End User Computing: The Dark Matter (and Dark Energy) of Corporate IT

Raymond R. Panko
University of Hawaii
panko@hawaii.edu

Daniel N. Port
University of Hawaii
dport@hawaii.edu

Professionals Built the Titanic
Amateurs Built the Ark

Abstract

End user computing (EUC) applications are like dark matter in physics. They are enormous in quantity and importance yet have been largely invisible to corporate IT departments, information systems researchers, and corporate management. EUC applications, especially spreadsheet applications, are also “dark” in the sense that they pose a number of overlooked risks for organizations, including errors, privacy violations, trade secret extrusions, and compliance violations. On the positive side, EUC applications are also like the dark energy of physics. They are bringing critical gains to decision making, operational systems, and other important processes to every corner of the firm. It is time to stop ignoring end user computing in general and spreadsheets in particular. One point to consider in particular is that we should only talk about development, regardless of whether the developer user a 3GL or 4GL.

1. Introduction

1.1 Dark matter and dark energy

In the 1930s, astrophysicists learned that ordinary matter makes up only about a fifth of the total matter of the universe [31]. The rest consists of dark matter, which is not simply too dim to see but actually is radically different from ordinary matter. In the last years of the 20th Century, astrophysicists received another shock. Something else, which physicists labeled dark energy, is pushing the universe to expand at an ever-increasing rate. This dark energy is almost three times as large as ordinary matter and dark matter combined [31]. Traditional physics, instead of describing the universe, had only considered about five percent of it.

The contention of this paper is that end user computing (EUC) is both the dark matter and the dark energy of corporate information technology. It is like dark matter in that it is both enormous in size and impact yet seems to be invisible to the central corporate IT group, general corporate management, and information systems (IS) researchers.

EUC is also “dark” in another sense. Research has shown that developers in departments outside the central IS departments develop many incorrect applications [34]. The problem does not appear to be end user computing tools per se. For instance, we will see that error rates in EUC spreadsheet development are no higher than error rates when professional programmers write applications in 3GLs. Rather, the root problem appears to be poor end user development practices, especially an almost total lack of testing. We need to teach new dogs old tricks learned painfully by programmers in the early years of software development. However, nothing can be done about EUC risks until IS researchers decide to study them, corporate management assumes responsibility for the identification and reduction of these risks, and corporate IT departments stop saying, “End user computing? That’s a business thing.”
More excitingly, EUC also appears to be the dark energy of corporate IT. Applications developed by end users may power nearly every knowledge worker and department in corporations today. These applications go well beyond downloading and massaging data from central information systems. We will see that many are enormous and complex. In addition, a large fraction of all knowledge workers and departments seem to have one or more EUC applications that they characterize as mission-critical. On the positive side, it means that EUC applications are having a critical impact on business. On the negative side, “mission-critical” means that the application’s failure, unavailability, or incorrectness would have large negative consequences.

1.2 Application development tools

Often, practitioners and researchers make a distinction between programmed applications developed with third-generation languages (3GLs) and EUC applications built with spreadsheets, database management systems, and other fourth-generation languages (4GLs). This seems counterproductive. In his early years, the first author wrote 2GL programs using assembly language. When programmers switched to 3GLs such as COBOL or FORTRAN, they did not stop calling what they did programming. Certainly, many old dichards argued that 3GLs were toys that did not let you reach into individual registers. They also argued that 3GLs were too domain-specific. However, 3GL productivity gains were enormous, and rapidly increasing processing speeds made 3GL processing penalties unimportant. In addition, individual 3GLs proved to be broad-spectrum languages that could create a wide variety of applications. Although one might have to learn FORTRAN for calculation-intensive applications and COBOL for file-handling applications if one wanted to develop both types of applications, the gains were certainly worthwhile compared to staying with 2GLs.

Beginning in the 1970s, another generation of development tools emerged. People inside and outside of the central IT department began to develop applications with 4GL tools, including spreadsheet programs, database applications, and other 4GL integrated development environments. Again, many 3GL proponents argued that this is not “programming,” but is that relevant? The goal is to develop effective applications productively. Arguments stating that using 4GLs is not programming again boiled down to the inability to do low-level things and the domain-specific nature of the tools. Again, however, 4GL productivity trumped most small losses in what users could do, and processing power growth quickly removed “insufficient power” problems.

