Revising the Panko–Halverson Taxonomy of Spreadsheet Risks

Raymond R. Panko
University of Hawaii
http://panko.shidler.hawaii.edu
Panko@hawaii.edu

Abstract

The Panko–Halverson Taxonomy of Spreadsheet Risks has been widely used since its creation in the 1990s. This paper revises that taxonomy. It introduces new ways to count cell error rates, the concepts of violations versus blameless errors, mistakes, slips, and lapses, context errors, and an improved discussion about risks by life cycle stage.

1. Introduction

The widely cited Panko–Halverson [1996] taxonomy of spreadsheet risks was created in 1993. After all this time, the taxonomy might be expected to need revision, and this certainly is the case. This paper presents a revised taxonomy of spreadsheet risks based on the original Panko–Halverson taxonomy.

Another reason to reconsider the Panko–Halverson taxonomy is that although it is widely used and cited, most uses of the taxonomy are based on misinterpretations that consider only one part of the taxonomy. For instance, the taxonomy really is a taxonomy of risks, and errors are only one cluster of risks [Panko and Halverson, 1996]. In addition, the taxonomy has a life cycle stage dimension which usually has been largely ignored.

Nearly every study of spreadsheet errors uses a taxonomy of errors. Why are spreadsheet taxonomies needed? Research on spreadsheet errors and human error in general has shown that there are distinct types of errors that vary in causation, frequency of occurrence, frequency of spontaneous detection, effective amelioration techniques, delectability during testing and inspection, and other important dimensions [Panko, 1998 revised 2008]. If an experimental treatment or good practice recommendation does not take error type into account, it is not likely to be very useful. In addition, it is difficult to compare the results from different studies unless these studies use the same or comparable taxonomies.

The Panko-Halverson taxonomy

The Panko–Halverson taxonomy was presented in a conference paper in 1996. This paper is not readily available, so many people who use the taxonomy are familiar with only part of it. Figure 1 shows the complete taxonomy of spreadsheet risks. It also shows that the taxonomy has three dimensions—risk research issue, life cycle state, and methodology.

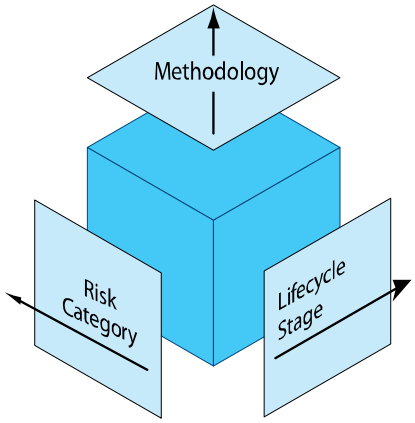

Figure 1: Panko-Halverson 1996 risks taxonomy

We will not consider the methodology dimension in this paper beyond mentioning its purpose, which was to note that most issues could be studied in several different ways, including fMRI imaging, protocol analyses, laboratory experiments, field audits, and surveys of corporations, users, and developers. Each methodology has strengths and weaknesses. For instance, they tend to vary widely in internal and external validity. Applying multiple methods provides convergent validity. If you get consistent with results different methods that have different strengths and weaknesses, you can have confidence in the results.

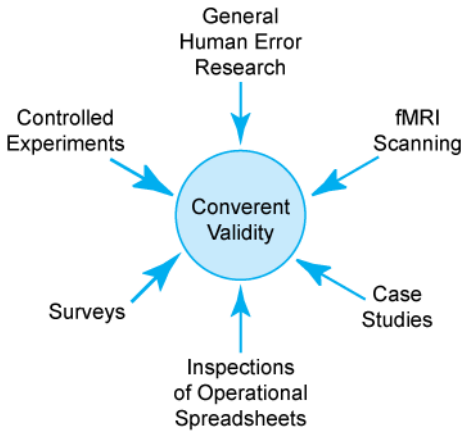

Figure 2: Convergent Validity in Spreadsheet Error Research

1. Actual Errors

The risks dimension lists many types of risks to be researched, including concerns about spreadsheet programs as a development environment, user work practices, the special dangers of assumptions, control policies, and different types of spreadsheet development environments that require different approaches to development [Panko and Halverson, 1996]. However, due to space limitations, we will focus on the most widely cited aspect of the risks dimension, namely what Panko and Halverson called "actual errors." Figure 3 shows how Panko and Halverson subdivided actual errors.

