Communications of the Association for Information Systems

Volume 20 Article 60

December 2007

Lessons from Mission-Critical Spreadsheets

Thomas A. Grossman
University of San Francisco, tagrossman@usfca.edu

Vijay Mehrotra
San Francisco State University

Özgür Özlük
San Francisco State University

Communications of the Association for Information Systems

LESSONS FROM MISSION-CRITICAL SPREADSHEETS

Thomas A. Grossman
University of San Francisco
tagrossman@usfca.edu

Vijay Mehrotra
San Francisco State University

Özgür Özlük
San Francisco State University

ABSTRACT

We present eighteen examples of mission-critical spreadsheets used by diverse people and organizations for application software development, financial risk management, executive information systems, sales and marketing business processes, business operations, and complex analytics. We argue the spreadsheet is a Rapid Development Language, an Integrated Development Environment, and a Fourth Generation Language, and has unusual challenges regarding source code protection. We note that intentional spreadsheet applications are largely absent from the error literature. We explain why people might prefer a spreadsheet to an application developed by the IT department, and show how some spreadsheet programmers choose to avoid—or do not have—an IT department.

We find that 1) Spreadsheets are widely used for mission-critical functions; 2) Spreadsheets are an effective application development platform; 3) There is diversity of development skill in creators of mission-critical spreadsheets; 4) Sophisticated programmers sometimes choose spreadsheets over other languages; 5) Spreadsheets are amenable to formal development practices, but such practices seem rare; 6) Spreadsheets play a central role in the evolution of business processes and work systems; and 7) Spreadsheets are a source of "accidental legacy systems". We provide the Skill-User Programming Paradigm to help interpret and explain our observations. We conclude that spreadsheets are vitally important to business, and merit sustained research to discover techniques to enhance quality, productivity, and maintainability.

Keywords: spreadsheets, empirical research, information systems

PART 1: EIGHTEEN CASE STUDIES

I. INTRODUCTION

The business world and the academic world have different perceptions of spreadsheets. Since the invention of VisiCalc (the first computerized spreadsheet) in 1979, spreadsheets have become ubiquitous in the business world, and are widely acknowledged to be "the oldest killer-app of them all" [Liebowitz 2002] that drove the growth of personal computing. Walk into any large bookstore, and you will find shelves displaying fast-selling books with titles such as *The Excel Bible*. To the extent that quantity matters, the numbers are eye-opening. Bach [2007] reports that some banks are wrestling with 15 million spreadsheets, and at the 2007 EuSpRIG

conference we received a verbal report of a single server containing more than 1 million spreadsheets that had been touched within the previous 30 days.

The ubiquity of spreadsheets in business is in marked contrast to their near absence from academic research. An October 2006 keyword search on "spreadsheet" at CAIS (http://cais.isworld.org/app/search.asp) returns but three papers on spreadsheets, of which one is a historical study, one a conference panel discussion, and only one a research paper [Panko 2006].

There seems to exist a perception that spreadsheets are somehow different than other programming tools and that spreadsheets are suitable for personal use but not for important tasks. The IS 2002 Model Curriculum [Gorgone, et al. 2003, p. 12] promulgated by The Association for Computing Machinery, The Association for Information Systems, and The Association for Information Technology Professionals identifies spreadsheets as "fundamental tools of personal computing," a list that encompasses "e-mail, Web browsing, spreadsheets, word processing, desktop database management systems, presentation graphics, and external database retrieval tools," suggesting that spreadsheets are simply personal productivity tools, of equivalent (un)importance to information systems as word processors or PowerPoint. A leading textbook [Senn 2004, p. 60, italics added for emphasis] states that "*Information systems differ in an important way from spreadsheet packages*. . . Information systems focus on business processes. . . while [spreadsheets] focus on solving problems, aiding in personal decision making, and increasing personal productivity," indicating that spreadsheets are not information systems and implying that spreadsheets do not merit the attention paid to information systems. The popular software engineering book [McConnell 1996, p. 516, italics added for emphasis] mentions "domain-specific tools such as spreadsheets. . . that solve problems that would *otherwise have to be solved by creating a computer program,*" suggesting that a spreadsheet is not a computer program.

