sophisticated presentational constructs and formulae.

1. BACKGROUND

Nature and Role of a Spreadsheet

The philosophical question of the core nature of a spreadsheet is not prominent in the literature. Its practical uses are so obvious that we have just got on and used it – for good or ill. However, Bricklin and Frankston, the spreadsheet creators [Power, 2004], did not consider it primarily as an accountancy program: as was the case with the first spreadsheet-type concepts built for mainframe computers. In particular, Frankston stated in an email to Power:

"The grid provided the simplifying structure that made it a spreadsheet as opposed to a more general surface."

The importance of the grid, as opposed to a table or similar database like structure, is huge. It allows spatial freedom with the placement of the grid and conceptual freedom over the nature and contents of the grid (e.g. numbers, text, formulae, images, colours etc.)

The grid structure is often obscured by other elements. There is a great deal of redundancy between the Microsoft business programs [Cook, 2010]. Primarily, Word is for word processing, Excel is the spreadsheet, Access is the database etc. But Excel has database features, Word has spreadsheet features, etc.

The emphasis on database functions within spreadsheets has obscured the functional elements of the grid style paradigm. This paper contributes to redressing the balance, by:

- Selecting a model that employs a grid structure to determine the relationship between individual identities
- Using numeric identities rather than number values, thus forcing a different programmatic approach to the management of the model
We selected the Sudoku puzzle as our experimental model. This model has two additional benefits:

- It involves a relatively complex grid structure where columns and rows and sub-grids are superimposed on top of one another
- We could create a modelling approach that allowed the user to select his own personal pathway for solving the puzzle

1.1. Sudoku as a Model

The objective of the puzzle is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 sub-grids that compose the grid (also called "boxes", "blocks", "regions", or "sub-squares") contains all of the digits from 1 to 9. The puzzle setter provides a partially completed grid, which typically has a unique solution.

A Sudoku grid, like a spreadsheet, is a two dimensional object. Accordingly it is easy to present the puzzle to the user in this medium and identify each of the rows and columns (Figure 1).

![Figure 1 – A Sudoku puzzle from The Guardian, 18 February 2012](image)

In Sudoku the numbers simply represent identities. The true purpose of the puzzle was to follow a series of very specific and relatively simple logic requirements associated with the individual identity in relation to the spatial elements of the grid.

The Sudoku puzzle employs overlapping ranges. It stipulates logical relationships based entirely on the overlaps such that it can be regarded as a three dimensional model in a two dimensional space. The capacity of spreadsheets to cope with overlapping ranges [Bricklin, 1999] was a major advantage and is fully utilised in this context.

1.2. The Structure of the Paper

Overall, this approach to solving Sudoku puzzles is fairly extended and complex. For a more comprehensive history and detailed description see [Allen, 2012a]. The spreadsheet styles and constructs necessary for the development of this conceptual overview will be divided over two papers. This first will deal with the methods and approaches adopted for the analysis of individual cells within the grid. A second paper will consider methods and approaches for changing the focus of cells under examination. These measures are designed to demonstrate the specific pathway through which all the Sudoku cells are evaluated.
This paper will consider the styles of analytical approach. The detail (Section 2) considers the essential decision making part of the design structures. We examine the relationship between the structure, the potential areas at risk of error and the control mechanisms adopted to manage those risks.

We take the opportunity here to consider the models in relation to previous academic papers that concentrate on the structural description of spreadsheets (Section 3) and draw tentative conclusions. The Sudoku model looks simple, but the overlapping grids hide an inherent complexity. We discuss this complexity in the light of Bregar’s [2004] paper.

2. THE MAIN ANALYTICAL MODELS

2.1. Introduction

We need two primary templates because there are two major methodologies for solving a Sudoku puzzle. One seeks to eliminate potential solutions for a particular cell until the player is left with only one. The other considers the relationships for individual identities within a group of three related columns or rows and their associated sub-grids. We also introduce a third template, but this depends on the output from the template that seeks to eliminate potential solutions. It is of lesser importance than the primary templates.

The basis for both the primary modelling templates was that the user would make a decision about the value of an individual Sudoku cell and demonstrate/record the rationale for that decision.

In each case we describe the primary decision and the major controls that the model employs to stop wholly irrational decisions that would wreck the model. We sought to draw a balance between decisions that were incorrect but were acceptable within the context of the model and those that would undermine the integrity of the model as a whole.