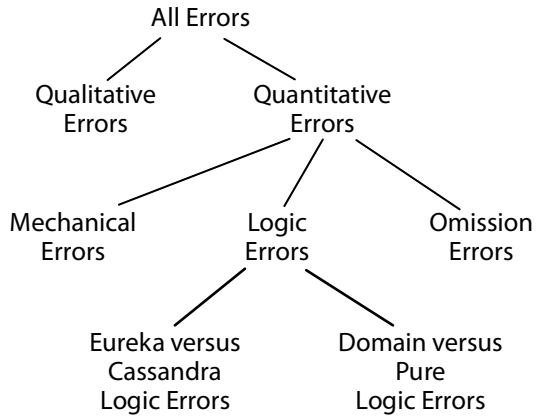

Figure 3: Categories of actual errors in the Panko-Halverson taxonomy

In general, the taxonomy of error types was based on human error research, most importantly on Allwood's [1984] research on problems when students engage in mathematical problem solving.

1.1 Qualitative versus quantitative errors

The taxonomy made a distinction between quantitative and qualitative errors. These two types of errors were defined in the following ways [Panko and Halverson, 1996]:

"Quantitative errors are numerical errors that lead to incorrect bottom-line values."

"Qualitative errors, in turn, are flaws that do not produce immediate quantitative errors. But they do degrade the quality of the spreadsheet model and may lead to quantitative errors during later 'what if' analyses or updates to the model. Others make debugging difficult, raise maintenance costs, or may cause users to misinterpret the model's results."

Notice that the core distinction is whether the error *immediately* produces incorrect bottom-line values.

1.2 Mechanical quantitative errors

Quantitative errors were further defined as "typing errors, pointing errors, or other simple slips." Mechanical errors are frequent, but they have a high likelihood of being detected if they occur [Panko, 1998 revised 2008].

1.3 Logic quantitative errors

Logic errors, in turn, were defined as "incorrect formulas due to choosing the wrong algorithm or creating the wrong formulas to implement the algorithm [Panko and Halverson, 1996]." Logic errors are dangerous because they are less likely to be detected than mechanical errors during development and testing [Panko, 1998 revised 2008].

Panko and Halverson further subdivided logic errors based on their research and previous spreadsheet research. First, they divided logic errors into errors of pure logic and logic errors relative to the specific domain being modeled.

Lorge and Solomon [1955] introduced the concept of Eureka logic errors. These are logic errors that will be immediately recognized as being errors when they are pointed out. In contrast, Panko and Halverson [1997] noted that when teams created spreadsheets, one person sometimes was correct but could not convince others in the group that the error really was an error. Panko and Halverson [1997] called these errors Cassandra errors, after a character in the Iliad who was doomed to predict disasters but never be believed. Cassandra logic errors obviously create special problems for testing and group development.

1.4 Omission quantitative errors

Omission errors were defined as 'things left out of the model that should be here. They often result from a misinterpretation of the situation." Omission errors are dangerous because research has shown that omission errors have a very low rate of detection [Panko, 2008].

1.5. Measures of error rates

Deciding how to measure rates might not seem to be a problem. However, Panko and Halverson [1996] noted that there are several ways to measure errors, including the percentage of models containing errors, the average number of errors per model, the magnitude of the errors, and the cell error rate.

Percentage of models that contain errors
The average number of errors per model
Error magnitudes
Cell error rate

Figure 4: Measures of Error Rates

Each counting method highlights different things. For example, in field audits, the fact that nearly all spreadsheets have been found to contain errors is extremely telling. In laboratory experiments, in turn, cell error rates for different types of models is important as a way of understanding how frequently people make errors when they enter a formula or other information in a cell.

The cell error rate (CER) was defined as "the percentage of cells that have errors [Panko and Halverson, 1996]." The concept of measuring the percentage of actions that are erroneous is widely used in human error research [Panko 2008]. For instance, in software, the measure of faults per thousand lines of noncomment source code allows error frequencies in programs of different sizes to be compared.

2. The lifecycle dimension

Although the error measuring aspects of the Panko and Halverson taxonomy are widely known, few authors, including those who use the taxonomy in their own work, have been aware of the lifecycle dimension. This is an important oversight because many risks and types of errors only appear in certain life cycle stages.