Our findings indicate that these perceptions are sometimes inappropriate. Spreadsheets can be the central software component of mission-critical systems, not mere personal productivity tools. Some spreadsheets focus on business processes and seem to fit Senn's definition of information systems. Spreadsheets are most certainly computer programs.

The research literature on spreadsheets is small. Spreadsheets are conspicuous by their absence from academic journals and conferences in information systems and computer science. In short, spreadsheets are almost invisible in the information systems and computer science research communities.

The purpose of this research is to make visible in the academic literature the vital importance of spreadsheets across multiple sectors of the economy and to illuminate the diversity of usage of important spreadsheets. It is hoped that increased awareness of the importance and complexity of spreadsheets and their use will stimulate academic research on important questions of practical value to business. There is clear potential to generate significant benefits by developing improved methodologies for many of the very important activities performed by millions of people who interact with spreadsheet information systems.

SURVEY OF THE EMPIRICAL LITERATURE

This paper adds breadth to the small empirical literature on spreadsheets. [Panko 2006] summarizes the empirical literature from the 1990s (there seems to be none earlier) regarding the importance of spreadsheets. He cites eight papers and an industry survey and concludes that "many spreadsheets are large, complex, and very important to their firms."

The Sarbanes-Oxley Act of 2002 has stimulated empirical research on spreadsheets in finance. Croll [2005] discusses important spreadsheets within the domain of the UK financial center known as the City of London. Croll concludes that, "Within certain large sectors, spreadsheets play a role of such critical importance that without them, companies and markets would not be able to

operate as they do at present." Panko [2006] cites four financial industry surveys in 2004 and 2005 and indicates that spreadsheets are used in critical financial reporting applications. In addition, during a Webcast Panko surveyed just over 800 financial professionals and officers in corporations and discovered that 88 percent answered affirmative to the question "Does your firm use spreadsheets of *material importance* in financial reporting?"

OVERVIEW AND CONTRIBUTION

We consider this work to be exploratory research in a new area. We divide our results into two parts. Part 1 presents the 18 case studies and their classification. Part 2 synthesizes our findings from these case studies, raises questions, proposes hypotheses, and provides a research agenda.

Part 1 shows mission-critical spreadsheets across industry, size of firm, and type of usage. It also provides a classification of mission-critical spreadsheet usage consisting of six categories: application software development, financial risk management, executive information systems, sales and marketing business processes, business operations, and complex analytics.

Part 2 provides seven major findings; a multidisciplinary analysis integrating theoretical and empirical insights from the disciplines of information systems, computer science, spreadsheet risks, management science, and ethnology; and an extensive research agenda that we hope will inspire and guide future researchers. Highlights include a new skill-user programming paradigm which generalizes the traditional end-user programming paradigm; and the discovery of accidental legacy systems which are systems intended for personal use that are inherited by a person when they take a new job. Our major findings are: 1) Spreadsheets are widely used for mission-critical functions; 2) Spreadsheets are an effective application development platform; 3) There is diversity of development skill in creators of mission-critical spreadsheets; 4) Sophisticated programmers sometimes choose spreadsheets over other languages; 5) Spreadsheets are amenable to formal development practices, but such practices seem rare; 6) Spreadsheets play a central role in the evolution of business processes and work systems; and 7) Spreadsheets are a source of "accidental legacy systems."

The remainder of Part 1 is organized as follows. In Section II we discuss our methodology for gathering data from informants who use mission-critical spreadsheets. Section III contains a classification of the spreadsheets based on usage. In Section IV we present 18 spreadsheet case studies.

II. METHODOLOGY

We interviewed people who use spreadsheets for mission-critical purposes. We sought interesting examples by asking contacts in industry about their use of spreadsheets, or people they knew who used spreadsheets. We had no difficulty identifying research informants. Everyone we approached agreed to be interviewed, and informants were generally eager to speak with us. Most interviews were conducted on the phone, with a few conducted in the informants' place of work. In a few cases we called back with clarifying questions. Most interviews were recorded on audiotape. A few interviews were not taped, and we rely on our contemporaneous notes of the interview. We showed each quotation in this paper to the research informant and received confirmation that it was accurate.

We performed these interviews under human-subjects research protocols approved by our universities, including a mandatory "informed consent" document. Our research protocols require that we not collect sensitive information, divulge personally identifiable information, nor reveal the name of the informants' organizations. To enhance confidentiality, all personal pronouns are masculine.