2.2. Analysis by Exclusion

The theory of this approach is that the user selects a cell within the puzzle that has yet to be solved. The role of the user is to eliminate all identities bar one by reference to their pre-existence in any one of the three dimensions directly related to that cell. In this context, the system automatically evaluates which numeric identities have already been employed within the row dimension, leaving the user to demonstrate the existence of identities in the other two dimensions.

In Figure 2, the cells shaded in background green represent the user input – the columns headed “Col” and “Grid”. The content of each cell, within this range, is restricted to 9 potential references which refer to an individual cell position within the Sudoku grid. In effect, the user is expressing the view that the cell under consideration cannot contain this identity because it already exists elsewhere within a related dimension. The value that a user employs is designed to show exactly where the identity is located within that related dimension.

We emphasise that this is a user decision, not an automatic one. There remain many possible reasons why the user decision can be incorrect. The point of the model is to record the rationale, so that a reviewer can validate it. This is a crucial element in the control methodology, which automates certain processes (e.g. posting a value to a Sudoku cell). It is the user’s logic that we record. The system evaluates the result on the basis of that logic.
Once the player has posted his analysis of the review of the column and sub-grid dimensions the evaluation of the results is automated. The process is both stepwise and visible (Figure 2). "AvailNow" is (effectively) a Boolean formula and shows whether the number identity for the cell is still available. The cell, two positions to the right, specifies the numeric identity. The group of 4 cells at the bottom of the analysis grid summarise the user's evaluation inputs.

The user input to the analytical process is strictly limited, but one of the positive impacts of the process is to expose clearly both the procedures and logic of generating a conclusion.

2.3. Location Analysis for a Specific Value

Since the three dimensions (row, column and sub-grid) are superimposed on top of one another, when a user selects a row/column/sub-group triplet, he should find a single identity repeated three times such that it is present once in each of the three dimensions. This characteristic can be employed for analytical purposes, where two of the locations of an individual identity are already known. In such a case the potential location for the third member of the group is limited to three identifiable cells.

The overall analysis (whether automated or through user input) seeks to eliminate two of the three potential locations, either because it is already solved or because the identity already exists in an associated dimension of the individual cell. The user input is limited to identifying those cells that can be excluded as a potential solution because the identifier is present in a related dimension (Figure 4)
Cell R44 (Figure 4) counts the number of locations excluded and determines the nature/presentation of the final result. The most useful result solves the value of a single cell. Once again the system is designed to record the user’s logic and interprets the result automatically.

2.4. Mutual Exclusion Analysis

This analysis depends on the ‘1 of 2’ output from “Analysis by Exclusion” process (Section 2.2). It relies on the concept that if, within a specified dimension, two Sudoku cells have identical values for their ‘1 of 2’ output then BOTH the identities must be associated with those two cells exclusively. Accordingly, if a third cell claims one of the original two values as an alternative, we can assert with certainty that the value is not available because of the mutual exclusivity claimed by the original two cells. This is better illustrated by a diagram (Figure 5).

Figure 4 – User input for location analysis

<p>| | | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>6</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>42</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5 shows the before and after effects of the Mutual Exclusion Analysis. The only user input is to identify the dimension for examination – in this case Column 8.

The worksheet automatically notes that the first and third cells in this column contain the identical values ‘2, 8’. These two integer identities are therefore deemed unavailable to the other cells within the Column 8 dimension. Nevertheless, the sixth cell in this column...
claims potential alternatives of ‘2, 6’. Since the identity ‘2’ is claimed on a mutually exclusive basis, by the first two cells the system knows that the result of that cell must be the identity ‘6’. It is illustrated accordingly in bold green.

3. SPREADSHEET STRUCTURES AND RISKS

3.1. Methods of Building a Spreadsheet

In their 2004 paper, Grossman and Ozluk stated that "When working with a spreadsheet, the developer makes a series of choices about what to do and how to do it". Such choices are personal to the developer. This reflects the development approach adopted for the construction of the Sudoku analysis. The layout and methods of analysis of this model are very unusual. This stems from the facts that we are using identities rather than numeric values as the core input to the model and that the model’s complexity derives from the overlapping nature of the ranges. The methodology employed is almost equivalent to a mathematician’s approach to solving simultaneous equations.