2.1 Requirements and design

In the traditional systems development life cycle, there are several activities that take place before coding begins. Panko and Halverson [1996] only mentioned requirements and design. In addition, the only mentioned one risk is the pre-coding stages: the fact that developers often do almost no prior thinking before beginning to type the model on the screen [Panko and Halverson, 1996].

2.2 Cell entry stage

During the 1980s, two studies [Olson, 1987-1988; Lerch, 1988] conducted protocol analyses of subjects creating spreadsheet models. They discovered that subjects made errors in roughly 10% of their spreadsheet formulas. However, the subjects fixed most of these errors spontaneously, just as Allwood's [1984] subjects made many errors that they fixed by themselves. Although cell entry is not a full stage, by defining it as a stage, Panko and Halverson [1996] emphasized the complex dynamics of error-making and spontaneous error detection and correction.

2.3 Draft stage

When a developer creates a module, he or she will at some point consider it "finished." Of course, it will still need to be tested because it probably contains errors in a few percent of its cells. Programmers usually refer to this as the module development stage and to the testing that takes place at the end of module development as unit testing.

2.4 Debugging stage

Instead of using the programming term *testing*, Panko and Halverson [1996] referred to the testing that takes place as the debugging stage. This was a poor choice of words because debugging has a particular meaning in programming. In debugging, the developer or tester already knows that an error has occurred. Debugging is the process of finding the error.

Another thing that Panko and Halverson [1996] did not make clear is that testing should not be done after an entire spreadsheet is created. As in programming, there should be unit testing for modules, as just noted. In addition, there should be several stages of functional testing as the modules and larger units are integrated together. Finally, the entire

spreadsheet must be tested when all the functions are knit together.

2.5 Operational stage (use errors)

The systems *development life cycle* in software ends when the program is created. However, the broader systems *life cycle* includes the program's (or spreadsheet's) use after creation. This operational stage is likely to continue for many months or years. This is far longer than the development stage. Panko and Halverson [1996] said little about the operational stage beyond noting that operational spreadsheets frequently contain errors that are only detected after months or years.

3. Rethinking error measurement

Now that we have looked at the Panko and Halverson [1996] taxonomy and problems with it, we will begin to develop a revised taxonomy of spreadsheet risks.

3.1 Beyond Cell Error Rates

Although most users of the Panko and Halverson [1996] taxonomy have focused on the cell error rate, the other error rate measures listed in Figure 4 are also important.

3.2 Error Seriousness

Error seriousness obviously is important. Errors that are not serious are not great causes for concern. Panko [1998 revised 2008] reviewed several studies that did field audits of real-world operation spreadsheets. Most of these audits found errors in nearly all of the spreadsheets they inspected. Yet most of these studies did not report errors unless they exceeded a certain level of seriousness, Most recently, Powell, Lawson, and Baker [1997] inspected 25 spreadsheets and found many serious errors.

However, the concept of "serious error" is ambiguous. The seriousness of errors is obviously a continuum from embarrassment through disasters.

More importantly, error seriousness is not the same as error size. In well-known and tightly-controlled situations such as annual budgeting, an error of even four percent can be devastating if the organization is held to the budget tightly. However, in doing rough estimates of the size of a potential market segment, much larger percentage errors are expected, and results are interpreted with that in mind. Overall, seriousness cannot be discussed except in terms of what error rates are acceptable or unacceptable in terms of what accuracy is expected from the spreadsheet.

For example, the author started doing work in spreadsheet error research when he was on the board of a small parochial school. A CPA prepared a budget and forecast a 6% shortfall. Teacher salaries were not increased, and the school lost its best teacher. At the end of the year, there was a 4% surplus, which went to the parent church instead of to the school. Board members satisfied themselves that the end-of-year discrepancy was due to several specific unforeseen positive events. When there was another deficit forecast for the next year, two board members asked the CPA to check for errors. He found an error in a simple table lookup that explained the false shortfall projections. (He also found quite a few smaller errors.) In this case, a very small percentage error led to significant staff morale problems and the loss of one of the school's teachers.

Figure 5 shows a table in a 2008 update of Panko's 1998 (revised 2008) paper on spreadsheet errors. Note that most of the studies of operational spreadsheets that listed in the figure only reported errors that were serious in the context of the spreadsheet.