We sought a diverse group of informants. Our informants work in companies ranging from four employees to the tens of thousands. They come from corporations, family business, and the

nonprofit sector. Our informants range from CEO to front-line analysts. Their industries include financial services, consulting, high-tech, heavy industry, travel and hospitality, software, and personal services. Informants work in many functional areas including information systems, manufacturing, marketing, finance, operations, and sales.

Interviews were conversational, guided by a list of questions. We allowed the discussion to roam freely but attempted to cover all the questions. During the first part of the interview, we asked informants demographic questions, including the role of spreadsheets in their current positions, their own experience and training with spreadsheets and other types of software development, and their experience with spreadsheet complexity and errors.

During the second part of the interview, we asked informants to identify a specific spreadsheet where "if the spreadsheet were destroyed you could not do your job, or there would be a very significant negative impact on the organization." We then had an in-depth discussion of this mission-critical spreadsheet.

We asked the informant to explain what the spreadsheet was used for, why it was so important, how the need fulfilled by the spreadsheet had been filled in the past, and what would happen if its contents were to become unavailable. We discussed the origin, use and history of the spreadsheet, including the organization's motivation to develop it and the way the spreadsheet was created, modified, and used; the integration of the spreadsheet with other data systems and business processes; the spreadsheet's users and frequency of use (intended at time of creation and actual); the role of the IT department in the design, development, support, and maintenance of the spreadsheet; and the individual's and organization's process for creating, testing, and maintaining the spreadsheet. In addition, research informants provided substantial additional information that was not on our list of questions.

III. CLASSIFICATION OF MISSION-CRITICAL SPREADSHEETS

We found wide diversity in the way people use spreadsheets. To provide a structure to understand this diversity, we organized the cases into six categories based on type of use.

- *Spreadsheet Used for Developing Application Software*: This category groups the cases where the development team creates a spreadsheet for the benefit of end users. *Application software* is "programs written for a specific application to perform functions specified by end users" [Laudon and Laudon 2006]. Despite a widespread perception that application software is usually created using procedural programming languages or object-oriented development environments, we found that some developers choose to create application software in spreadsheets.
- *Spreadsheets Used for Financial Risk Management*: This category of spreadsheets is used for financial purposes. Problems with these spreadsheets can lead to serious financial losses.
- *Spreadsheets Used for Executive Information Systems*: This describes the type of spreadsheets that enables executives to understand their business and make important decisions. An executive information system is a "computer-based system intended to facilitate and support the information and decision making needs of senior executives by providing easy access to both internal and external information relevant to meeting the strategic goals of the organization" [Wikipedia 2005].
- *Spreadsheets Used for Sales* and *Marketing Business Processes*: Such spreadsheets serve as an integral part of an essential business process that convinces a potential customer to purchase a product or service.
- *Spreadsheets Used for Business Operations*: These spreadsheets play an integral part in business operations, where the spreadsheet is essential to the efficient provision of a product or service to a customer.

- *Spreadsheets Used for Complex Analytics*: This category of spreadsheets is used for spreadsheet for advanced scientific or engineering analysis.

Some of the case studies contain elements of more than one category, and we placed them in the category with the strongest fit. In particular, we limit the category of "application software" to situations where the users have essentially no interaction with the developers. Table 1 provides a general overview of the case studies.

Table 1. Overview of Case Studies on Spreadsheet Uses

Classification	Company Pseudonym	Company Description	Spreadsheet Use
Spreadsheets Used for Developing Application Software	1: SmallSoft	Privately held small software start-up	Application Software for Sale
	2: AnalyCo	Large consulting firm	Application Software for Internal Use
	3: ConsultCo	Small supply chain management consulting firm	Decision Support System for Client Use
Spreadsheets Used for Financial Risk Management	4: BigSoft	Large publicly traded international software company	Foreign Exchange Hedging
	5: I-Bank	Major U.S. investment bank	Bond Underwriting
Spreadsheets Used for Executive Information Systems	6: MajorCorp	Fortune 50 company in capital-intensive industry	Capital Investment Performance and Decision Making
	7: AssetFund	Large Wall Street financial firm	Evaluating Investment Strategies
	8: Shrinksoft	S&P 500 corporation selling shrink-wrapped software	Understanding Customer Perceptions
Spreadsheets Used for Sales & Marketing Business Processes	9: DevCo	Small, publicly traded software corporation	Software Project Cost and Time Estimation
	10: CommCorp	Publicly traded Web-based communications services company	Lead Generation and Sales Development
	11: DealCo	Small publicly traded company	Contact Management
Spreadsheets Used for Business	12: Archiva	Small professional services company	Managing and Interpreting Qualitative Data