Furthermore, many new 4GLs revealed themselves to be extremely broad-spectrum tools. In particular, the spreadsheet’s grid-and-worksheets format proved itself equally adept at handling scratchpad calculations and applications with thousands of root (noncopied) formulas. To call spreadsheets personal productivity applications, as many IT professionals, IS researchers, and model curricula do, really makes very little sense given the enormous number of very large spreadsheet applications in business. Even without taking into account macros and Visual Basic for Applications, we have simply moved to yet another generation in application development tools. We should only talk about development, regardless of whether the developer user a 3GL or 4GL.

When Grossman et al. [19] studied 18 mission-critical applications, they found that in three cases, applications were developed by experienced 3GL programmers. In two of these cases, the developers considered both spreadsheets and 3GLs and consciously chose spreadsheets. In the other 16 cases, spreadsheets were such an obvious choice that nothing else was considered. In one case, the application was later ported to a 3GL because of spreadsheet size limits at the time.

Accordingly, we will simply focus on application development without regard to the generation of the integrated development environment (IDE) used to create applications. While one can usefully argue whether to use 3GL IDEs or 4GL IDEs for individual applications, we argue that the term “programming” has become a distraction.

1.3 The evolution of EUC

In the earliest days of computing, there were two distinct types of computing. First, there was scientific and technical computing, in which computational speed was the paramount hardware and software design concern. Companies such as Control Data Corporation dominated this type of computing in the early years. Today, supercomputers with massively parallel processing are often used for this type of application, but for many scientific and technical applications, spreadsheets and other EUC tools are widely used. At the Jet Propulsion Laboratory, for
example, spreadsheets are widely used despite extensive 3GL expertise in the programming staff [20].

The second type of computing was data processing, in which the critical design concern was the handling of record-structured files. IBM was initially dominant and largely continues that dominance today. Information systems programs in business schools focused primarily on data processing and its successors, including management information systems. Scientific and technical computing, despite having been the first type of computing and despite continuing in importance today, has tended to be outside the radar of IS programs. Perhaps this is because business schools are the home schools of IS programs, so there is a tendency for them to focus on managers and accounting and finance functions.

Scientific and technical computing actually predated data processing. ENIAC, for instance, was designed for technical computing. As 3GL programming languages emerged, many scientists and technologists (including the first author of this paper) began to write programs in FORTRAN and successive mathematically oriented 3GLs. This began in earnest in the 1970s, and 3GL programming skills were soon expected of scientific and technical professionals.

A little later, data processing end users began to use report generators and similar tools to create reports and do other light work on their own. However, these were narrow-spectrum tools.

When spreadsheet programs appeared, they rapidly became development environments for both technical and DP users. However, they also began to be used by many other types of users, for instance those in the humanities who did computational textual analysis of novels. Scientific and technical professionals also began using spreadsheets extensively [e.g., 20].

### 1.3.1 Spreadsheets in operational systems

Although IS researchers tend to think that spreadsheets are used primarily for decision support, research generated by Sarbanes-Oxley (SOX) requirements showed that spreadsheets are used in many operational systems such as financial reporting [32]. The Sarbanes-Oxley Act of 2002 spurred research into the critical process of financial reporting, which is important in almost every firm. The results were surprising to those who viewed spreadsheets as decision aids alone.