Study	Pct. of SSs with Errors	Seriousness Measure
Davies & Ikin, 1987	21% of 19	Only serious errors were reported
Butler, 1992	11% of 273	Only errors large enough to demand additional payments
Hicks, 1995	100% of 1	Error of 1.2% would have cost about $1 billion
Coopers & Lybrand, 1997	91% of 23	Off by at least 5%, which is a material error in financial reporting
KPMG, 1998	91% of 22	Only errors that could lead to an incorrect decision
Butler, 2000	86% of 7	Only errors large enough to demand additional payments
Powell, Lawson and Baker, 2007	44% of 25	Of the 10 spreadsheets in which error size was recorded, all had errors of $100,000 or more

Figure 5: Seriousness of Errors [Panko, 2008]

In addition, the 2008 paper describes the results of two interviews the author had with principals of spreadsheet auditing consultancies in the UK (where certain spreadsheets are required by law to be audited). Both said that they had never audited a spreadsheet and found it to be error-free. In addition, they both said that about five percent of the spreadsheets they audited had very serious errors (not

simply serious errors). One referred to such errors as show-stopper errors.

3.3 Types of Cell Error Rates (CERs)

As noted earlier, Panko and Halverson [1996] defined the cell error rate (CER) as "the percentage of cells that have errors." There are two problems with this definition. The first is how to count errors. The second is what "percentage of cells" means.

Counting errors. Panko and Halverson [1996] did not describe how to count errors in their taxonomy paper. However, in their paper that reported on the experiment during which the taxonomy was created, they did specify how to count errors [Panko and Halverson, 1997].

Their method might be called the "original sin" method. First, if the error occurred in a single cell, it was counted as a single error despite the fact that subsequent cells might be wrong as a result. This is a good approach because some errors affect no subsequent cells and others affect many subsequent cells. In looking at error rates, counting cells in which an error is made makes the most sense.

Second, if an error is copied to other cells mechanically or by making the same errors in multiple cells, Panko and Halverson [1997] counted this as a single error.

Overall the original sin approach focuses on counting distinct human error events during development.

The denominator. The denominator in the cell error rate was defined as being "cells." However, Panko and Halverson [1997] never counted label cells. In addition, some types of errors only occur in formulas, so dividing these by both formula cells and number cells is misleading, especially because different spreadsheets can have vastly different ratios of formula cells to number cells. These considerations lead to the suggestion that the name cell error rate should be modified to specify the type of denominator used in calculations. Figure 6 presents our suggestions, which include CERF (formulas), CERV (value cells, including both formulas and numbers), CERN (numbers), CERT (text), and CERA (all nonempty cells).

Name	Denominator
CERF	Formula cells
CERV	Value cells: number and formula cells
CERN	Number cells only
CERT	Text cells
CERA	All nonempty cells

Figure 6. Types of Cell Error Rates

4. Rethinking error types

Figure 3 showed how Panko and Halverson defined "actual errors." Figure 7 shows our revision to this error typology.

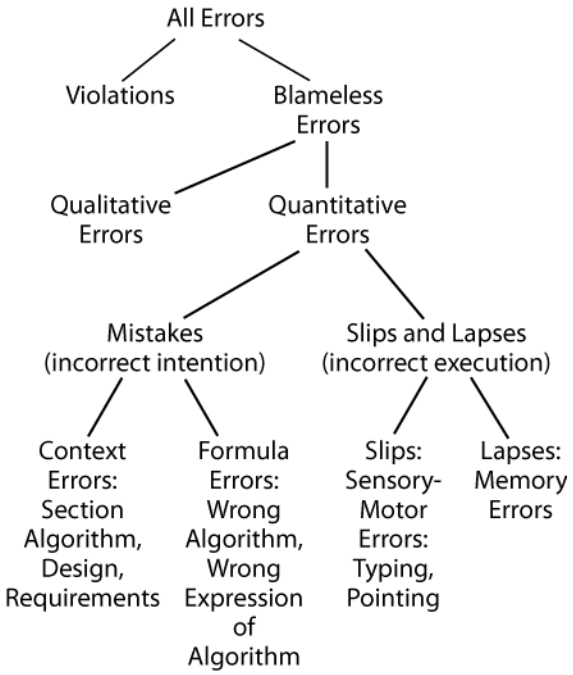

Figure 7: Revised Error Typology

4.1 Violations versus blameless errors

In software inspections, it has long been known that software developers should not be blamed when errors are found [Beizer, 1990]. For one thing, it would be unfair because human error can never be avoided entirely. More importantly, if we do not hold developers blameless, they are likely to withhold error discoveries. This same problem has been seen in medicine [Edmondson, 1996]. Consequently, in most cases, errors should be considered to be blameless errors.