Classification	Company Pseudonym	Company Description	Spreadsheet Use
Operations	13: CompEng	Small Engineering Firm	Project Planning and Management
	14: CleanCo	Family business providing dry cleaning etc.	Tracking Productivity
	15: NonProfOrg	Small nonprofit organization helping disabled individuals	Cost Allocation
Spreadsheets Used for Complex Analytics	16: TechStart	Technology start-up firm	GPS Algorithm Development
	17: PowerCo	Large electric power provided	Pricing Power Contracts
	18: GroupHotel	Large business hotel	Hotel Revenue Management

IV. CASE STUDIES: EIGHTEEN MISSION-CRITICAL SPREADSHEETS

In this section, we provide 18 case studies of mission critical spreadsheets. We describe the business, discuss the essential business activity that is enabled by the spreadsheet, describe the spreadsheet and its use, and indicate how the spreadsheet was created. Quotations are the words of our research informants.

1. SMALLSOFT: APPLICATION SOFTWARE FOR SALE

SmallSoft is a privately held software start-up with four employees. They sell a software application package for compiling a business case. This application is implemented as an Excel spreadsheet, with a couple of dozen sales to date. The product is implemented as three tightly linked spreadsheets with a total size of 73 megabytes, accompanied by a 150-page training manual. This spreadsheet product is the company's only source of revenue. It is priced at $30,000, plus with ongoing fees for technical support and maintenance releases.

The CEO and software architect of SmallSoft is a former software product developer with successful software start-up experience. He created the product over a series of consulting engagements. At each round of changes, the CEO "carefully architected" the spreadsheet, sometimes necessitating extensive recoding; formally designed the user interface; and followed "disciplined coding practices." He chose to implement the product in a spreadsheet rather than in traditional software due to "the rapid way that we had to change it because we were learning from the customers as fast as we had our ideas." SmallSoft is now porting their product to a third-generation language because the sheer size of the spreadsheet program was causing Excel to fail.

2. ANALYCO: APPLICATION SOFTWARE FOR INTERNAL USE

AnalyCo is a large consulting firm that provides analytic and organizational development services to customers around the globe. One popular service is a large scale study of the attitudes and behaviors of client's customers, prospects, and/or employees. A typical study considers thousands of individuals. AnalyCo performs extensive analysis and compares the results to those

from similar companies. They interpret the results and share them with executives, managers, and employees within the client organization.

The bulk of the data analysis is done by scientists in a centralized group, utilizing a statistics application software package and special software to populate and query AnalyCo's database. At the conclusion of the analysis, field consultants examine the results and provide detailed output to clients. Unfortunately, this final project phase was an "exhausting race to print thick binders" containing thousands of pages of results, followed by a "tedious, expensive and painful" combing of these printouts to prepare multiple client presentations. When clients asked questions, the field consultants found it very difficult to conduct even simple follow-up analysis due to the structure of the data and difficulty accessing it. These were serious problems, inhibiting AnalyCo's ability to make full use of its data and harming client satisfaction.

A group of field consultants obtained internal resources to develop new application software called the "proprietary calculator." They prepared a set of requirements for application software that would enable AnalyCo field consultants to access and analyze the results generated by the central scientific group. The software applications team at AnalyCo's headquarters developed the software as a client-server system using Excel as the client. The applications team created a prototype and obtained feedback from the field consultants on bugs, workflow issues and possible enhancements. They performed formal testing and created detailed documentation before distributing the Proprietary Calculator template to AnalyCo offices worldwide.

Today, this spreadsheet-based application software is used several times a day by thousands of people all over the world. "Our executive presentations to clients depend on the content of these worksheets," said the project manager, "and we would lose significant credibility if we presented incorrect results. We're able to respond very, very quickly to follow-up questions as a result of the spreadsheet's capabilities, and we've won millions of dollars of additional consulting services as a result."