- In 2004, financial intelligence firm CODA reported that 95% of U.S. firms use spreadsheets for financial reporting (www.coda.com).
- RevenueRecognition.com (2004) (now Softtrax) had the International Data Corporation interview 118 U.S. business leaders. IDC found that 85% were using spreadsheets in financial reporting and forecasting.
- CFO.com [11] interviewed 168 finance executives about information technology in the finance department. Out of 14 technologies discussed, only two were widely used— spreadsheets and basic budgeting and planning systems. Every subject said that his or her department used spreadsheets.
- A.R.C. Morgan interviewed 376 individuals responsible for overseeing SOX compliance in multinational corporations [34]. More than 80% said that their firms used spreadsheets both for managing the financial reporting control environment and for financial reporting itself.
- In a webcast for Deloitte on May 22, 2005, the first author was able to ask a series of questions to just over 800 financial professionals and officers in corporations. One question specifically asked, “Does your firm use spreadsheets of material importance in financial reporting?” Of the respondents, 87.7% answered in affirmative, while 7.1% said, “No.” (Another 5.2% chose “Not Applicable.”) This is very significant because every part of the financial reporting process that is of material importance must be controlled or the firm will receive a negative attestation, and this is extremely bad.
- Furthermore, when companies use spreadsheets for financial reporting, they often use many. One firm used more than 200 spreadsheets in its financial planning process.
- Heavy spreadsheet use in financial reporting was even common in companies that used financial reporting applications; in such cases, companies often use spreadsheets to do such critical and risky work as computing end-of-period adjustments.
- Although the most evidence for the importance of operational spreadsheets comes from SOX-related studies, operational spreadsheets are often seen in other contexts [e.g., 19, 23].

### 1.3.2 Spreadsheets in tracking applications

Spreadsheet programs are also used for tracking applications, and the Excel 2007 table functionality leap probably has probably increased this type of use markedly. Overall, spreadsheet programs are ex-
tremely broad-spectrum IDEs, not limited to traditional decision support applications.

Overall, spreadsheet programs are extremely broad-spectrum IDEs, and they are not limited to traditional decision support applications. They are used for operational systems, tracking systems, and many other large and important systems.

1.3.3 Beyond personal productivity
Another point to note is that many writers classify spreadsheets as personal productivity tools. However, while scratch-pad personal applications are common, we will see that data since the 1980s has shown that many spreadsheets in organizations are extremely large and complex. Spreadsheets are used as personal productivity tools, but they are also used extensively as organizational performance tools.

Spreadsheets are used as personal productivity tools, but they are also used extensively as organizational performance tools.

2. Evidence for the importance of end user computing
Of course, if spreadsheets are widely used but are not important, then there is no problem. However, there is strong convergent evidence suggesting that spreadsheet applications are extremely important in corporations today.

2.1 Spreadsheets and Sarbanes–Oxley
We saw in the previous section that corporations are required to maintain strong controls over all material aspects of financial reporting. The Delloite webcast in 2005 showed directly that spreadsheets were used extensively in materially important aspects of financial reporting, and the other sources strongly supported the idea that spreadsheets are used very heavily in financial reporting.

2.2 Computer and application penetration
The most basic data on the importance of end user computing is the extent to which end user computing tools are used. The Bureau of Labor Statistics conducted a series of surveys regarding computer penetration and application use at work. It analyzed the data by occupational category. In 2001, the Bureau found that 72 million Americans used computers at work, with 60% using spreadsheets or databases and 15% saying that they “did programming.” Scaffidi, Myers, and Shaw [39] projected these figures to 2012, using BLS population projections and a projection of use parameters based on historical data. They estimated that 90 million Americans will be using computers at work in 2012, that over 60% (55 million) will be using spreadsheets and databases, and that 15% (13 million) will be doing programming. In comparison, the BLS projects 3 million programmers for 2012 [39]. Certainly, the number of employees using application development tools will be far larger than the number of programmers.

2.3 End user versus IS application development
Although there is a large number of end users using application development tools, end users almost certainly use these tools for application development less frequently than IT application developers use their tools. So how do end user and IT development compare?

Even during the 1980s, the data indicate that end user development was significant compared to IT development.
- In 1982, Benjamin [3] presented an analysis of mainframe usage. At the time, business end user development was dominated by mainframe tools. In 1980, he found that end user computing was consuming 40% of CPU cycles. Furthermore, this number was growing rapidly, and Benjamin forecast that end user computing would be larger than centrally developed computing by 1995, consuming 75% of all CPU cycles. Of course, the emergence of personal computers changed the picture, but Benjamin’s analysis indicated that even thirty years ago, end user computing was not trivial compared to central IS systems.
- Panko [35] listed two other sources that looked at end user computing in the 1980s. In 1986, Ronald Brezinski of Quaker Oats found that the IS department was paying all the bills but was only managing 40% of the CPU cycles. IBM Canada, about the same time, indicated that 25% of its computing was coming from end user computing.

Although these numbers are not precisely the same, they all indicate that end user computing was very large before a large fraction of all today’s workers were born.