However, research on automobile accidents has emphasized that it is useful to distinguish between ordinary errors and violations, such as speeding or driving while under the influence [Reason, 1990]. In spreadsheet development also, violations should be treated separately from blameless errors.

Creating a misleading spreadsheet, which can range from some puffery to actual fraud, is certainly a violation. So is not following a company's policies regarding spreadsheet development testing, documentation, control archiving, and other matters. Having the spreadsheet compute results that are explicitly

forbidden by the corporation in a particular business domain (such as having one person write purchase orders, approve them, and audit them; or computing overtime pay for salaried employees who are not eligible for overtime pay) is also a violation, as is doing something against the requirements of a particular compliance law.

Another consideration is that there are several ways in which errors can be blameful, including making deliberate errors for personal gain or advancement, negligence, failing to understand corporate policies, and developing spreadsheets beyond the capabilities of the developer. In general, the whole concept of violations needs to be examined in much more detail. As in the case of error seriousness, violation seriousness is likely to be very difficult to define well.

4.2 Mistakes versus slips and lapses

The Panko-Halverson taxonomy [1996] distinguished between logic errors and mechanical errors. In our revised error taxonomy in Figure 7, this terminology is replaced by the far more widely used terminology introduced by Norman and Reason [Norman, 1981; Reason, 1990].

This terminology assumes a basic distinction between mistakes on the one hand and slips and lapses on the other. A mistake is a wrong intention, while a slip or lapse is a failure to carry out an intention correctly [Norman, 1981; Reason, 1990]. This is a simple but robust distinction.

4.3 Slips versus lapses

The distinction between slips and lapses was proposed by Norman [1981]. A slip is an error during a sensory-motor action, such as typing the wrong number or pointing to the wrong cell. In contrast, a lapse occurs within the person's head. Typically, a lapse is a failure in memory, and this failure is often caused by overloading the limited human memory capacity.

To give an example of a lapse, the Galumpke task specifies unit costs for labor and materials for two years. Keeping four numbers in a person's head is difficult, and subjects often transposed the four numbers or made similar mistakes. Skipping a step in a solution is also a lapse.

The distinction between slips and lapses is important because while slips may leave traces on the spreadsheet (such as a formula reference that points to a blank cell), lapses are much less likely to leave traces on the spreadsheet's structure or in a particular formula.

The impact of dividing the Panko and Halverson mechanical error category into slips and lapses is shown in Figure 8. In a corpus of spreadsheets described by Panko [2000], 82 subjects each developed a spreadsheet to provide a decision maker with a pro-forma income statement. In the corpus, the author classified 28% of the errors as logic errors, 21% as omission errors, and 41% as mechanical errors. The high percentage of mechanical errors was good news for automated error detection software because some mechanical errors such as pointing errors may leave discoverable artifacts on the spreadsheet.

Type of Error	Mechanical	Slip	Lapse
Pointing errors	8	8	0
Year 1 and Year 2 sales salaries translated into two salespeople instead of two years	1		1
Owner salary = 60,000 instead of 80,000	1		1
Typing incorrect value for unit materials and labor cost (usually due to a transposition)	12		12
Units sold value for Year 2 used in Year 1	1		1
Units sold value 32,000 instead of 3,200*	1	1	
Sign incorrect	2	2	
Parenthesis error	1	1	
Rent = 3,600 instead of 36,000*	1	1	
Total Mechanical/Slip/Lapse Errors	28	13	15
Percentage of errors	41%	19%	22%

Figure 8: Mechanical Errors, Slips, and Lapses

However, even a cursory inspection of the mechanical errors indicates that quite a few were lapses. Figure 8 shows that when a classification based on slips and lapses is used, slips only account for 19% of the total errors, while 22% of the errors were lapses. In other words, slips, which are most likely to leave traces such as a link to an empty cell, accounted for only one of every five errors in this corpus.