3. CONSULTCO: DECISION SUPPORT SYSTEM FOR CLIENT USE

ConsultCo is a small supply chain management consulting firm that develops and implements sophisticated mathematical models for forecasting and optimization. A large retailer of fashion goods contracted with ConsultCo to deliver a system to provide store managers with guidance on marking down prices as fashion goods reach the end of their life. The revenue from such markdowns represents a significant portion of the retailer's profit, so the timing and amount of the markdowns are essential decisions.

The client stipulated that ConsultCo's solution must be easy to distribute to its stores, very straightforward for store personnel to use, and provide robust pricing recommendations. ConsultCo chose to implement and deliver this decision support system in an Excel spreadsheet, with live feeds from transaction processing systems via SQL queries. The client paid a price "in the six figures" for this spreadsheet.

The spreadsheet decision support system was a major success for its client, increasing profitability by better managing prices and inventory during the product lifecycle. This success was instrumental to ConsultCo's establishment of their product in the marketplace. ConsultCo ported the spreadsheet system to a third-generation language to improve performance, and because prospective customers balk at spending hundreds of thousands of dollars on a spreadsheet, although they "have no qualms spending the same amount for a production-quality piece of software."

4. BIGSOFT: FOREIGN EXCHANGE HEDGING

BigSoft is a large publicly traded international software company. BigSoft has foreign currency revenue in the hundreds of millions. To minimize financial risk from fluctuations in exchange rates, they routinely hedge their foreign exchange exposure. Hedging allows them to "lock in"

foreign exchange rates to control variability in the revenue numbers that are reported to Wall Street.

BigSoft's assistant treasurer has responsibility for foreign exchange hedging. He uses a spreadsheet on a monthly basis to manage the information that is sent to the bank that executes their hedging contracts. The spreadsheet takes data from multiple sources and performs a complex series of calculations to determine the proper hedging values. Without the spreadsheet, BigSoft would not be able to determine the details of its hedging plan.

The assistant treasurer inherited the spreadsheet from his predecessor. At the time that the Euro currency supplanted most western European currencies, the assistant treasurer's predecessor's predecessor modified the spreadsheet to include the Euro. This modification resulted in a subtle error that intermittently computed incorrect hedging values. After a cell-by-cell code inspection and other work, the assistant treasurer repaired the spreadsheet, and concluded that the error resulted in a direct financial loss of approximately $3 million.

5. I-BANK: BOND UNDERWRITING

I-Bank is a major U.S. investment bank that underwrites financial transactions for its clients. One important line of business is underwriting municipal bonds, with its transactions typically valued in the tens of millions of dollars. It is vital that the bond's structure and pricing be attractive to the client selling the bond and to investors who will ultimately buy the bond. Because a default (inability to pay back the investors) by the municipality has a major impact on the municipality, bondholders, and I-Bank, it is vital that I-Bank determine that the bond is appropriate for the client's unique financial circumstances. Therefore, throughout the underwriting process, I-Bank performs extensive financial analysis customized to the many unique features of each bond deal.

For each deal, an analyst creates a spreadsheet model in Excel. The process of data acquisition, spreadsheet development, and use of the model is continuous and iterative. Analysts use the model throughout the underwriting lifecycle such that "the entire underwriting of the deal depends on the content of the model."

The spreadsheet model serves several essential functions: it is a repository of the data inputs and sources; a living record of the assumptions made in the underwriting process; and a vehicle for implementing complex financial calculations. Upon completion of the deal the model is retained on a server as a lasting record of the analysis that was behind it. The spreadsheet model is so tightly integrated into I-Bank's business processes that one analyst told us "I have no idea how we could run our business today without the model spreadsheets."

6. MAJORCORP: CAPITAL INVESTMENT PERFORMANCE AND DECISION MAKING

MajorCorp is a Fortune 50 company in a capital-intensive industry with operations around the world. Senior management and stock market analysts pay particular attention to the financial return on capital and the political risk associated with capital invested around the world.

As part of their annual investment planning cycle, MajorCorp's top executives require detailed information on the geographical distribution over time of earnings, cash flow, capital expenditure and capital investment. A planning analyst uses the Long Term Geographic Distribution of Earnings, Production and Capital (LTG) spreadsheet to conduct the analysis needed to support this requirement.