But how much time do users spend in application development? We will do an analysis based on Bureau of Labor Statistics population data, past use-of-time data for programmers, and estimates of how much time managers and professionals spend on
spreadsheet analysis based on a survey by Baker, et al. [2], whose respondents indicated that they spent a quarter of their time doing spreadsheet development (as opposed to spreadsheet use).

Table 1: Application Development Time

<table>
<thead>
<tr>
<th>Category</th>
<th>IT Application Developers</th>
<th>Managers</th>
<th>Managers and Professionals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employment (millions)</td>
<td>2,158</td>
<td>12,642</td>
<td>27,147</td>
</tr>
<tr>
<td>Percentage of time with</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>application development tools</td>
<td>60%</td>
<td>25.4%</td>
<td>25.4%</td>
</tr>
<tr>
<td>FTE development employment</td>
<td>1,295</td>
<td>3,213</td>
<td>6,900</td>
</tr>
<tr>
<td>Ratio</td>
<td>1.0</td>
<td>2.5</td>
<td>5.3</td>
</tr>
</tbody>
</table>

The table indicates that although managers and professionals spend less time with application development tools than do professional application developers, their large numbers seem to mean that they collectively spend more time developing applications than do IT application developers. To equate application development with IT programmers seems ill advised.

2.4 Censuses of spreadsheets

Regulatory compliance requirements like the Sarbanes-Oxley Act of 2002 have forced companies to look at how they are using spreadsheets in several critical business functions. As we saw earlier, when organizations began to look at their financial reporting processes, they discovered an enormous number of spreadsheets.

Recently, vendors have developed tools to help companies manage their important spreadsheets. Most of these vendors recommend undertaking a series of tasks, often beginning with a census of all spreadsheets available on servers and, if possible, on PCs and mobile devices. This usually results in companies realizing that they have huge numbers of spreadsheets on their servers.

In a 2010 presentation at the EuSpRIG conference, CIMCON gave sample data from a few of its recent censuses.

- A large financial services company had 180,000 spreadsheets.
- A very large global bank had 8 to 10 million spreadsheets, not the one or two million it believed it had.

After the presentation, a principal from another census tool vendor in the audience noted that his firm had one recent case in which it identified one million spreadsheets at a company, counting only those accessed in the previous month.

In 2011, McDaid, et al. [29] used the Luminous Map product to do an analysis of spreadsheets in the financial departments in two organizations. Although the total number of professional employees in the two financial departments was only 30, the census found 65,806 spreadsheets on the department servers—2,194 per professional employee.

2.5 Spreadsheet size and complexity

There is a common belief that spreadsheet programs are used to develop small and simple models, many of which are one-time calculations. However, as noted earlier, spreadsheets are broad-spectrum calculation and data tools that can be used to develop extremely large and complex applications, especially now that PC processing power and spreadsheet size limits have become nonissues except in the most extreme cases.

There have long been indications that many spreadsheets are very large [e.g. 5, 10, 15, 20] and extremely complex [20]. Census programs have given us the ability to characterize the complexity of spreadsheets based on very large samples. For example, the 65,806 workbooks that McDaid, et al. [29] analyzed contained over a billion cells. A quarter billion of these cells (24%) were formula cells. This was an average of 4,000 formulas per spreadsheet. Even with extensive formula copying, this indicates a large number of root (original) formulas per spreadsheet. Cragg and King [10], Hicks [22], Lukasic [28], Lawrence and Lee [27], and Powell, Baker, and Lawson [36] also found large and complex spreadsheets when they identified operational spreadsheets to audit.

2.6 Importance

If there are many large and complex spreadsheets in organization, we need to ask if many of these are actually important to the firm.
2.6.1. Samples of spreadsheets

There are two sources of data on the importance of spreadsheets. The first is surveys of samples of spreadsheets.

In Hall’s [20] study, only 7% of the spreadsheet developers sampled called the importance of their spreadsheets low.

In another early study, Gable, Yap, and Eng [16] had spreadsheet developers and users rate all 402 spreadsheets saved on the corporate server. Respondents rated 96% as important or very important to themselves and 46% as important or very important to the organization. Furthermore, respondents said that 96% of the spreadsheets would have to be rebuilt if lost. Of the spreadsheets on the server, 13% were used daily, 6% weekly, and 40% monthly. At the other extreme, only 9% were rarely or never used.