4.4 Formula Mistakes

The figure shows two types of mistakes involving individual formulas. First, the formula may be based on the wrong algorithm. Second, the developer may have the wrong algorithm but may

4.5 Context Mistakes

Flower and Hayes [Flower and Hayes, 1980; Hayes and Flower, 1980] studied the process of writing. Using protocol analysis, they found that their subjects had to work at several levels of abstraction simultaneously. They had to select specific words while generating sentences, and sentence production had to fit into the author's plan for the paragraph, for larger units of the document, and for the document as a whole. Planning had to be done at all levels of abstraction, and it had to be done simultaneously. Each level of abstraction created constraints that had to be obeyed when considering other levels.

Figure 9 shows that the Flower and Hayes taxonomy of concerns can be viewed as a hierarchy that places all of the weight of all context levels on the writing of a word. This can create enormous overload on the writer's memory and planning resources.

Writing	Spreadsheets
Purpose	Requirements
Document	Spreadsheet
Chapter	Module
Paragraph	Section Algorithm
Word	Formula

Figure 9: A Context Hierarchy

In spreadsheet development, the same mental load is generated. Whenever a developer types a formula, he or she also has to be cognizant of the algorithm for the formula, the algorithm for a larger section of the spreadsheet, and for the spreadsheet as a whole, and for requirements.

The use of context errors to describe mistakes is theoretically important, but it is likely to be very difficult to apply in process unless one is doing protocol analysis as the spreadsheet developer works.

Note that there are no longer omission errors in the taxonomy. Omission errors and misinterpretations are best viewed as context errors.

4.6 Applicability

Like the original Panko-Halverson taxonomy, the revised taxonomy shown as Figure 7 was created with laboratory experiments in mind. In addition, it was implicitly created, like its predecessor, for spreadsheets with known solutions. Later, the taxonomy was found to be useful for spreadsheet inspection experiments to detect seeded errors in spreadsheets. These two restrictions are likely to limit its usefulness in field audits, although it has been successfully applied there by Hicks [1955] and Lukasic [1998]. The value of experiments is that they are fairly close to the level of mental processes, allowing educated guesses to be made about why certain errors occur, particularly because data from many subjects can be collected.

5. Errors in life cycle stages

Earlier, we saw that the Panko-Halverson taxonomy took the concept of life cycle states into account, albeit sketchily. Most importantly, their work focused on spreadsheet development and testing, so these are the stages they fleshed out in the most detail.

Life cycle thinking is important for several reasons in error research. Figure 10 shows one of these—the fact that error rates within a spreadsheet are likely to vary by spreadsheet stage. During analysis, requirements, and design, error rates are relatively small, increasing as these stages develop. However, if these errors propagate into coding, their amelioration cost will be much higher.

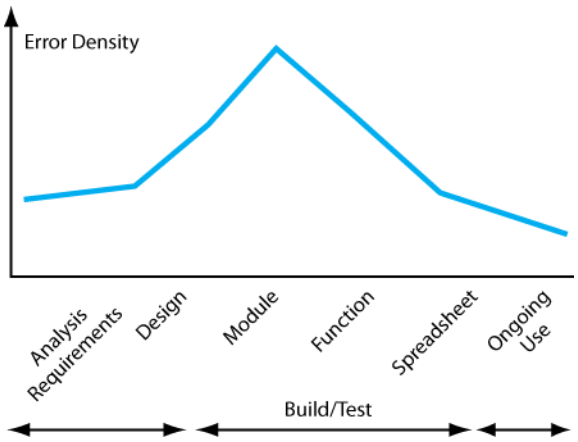

Figure 10: Error Rate by Life Cycle Stage

Next, the figure shows a number of build/test stages. The term build/test is used to indicate that development and testing are not separate stages. Rather, as noted earlier, they occur at successive levels in development—module (unit) development and testing, functional development and testing, and full model development and testing. The figure shows that error rate is likely to drop as these stages proceed because of extensive testing at earlier build/test stages.

Finally, the life cycle has ongoing use. In software products, error rates tend to fall during ongoing use as remaining issues are discovered and addressed. The same basic dynamic is likely to hold true for spreadsheets. Consequently, in analyses of operational spreadsheets, it is important to understand how long they have been in use.