The LTG spreadsheet consists of more than 10MB of data, along with extensive calculations that describe the distribution and financial performance of MajorCorp's $50 billion of deployed capital. The analyst who owns the LTG spreadsheet uses it for several different purposes, most notably to generate a report that plays a vital role in the annual capital planning cycle. The analyst inherited the spreadsheet from his predecessor, and then reprogrammed it from scratch while adding new functionality.

The MajorCorp Strategy and Planning Committee, which consists of the top 14 executives including the chairman of the board, uses the output of LTG as "part of the fabric of evaluating where we are and where we're going." This information plays an important role in investment decisions, and a large error in the spreadsheet would "alter the capital spending pattern and approval of capital projects." Some of the information from LTG is shared with Wall Street analysts who influence the company's share price.

7. ASSETFUND: EVALUATING INVESTMENT STRATEGIES

AssetFund is a large Wall Street financial firm. One division of the firm manages several portfolios of bonds and related financial instruments for major institutional clients. To remain competitive, AssetFund must devise new ways to produce superior returns for its clients: "Client capital is very portable and the markets are more efficient every day . . . we can't afford to sit still." Therefore, the firm employs a research team to develop and test new investment strategies, which they refer to as "bets." Portfolio managers are constantly examining and changing the allocation of assets in the context of different strategies.

AssetFund has accounting systems that are used to execute and track transactions and to provide timely information on portfolio assets and performance. However, the executive team, the head of institutional research, and the portfolio managers were unable to connect their decisions and investment strategies to the resulting contributions in each portfolio. This lack of visibility limited their ability to evaluate the performance of different investment strategies: "We were continually getting questions about different components of the overall return, but we didn't really know how well which bets were paying off."

The head of the research team charged a senior analyst to develop a Performance Attribution Model (PAM) that he could use to effectively estimate and allocate portfolio-level returns to different bets within each portfolio. The PAM spreadsheet receives raw data from seven mainframe production information systems, performs data pre-processing, and implements a proprietary algorithm to compute the desired outputs. The results are distributed to senior management and other stakeholders on a daily basis through the PAM spreadsheet. In turn, this information is a key source of feedback and guidance to portfolio managers and to the research team as they allocate assets.

8. SHRINKSOFT: UNDERSTANDING CUSTOMER PERCEPTIONS

ShrinkSoft is an S&P 500 corporation that sells shrink-wrapped software products and professional services through multiple channels. Shrinksoft is the market leader. They have a very large customer base and a strong record of profitability and growth.

ShrinkSoft's senior executives are committed to implementing the customer perception survey concepts presented in Reichheld [2003]. They believe that they must obtain a detailed, quantified understanding of how customers perceive the company, and how loyal they are to the company and its products and services. ShrinkSoft acquires information by inducing customers to give feedback via Web-based surveys on a regular basis.

To provide senior management with the necessary understanding of customer perceptions and loyalty, the division's chief operating officer directed an internal consultant to develop a tool that can be used to quantify and track customer attitudes. He used Excel to develop the Net Promoter Correlation Analysis (NPCA) spreadsheet. An analyst downloads survey data and uses NPCA to analyze it daily. Senior managers review NPCA reports weekly.

Shrinksoft bases most of its decisions about major new initiatives on NPCA reports. New value-added services are often developed to meet needs that are identified through NPCA. Product managers facing critical choices between different product features and bug fixes use NPCA results to support design decisions. Shrinksoft uses NPCA outputs to develop customer

communication campaigns designed to increase customer loyalty, and to measure the effectiveness of those campaigns.

9. DEVCO: SOFTWARE PROJECT COST AND TIME ESTIMATION

DevCo is a small, publicly traded software corporation that performs custom software development and technology consulting. Software project management is a core competency, and they believe they manage and deliver software projects particularly well.

DevCo bids for software development projects by offering the customer a fixed price for developing a specified piece of software. DevCo determines the bid price by estimating the cost to develop the software, multiplied by a standard number to provide for profit.

DevCo requires accurate cost estimates in a short period of time. Accuracy is essential, because under- and over-estimates cause serious problems. “If we miss on the low side, we take it on the chin” and incur a financial loss, whereas overly high-priced estimates run the risk of losing the customer’s business. DevCo’s sales team must wait for the cost estimate to complete its proposal to each prospective customer, so rapid estimation is essential.