In yet another early study, Wilkins [44] asked the broadest level at which a spreadsheet application’s data was used. Table 2 shows that few spreadsheets were limited to personal productivity.

Table 2: Broadest Level of Data Use

<table>
<thead>
<tr>
<th>Broadest Level of Data Use</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other departments</td>
<td>69%</td>
</tr>
<tr>
<td>User’s department</td>
<td>26%</td>
</tr>
<tr>
<td>Only user</td>
<td>5%</td>
</tr>
<tr>
<td>Other</td>
<td>1%</td>
</tr>
<tr>
<td>Total (includes rounding error)</td>
<td>101%</td>
</tr>
</tbody>
</table>

Source: Wilkins [44]

Chan and Storey [7] asked a sample of spreadsheet users a slightly different question—“What was the highest level in the organization in which an application was used?” As Table 3 shows, the highest level of use was extremely high. Over half produced data used by the CEO or a vice president.

Table 3: Highest Level of Data Use

<table>
<thead>
<tr>
<th>Highest Level of Data Use</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEO</td>
<td>42%</td>
</tr>
<tr>
<td>VP</td>
<td>13%</td>
</tr>
<tr>
<td>Department head</td>
<td>20%</td>
</tr>
<tr>
<td>Self</td>
<td>14%</td>
</tr>
<tr>
<td>Other</td>
<td>10%</td>
</tr>
<tr>
<td>Total (includes rounding error)</td>
<td>99%</td>
</tr>
</tbody>
</table>

Source: Chan and Storey [7]

These studies were done some time ago. Spreadsheets are undoubtedly far more widespread today, and a smaller fraction may be as important as these data indicate. However, it is impossible to say more because little research like these studies has been done recently. In addition, even if the fraction of highly important spreadsheets is lower today, the enormous number of spreadsheets found in recent censuses indicates that there probably is a large number of very important spreadsheets in corporations currently.

2.6.2. Mission-critical applications

A second indication of the importance of spreadsheets is the prevalence of mission-critical spreadsheets that would do considerable harm if they were lost. In recent years, two studies have looked at mission-critical applications in depth. Both Grossman, Mehrotra, and Ozel [19] and Wagner [43] discovered a great deal about these applications. Perhaps the most important finding, however, was that these researchers had no difficulty locating managers and professionals with mission-critical applications. This indicates that mission-critical applications are common.

2.6.3 Spreadsheet dominance over packaged applications

A third indication that spreadsheet applications are important is that they dominate application use in many application categories in which packaged application software is available. For instance, about a year before the financial crisis, a software vendor selling an application for creating derivatives called the first author saying that nearly everyone in financial services was creating derivatives using spreadsheets. The author receives calls from vendors in application categories several times per year telling similar stories. In many cases, spreadsheet applications may be developed because finding and evaluating commercial applications is expensive compared to the creation of a spreadsheet application. In addition, commercial packages often are difficult or impossible to customize, leaving out features important to users or forcing users to work in ways they find unnatural. Of course, the “not invented here” syndrome is often at work. However, it seems presumptuous to assume that users who build their own spreadsheet applications instead of using computer tools are irrational.

2.6.4. Other indications of importance

There are many other indications that spreadsheet applications are extremely important. For example, Croll [11] studied spreadsheet use in the City of London (the financial district of London). He concluded that “the City of London runs on spreadsheets.” He noted that in an environment that routinely handles transactions worth hundreds of millions to billions of dollars, nothing of importance
happens without passing through a spreadsheet. Hinh, Lewicki, and Wilkinson [23] at the Jet Propulsion Laboratory discussed the importance of spreadsheets at JP in a paper titled, “How Spreadsheets Got Us to Mars and Beyond.”

2.7. Perspective on size, complexity, and importance

Overall, there are many indicators that a large number of spreadsheets are large, complex, and extremely important. The evidence is less than completely satisfying because it represents many individual studies without any study being completely convincing. However, the most important fact about this research is its convergent validity. Every study, without exception, that has looked at size, complexity, and importance has found spreadsheets at the high ends of these concerns.