In the operational stage, the spreadsheet is embedded in an organizational work system, and this leads to human errors in use of the spreadsheet. Consequently, at the operational stage, we need to understand the entire organizational system, not merely the spreadsheet.

One ongoing use error common in uprotected spreadsheets occurs when a user types a number into a formula cell. While such a "hardwiring" error does not necessarily make that particular use of the spreadsheet incorrect, if the spreadsheet is saved with the overwritten formula, future uses of the spreadsheet will almost certainly be incorrect.

5.1 Analysis, requirements, and design

In analysis, requirements, and design, there are many possible errors that can occur. In analysis, a poor analysis of the current system can lead to a spreadsheet that lacks important functionality in the current system (including manual systems). This will require extensive refitting later. In requirements, it is easy to overlook an important requirement or misinterpret a user's needs. This again can result in high costs later.

Problems in analyzing the current system
Problems in defining requirements
Problems in using design tools
Comparison of design tools

Figure 11: Needed Analysis, Requirements, and Design Research

The design stage usually consists of at least two stages—high-level design and detailed design. Design is a complex intellectual activity, and mistakes in design are likely to occur. This is an area that has received little attention, perhaps because so few end user developers actually do design prior to building. We need to understand influence diagrams and other design tools individually for their applicability to end user developers and spreadsheet design, and we need to compare the comparative strengths and weaknesses of different design approaches.

5.2 Cell entry stage

Figure 10 does not have a cell entry stage. However, we need to understand error making and error correction as developers and testers work. When Allwood [1984] watched students do mathematical problems, he noted that they often stopped to check their work and in fact identified three different error detection methods. In addition, Gould [1980] noted that when people wrote, they spent a good deal of their time planning and reviewing what they have already done instead of writing. We need to use protocol analysis to understand detailed planning, error-making, and error checking processes when users develop spreadsheets. This is especially true for context management and context errors, which have not been studied explicitly. This detailed research also needs to be done in testing, to understand the approaches that different subjects use to identify errors and then to debug the system to remove these errors.

Protocol analyses in development and testing
Context management: planning and reviewing
Error checking strategies during development
Comparisons of alternative testing techniques

Figure 12: Needed Detailed Development and Testing Research

5.3 Build/Test Stages

As noted earlier, the errors shown in Figure 7 are designed to reflect the types of errors made during build/test stages. And as just noted, we need to do protocol analyses to understand what people actually do during build/test stages. We also need to examine different error-reduction strategies for the build/test stages and identify which ones are productive, safe, and effective.

Assessment of error reduction strategies
Productivity
Safety
Effectiveness in reducing errors

Figure 13. Needed Build/Test Strategy Research

5.4 Operational stage

For spreadsheet error researchers, the operational stage is a "target-rich environment" for spreadsheet researchers because much research needs to be done beyond inspections of operational spreadsheets.

Inspections of operational spreadsheets

Input error (human and automated)
Organizational systems analysis
Interpretation errors research
Compliance with laws
Security
Fraud

Figure 14. Needed Operational Stage Research

One thing we need to look at is input errors and how they occur. For example, in the massive Fidelity mutual fund error, a clerical worker entered the sign incorrectly when faced with entering a loss instead of the usual gains. More generally, we need to study the full human and organization systems that surround spreadsheet usage in organizations. This extends from issues about errors in transferring data from other databases to the mechanisms that organizations use to ensure that that they are using the spreadsheet properly.

We also need to understand output. Of course, if output contains errors, this is certainly bad. However, because of poor design or lack of user training, it is possible for customers of the spreadsheet to misinterpret its results. This is another new area for research.

We are in a "compliance era" in which organizations have to be able to demonstrate their most important business processes are protected to a reasonable extent against error and fraud. In pharmaceuticals, the 21 CFR 11 rules already require the careful control of research that is analyzed or reported on spreadsheets. In Sarbanes–Oxley, the importance of controlling spreadsheets has also become obvious. There are now many privacy laws, and protecting personally identifiable information is difficult because so much of it may be uncontrolled spreadsheets. Although there are tools for access control, auditing, the prevention of data extrusion, and other matters, these are not widely used in organizations.