We considered it a worthwhile exercise to compare our approach to the construction of the model and its various elements to academic descriptions of the processes that should be considered. We selected two significant papers [Grossman and Ozluk, 2004] and [Ko et al., 2011]. The later paper considered the approaches to the creation of all types of software by end-users.

The review [Allen, 2013] displayed an unexpectedly good fit between the methods adopted for the construction of the Sudoku model and the academic descriptions. Accordingly, the question arises as to whether the model conforms to expected standards of construction or not. If not, should a new standard be created?

Clearly the structures and methods of construction employed for this model do not conform to the standards that we would apply to the financial and actuarial models. These models form the bedrock of the types of spreadsheets investigated by academia. This lack of conformity is not surprising since the core of spreadsheets under investigation by academics is fundamentally numeric in character rather than spatial.

Spreadsheets that adopt this spatial mode of analysis are rare. There is an example concerning the mapping of the ecology of a national park in the Northern Territories of Australia [McMahon et al., 2010] and other approaches to solving Sudoku puzzles in Excel – for example [Weiss & Rasmussen, 2007]. Also, ACBA (UK) LTD developed a “spatially aware” version of a daybook for its commercial accounting system [ACBA, 2000]. These examples (including the current model) number only four amongst the huge numbers (millions?) of spreadsheets created since their introduction in the 1980s. It is open to argument therefore whether this style of spreadsheet construction and approach deserves its own separate classification.

3.2. Complexity in Spreadsheets

In his 2004 paper Bregar discussed potential metrics for evaluating the complexity of spreadsheets. He stated that:

"It has been proven that the complexity of a model or a particular formula represents an important factor to be considered in the process of spreadsheet development, because a
complex spreadsheet makes error finding difficult and because errors come in relation with cells that have a high potential for faults.”

He examined the impact of IF statements and LOOKUP tables on the reliability of models. He concluded that the conditionality of these constructs, which allowed some paths to be followed and others not, hugely increased the complexity of a spreadsheet from the perspective of a possible metrical analysis.

We examine these concepts in relation to the Sudoku model. Lists of all the formulae employed within the model are available from the ACBA (UK) LTD website [Allen, 2012].

Relationship between the design requirement and complexity

Some of the formulae within this model employ complex constructs. The underlying question is whether these constructs are necessary to the model. That in turn depends on both the nature of the model and the nature of the question that the user is seeking to answer.

In this context the model looks relatively simple, but the logical relationships between the internal areas of the model are complex. The model as described here is designed to present this complexity such that a reviewer can evaluate each step in the process.

Since, from a presentational viewpoint, the answer is a representation of the pathway rather than merely the final solution, we needed to consider how much control the design of the system should impose upon the user. This model is designed to demonstrate both the misuse of logic that generates an error and also the correct use of logic. This might be regarded as the human connection.

In order to proceed with the construction of a usable model we sought to limit the potential range of user created errors – see the green coloured input cells in Figures 2, 4 and 5 – which all employ drop down lists.

The majority of the calculations/assessments, subsequent to this user input, remain open to inspection. They are controlled by formulae and/or programmed relationships. This leaves open the question of whether the design of these programmed relationships is accurate. Overall, there is no substantive evidence to suggest we have failed in this programmed approach.

This approach to construction generates a very different perception of the nature and control of a relatively complex model from that proposed by Bregar. The emphasis here is on an open presentation of the complex logic so as to encourage understanding, rather than limiting complexity so as to avoid errors.

Formula Complexity and Context

The formula complexity within this environment seeks to ensure that ALL the processes necessary for completing the analysis and creating a new version of the puzzle are indeed complete. In a form-based environment (e.g. a database) this can be processed progressively, whereas within a worksheet one has to create artificial controls. Part of this artifice is to deliberately generate an error (or something that looks like an error).
The following formula is extracted from the bottom right hand cell in Figure 2. It is characterised by the nested IF statements that Bregar calls into question and also includes the artificial error discussed above.

=IF(AND(RowNo<>"",F37=""),"#Error",IF(OR(F37="Pre-Set",F37="Solved",F37="1 of 2"),INDIRECT("L" & (RowNo+7)),IF(F51="Solved",SUM(H40:H48),IF(F51="1 of 2",MID(H49,1,1)&", ",&MID(H49,2,1),"."))))