Every study, without exception, that has looked at size, complexity, and importance has found spreadsheets at the high ends of these concerns.

In addition, we have only looked at data on spreadsheets. Although spreadsheets are critical in end user application development today, there are also end user database management systems (especially Access and MySQL) and other end user tools for developing applications. Our focus on spreadsheets has merely been a consequence of the fact that spreadsheets have received the most research attention, as sparse as that attention has been.

3. Conclusion

End user computing really does seem to be the dark matter and dark energy of corporate IT. We believe that this fact has profound implications for managers, IT practitioners, and IT researchers.

3.1 Errors

One of the most heavily studied aspects of end user computing development research has been research on spreadsheet errors. The first author’s spreadsheet research website [34] summarizes some of the research in this area.

The results are disquieting. Spreadsheets typically consist of multiple long chains of calculation. Consequently, a single error in a spreadsheet is likely to produce an incorrect bottom-line value. The research indicates that developers make errors in about 2% to 5% of all formula cells at the end of module development. This would suggest that nearly all large spreadsheets are incorrect, and inspections of operational spreadsheets bear out this expectation.

Table 4: Percentage of Incorrect Spreadsheets in Audits of Operational Spreadsheets

<table>
<thead>
<tr>
<th>Study</th>
<th>Percentage of spreadsheets with errors</th>
<th>Percentage of cells with errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hicks</td>
<td>100%</td>
<td>1.2%</td>
</tr>
<tr>
<td>Coopers &amp; Lybrand</td>
<td>91%</td>
<td>Did not report</td>
</tr>
<tr>
<td>KPMG</td>
<td>91%</td>
<td>Did not report</td>
</tr>
<tr>
<td>Lukasic</td>
<td>100%</td>
<td>2.2%, 2.5%</td>
</tr>
<tr>
<td>Butler</td>
<td>86%</td>
<td>0.4%</td>
</tr>
<tr>
<td>Clermont, Hanin, &amp; Mittermeier</td>
<td>100%</td>
<td>1.3%, 6.7%, 0.1%</td>
</tr>
<tr>
<td>Lawrence &amp; Lee</td>
<td>100%</td>
<td>6.9%</td>
</tr>
<tr>
<td>Powell, Baker &amp; Lawson, 2008</td>
<td>86%</td>
<td>0.9%</td>
</tr>
<tr>
<td>Powell, Baker &amp; Lawson, 2009</td>
<td>44%</td>
<td>Did not report</td>
</tr>
<tr>
<td>Overall</td>
<td>84%</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Testing

However, few corporations believe that spreadsheet errors are important. They rarely test their spreadsheets extensively [6,10,12,14,15,17,18,21,20,30,40,43]. Few companies have policies that require testing [5,6,10,14,18,20,41,43], and among those that do, compliance is low [17, 20].

Ironically, because companies rarely test their spreadsheets, they do not see the extent of errors. In a vicious cycle, they do not see the need to do extensive testing.

One might think that significant errors would be visible even without testing, but unlike 3GL programs, spreadsheet models rarely “crash” if they are wrong. They merely give the wrong answers. In many cases, wrong answers lead to incorrect decisions. However, losses are typically invisible, consisting only of opportunity costs when bad choices were made.

This era of benign neglect toward errors may be coming to a close. As noted earlier, a number of regulatory compliance regimes, such as Sarbanes-Oxley, require that corporations increase their control over IT, and regulators and companies are beginning to understand how integral spreadsheets are to many operational processes. In addition, we are beginning to see a rise in adversarial forensic testing, in which adversaries examine the spreadsheets of their opponents to find flaws. Even relatively small errors can discredit spreadsheets in negotiations.

Spreadsheet error rates at the unit module level are about the same as statement error rates in 3GL...
programming, so spreadsheet development tools per se are not the problem. The problem appears to be poor application development, especially an almost complete lack of comprehensive testing.

Fault rates in programming (in programming, defects in the program are called faults) are very similar to error rates in spreadsheet models, as are detection rates in inspection [34].

Programmers, knowledgeable about fault rates spend a great deal of time on testing. In a sample of 84 projects in 27 organizations, Jones [24] found that the amount of time spent in testing to reduce errors ranged from 27% to 34%, depending on program difficulty. In every case, furthermore, subjects reported that insufficient time was allotted to testing. In another study, Kimberland [25] found that Microsoft software development teams spent 40% to 60% of their total working time in testing.