Fraud is even more difficult to reduce to a reasonable level. John Rusnak's $691 million fraud at Allfirst Bank from the mid 1990s through 2002 is an excellent example of how corporations, which almost all use spreadsheets extensively, need to rethink security and fraud control.

6. Conclusion

This paper proposes a revision to the Panko–Halverson taxonomy of spreadsheet risks. The specific proposed changes, which are listed in the Abstract, are provisional at this point and may be changed following feedback and experiences using the revised taxonomy.

7. References

Allwood, C. M. (1984). "Error Detection Processes in Statistical Problem Solving." *Cognitive Science, 8(4),* 413-437.

Beizer, B. (1990). *Software Testing Techniques*. 2nd ed., New York: Van Nostrand.

Edmondson, A. C. (1996). Learning from Mistakes is Easier Said than Done: Group and Organizational Influences on the Detection and Correction of Human Error. *Journal of Applied Behavioral Science, 32(1),* 5-28.

Flower, L. A., & Hayes, J. R. (1980). "The Dynamics of Composing: Making Plans and Juggling Constraints," *Cognitive Processes in Writing.* Eds. L. W. Gregg & E. R. Steinberg. Hillsdale, NJ: Lawrence Erlbaum Associates. 31-50.

Gould, John D. "Experiments on Composing Letters: Some Facts, Some Myths, and Some Observations, Chapter 5 in Lee W. Gregg and Erwin Steinberg (eds.) *Cognitive Processes in Writing*, Lawrence Erlbaum: Hillsdale, NJ, 1980, pp. 97-127.

Hayes, J. R. & Flower, L. (1980). "Identifying the Organization of Writing Processes," *Cognitive Processes in Writing.* Eds. L. W. Gregg & E. R. Steinberg. Hillsdale NJ: Erlbaum. 31-50.

Hicks, L., NYNEX, personal communication via electronic mail, June 21, 1995.

Lerch, F. J. (1988). *Computerized Financial Planning: Discovering Cognitive Difficulties in* Knowledge *Building.* Unpublished Ph.D. Dissertation, University of Michigan, Ann Arbor, MI.

Lorge, I., & Solomon, H. (1955). Two Models of Group Behavior in the Solution of Eureka-Type Problems. *Psychometrika, 20(2),* 139-148.

Lukasik, Todd, CPS. Personal communication by e-mail, August 10, 1998.

Norman, Donald A., "Categorization of Action Slips," *Psychological Review, 88*, 1981, 1-15.

Olson, J. R., & Nilsen, E. (1987-1988). Analysis of the Cognition Involved in Spreadsheet Interaction. *Human-Computer Interaction, 3(4),* 309-349.

Panko, R. R. (2008a). Human Error Website. (http://panko.shidler.hawaii.edu/humanerr.htm). Honolulu, HI: University of Hawaii.

Panko, R. R., (2000, January) "Two Corpuses of Spreadsheet Errors," *Proceedings of the Thirty-Third Hawaii International Conference on System Sciences*, Maui, Hawaii.

Panko, Raymond R., (1998, Spring) *Journal of End User Computing*, Special issue on Scaling Up End User Development, 10(2), 15-21. Revised May 2008. http://panko.shidler.hawaii.edu/ssr/Mypapers/whatknow.htm

Panko, Raymond R. and Halverson, Richard P., Jr., (2001, July) “An Experiment in Collaborative Spreadsheet Development,” *Journal of the Association for Information Systems*, *2(4)*.

Panko, Raymond R. & Halverson, Richard Jr. (1997. Spring), “Are Two Heads Better than One? (At Reducing Errors in Spreadsheet Modeling),” *Office Systems Research Journal 15(1)*, 21-32.

Panko, Raymond R. and Halverson, R. H., Jr. (1996, January) “Spreadsheets on Trial: A Framework for Research on Spreadsheet Risks,” *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, *Volume II*, Kihei, Maui, 326-335.

Powell, Stephen G.; Lawson, Barry; and Baker, Kenneth R. (2007, July). “Impact of Errors on Operational Spreadsheets,” *Proceedings of the European Spreadsheet Risks Interest Group, EuSpRIG 2007 Conference*, University of Greenwich, London, pp. 57-68.

Reason, James P. (1990). *Human Error*, Cambridge University Press: Cambridge, England.