It may be instructive to consider this formula in detail. We need to consider why it has been encapsulated into a single cell and whether we regard this as suitable given the number of nested levels that it incorporates.

**Table 1 – Deconstructing a complex Excel formula**

<table>
<thead>
<tr>
<th>Nested IF Statement</th>
<th>Result Type</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF(AND(RowNo&lt;&gt;&quot;&quot;,F37=&quot;&quot;),&quot;#Error&quot;,</td>
<td>Fixed String</td>
<td>Validates that the Sudoku cell under investigation has been identified and prevents any failure to do so going unnoticed</td>
</tr>
<tr>
<td>IF(OR(F37=&quot;Pre-Set&quot;,F37=&quot;Solved&quot;,F37=&quot;1 of 2&quot;),INDIRECT(&quot;L&quot; &amp; (RowNo+7)),</td>
<td>Numeric Identity</td>
<td>Checks whether the cell has been already either Pre-set or Solved and delivers the identity</td>
</tr>
<tr>
<td>IF(F51=&quot;Solved&quot;,SUM(H40:H48),</td>
<td>Number</td>
<td>Adds all the values in range H40:H48 but since only one value has been posted it will be correct</td>
</tr>
<tr>
<td>IF(F51=&quot;1 of 2&quot;,MID(H49,1,1)&amp;&quot;, &quot;,&amp;MID(H49,2,1),&quot;.&quot;))))</td>
<td>Variable string</td>
<td>Generates a string of which two identities could form the result</td>
</tr>
</tbody>
</table>

This creation of a deliberate error is designed to prevent such mistakes carrying forward unnoticed. The other four nested elements of formula are also designed to be carried forward into a revised puzzle grid. In these cases they are all potentially valid responses to the analysis. It is very unusual to permit such a mixture of data types within a single cell. In the context discussed here, however, it provides the range of values needed to continue working with the Sudoku puzzle.

**4. CONCLUSION**

Through the medium of the Sudoku model we can visualize approaches to spreadsheet construction which are based on spatial relationships rather than numeric analysis. This model allows us to investigate a range of control measures to manage errors and risks that are not present in conventional number-based models. The model is designed to illuminate errors in logic rather than eliminate them.
We have considered two of the academic methods for describing or categorising spreadsheets (or similar end-user constructs). The methods of construction match, but the end product looks and feels very different to a standard spreadsheet. The question of whether there should be a new category for this style of spreadsheet construct has been posed but left unresolved.

We considered both the use of open spreadsheet structures and the associated complexity in formulae that this appeared to generate. We believe that the structural requirement of any model is very much dependent on its purpose. This in turn influences the complexity of the formulae and relationships employed. Simple constructs are valuable but not a universal panacea.

BIBLIOGRAPHIC REFERENCES


Ko, Andrew; Beckwith, Laura; Abraham, Robin (2011) "The state of the art in end-user software engineering", ACM Computing Surveys Vol. 43 No. 3 pp 1–44


This volume contains the proceedings of the fourteenth EuSpRIG conference on ‘Spreadsheet Risk Management’ held in July 2013 at the University of Greenwich, London, England.

The objective of this conference is to promote discussion and co-operation amongst those concerned with authorising, auditing or developing spreadsheet models and by so doing, improve the reliability and integrity of information portrayed in spreadsheet models.

The papers cover a broad spectrum of practical experience and research.

Front cover cartoon courtesy of Jocelyn Ireson-Paine. Screen shot from the Spreadsheet Inquire component of Excel 2013, courtesy Patrick O’Beirne.

<table>
<thead>
<tr>
<th>EuSpRIG 2013 Conference Sponsor:</th>
<th>CIMCON Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>EuSpRIG is Supported by:</td>
<td></td>
</tr>
<tr>
<td>UNIVERSITY of GREENWICH</td>
<td>Cardiff Metropolitan University</td>
</tr>
<tr>
<td>F1F9 SPREADSHEET ENGINEERING</td>
<td>ISACA® Northern England Chapter</td>
</tr>
<tr>
<td>OPERIS</td>
<td>BAKER TILLY</td>
</tr>
<tr>
<td>SYSTEMS MODELLING</td>
<td>AuditNet</td>
</tr>
</tbody>
</table>

ISBN 978-1-9054045-1-3