Furthermore, when spreadsheet users do what they call testing, what they often do is a pale imitation of what programmers do when they test. In the Deloitte webinar mentioned earlier, nearly all of the 800 respondents said that they do testing. Yet only 12% tested all cells, and only 2% used multiple inspectors to examine all cells. In software development, inspecting all statements is considered to be critical [45].

Testing in programming is not simply one of many controls. It is the main control. Although good practice in defining, designing, and developing programs are all important, the residual error rate after good development still requires extensive testing.

3.3. Other concerns

End user applications raise other concerns. One is privacy. Too many application files contain personally identifiable information that companies have an obligation to protect. The leakage of even a single spreadsheet containing such information can be a disaster for a firm.

Another concern is security. In recent years, hackers have increasingly focused on stealing trade secrets. Many organizational processes and much essential knowledge are contained in spreadsheets. Spreadsheet loss can seriously harm a firm’s competitive position.

A third risk is fraud. In 2002, Allfirst Corporation discovered that its currency trader, John Rusnak, had lost $691 million in trades over a period of several years. A spreadsheet was critical in keeping his growing losses hidden [4]. Bernie Madoff also used a spreadsheet to maintain his fraud [1].

More generally, corporations are finding themselves subject to an increasing number of compliance requirements. Some requirements, like 21 CFR 11 for the pharmaceuticals industry, explicitly mention spreadsheets. Others, such as Sarbanes-Oxley, do not mention spreadsheets explicitly but do require anything that might create material errors in financial reports to be controlled. Eventually, the PCAOB did note that spreadsheets had been under-considered in previous years and needed to be considered explicitly.

3.4. End user support

Corporations have attempted to support end user computing through help desks that deal with low-level information about individual applications. This is inexpensive, but it does not help end users learn how to develop applications well or how to be more productive and effective while they work. Nor does it give any corporate oversight.

Either IT or another corporate staff will have to take the lead in helping users control and develop their applications more effectively. This requires a higher level of focus and will need a radical restructuring in user support.

3.5 Descriptive research

One reason that EUC has been overlooked is that IS researchers have generally failed to do research on spreadsheets and other end user application development practices. CIOs have also tended to be blind to EUC and its implications. When a CIO steps out her office door, she sees programmers, systems analysts, networking professionals, security professionals, database administrators and the other personnel critical to centralized applications. Regarding the rest of the firm’s IT, she may have strong feelings about what is going on but little knowledge. In one case, a CIO reported that no spreadsheets were in use after an Enterprise Resource Planning system was put into place. The researcher [43] he said this to had already interviewed employees and discovered that there were many spreadsheets in place, some of which users regarded as mission-critical.

The very diversity of employees and work units in real firms makes it difficult for anyone to know what is really going on overall. As an academic field, we need to be willing to do and publish the “D Word”—descriptive research. Focused research is good if you know where to look, but if you do not have an excellent understanding of the landscape, you may not be doing the kinds of focused research our reference discipline in industry needs us to do.

In marketing, there is an important distinction between market research and marketing research.
Market research is research on the size and structure of a market, including market segments. Marketing research addresses a specific question. Marketers have long known that unless you understand the market broadly, narrowly focused marketing research to answer a specific question is not likely to be productive because it may be aimed at the wrong question or at least at a meaningless question.

3.6 User research

Information systems researchers have not done a great deal of research on users. In today’s IT world, user research has the potential to be extremely exciting. Users are creating their own applications, and they are adapting packaged applications in ways that developers never anticipated. To borrow a marketing term, the user is the familiar unknown. We see them every day, but we do not really understand them. That is sad, because users are an exciting group to study.

3.7 Perspective

In this paper, we have looked at data on the importance of end user computing and the size and complexity of end user development. The data is remarkable because of its almost perfectly convergent validity. Although the data come from many studies using different methodologies and different specific protocols, they are completely convergent in their findings. Put another way, there is no data indicating that the popular view of end user computing is a matter of simple personal productivity applications.

References


[28] Lukasik, Todd, CPS. (1998, August 10). Personal communication with the first author.