DevCo creates all of its estimates using a software project estimation tool (SPET), which is a spreadsheet template. The SPET spreadsheet captures “all knowledge in the company about how to perform a project estimation.” The only process documentation of the way DevCo estimates projects is instructions contained in the spreadsheet. DevCo’s sales process is completely dependent on the SPET spreadsheet. The project’s software architect develops and maintains the spreadsheet for a particular estimate. The spreadsheet provides the “foundation for project management,” and represents the “project plan and all the thinking that went into it.” In essence, DevCo’s estimation process is so closely connected with the SPET spreadsheet that the two can not be separated.

10. COMMCORP: LEAD GENERATION AND SALES DEVELOPMENT

CommCorp is a publicly traded company that provides Web-based communications services to corporations and small businesses. An essential part of their strategy is to maintain their existing customer base while acquiring new customers.

CommCorp’s “whole sales process and business model is lead-driven, so everyone in the company is intensely interested in understanding how we are doing with lead generation and development.” CommCorp devotes significant resources to stimulating demand via lead generation activities, and yet the company, which had grown much faster than its IT infrastructure, had struggled to gather information about which efforts were producing what quality of results. CommCorp staff regularly engaged in “fire drills to madly chase reports from multiple systems and decipher them” in order to develop information on their sales process.

In order to meet the constant demand for this difficult-to-obtain information, a marketing operations manager created a spreadsheet called the Lead Performance Analysis Workbook (LPAW). The spreadsheet takes live inputs from several departmental databases, including CommCorp’s sales force automation system, marketing automation database, order fulfillment system, and billing systems. LPAW analyzes the performance of the various lead generation and sales development processes and is recognized as the one source within CommCorp that can be relied upon to accurately quantify the return on specific lead generation investments.

The information created by the LPAW spreadsheet is eagerly sought by CommCorp executives and marketing managers. It is used for a wide range of management communication and decision making. The spreadsheet is reviewed at every Monday’s business operations meeting. Within an hour of the weekly distribution, the marketing operations manager receives “tons of e-mail” requesting additional clarification and detail about the LPAW results. The marketing operations manager receives requests for changes every week, and his team “puts out a major new version” of the LPAW spreadsheet approximately every three months.

11. DEALCO: CONTACT MANAGEMENT

DealCo is a small publicly traded firm that licenses medical technologies, commercializes them, and manages their distribution. Their business depends on their ability to identify, understand, and evaluate the commercial potential of new innovations, and to create effective long-term relationships with the technology developers. They invest great effort into contacting research leaders and inventors in companies and laboratories around the world. They use the expertise of many individuals in the company plus the members of their medical advisory board when interacting with potential partners. It is a lengthy process to determine which of many promising opportunities should be pursued, and to attempt to negotiate a licensing arrangement.

Initially, when nearly everyone on staff was actively pursuing new business opportunities, there was a "blizzard of memos, e-mails, and documents" being exchanged "helter-skelter" across the company. "It was crazy, and you were never quite sure if you had all the background when you called into an account. You just tried not to put your foot in your mouth."

The business need was clear. "We needed a common log of all contacts with all potential business partners to be easily shared by all core team members." DealCo executives decided to create this functionality using a spreadsheet residing on a shared drive for everyone in the company to access freely. The founder and CEO spent half a day designing and programming the spreadsheet. The spreadsheet contains no cell formulas. There was extensive discussion across the firm about the need for every individual to be "extremely disciplined" about data entry. The spreadsheet is modified "several times a day, every day!" as new information is logged and new needs emerge.

"This spreadsheet is the lifeblood of our business," says DealCo's founder and CEO. The board of directors sees the spreadsheet, and it is used by the chief financial officer, executives, and sales staff. "If we lost its contents, we would suffer a lot of pain and embarrassment."

12. ARCHIVA: MANAGING AND INTERPRETING QUALITATIVE DATA

Archiva is a small professional services company that provides information management and historical research studies for its clients. A large institution hired Archiva to write a history of a certain class of transactions that occurred over a century ago. Archiva dispatched several teams of researchers to find relevant historical documents in public archives in a large number of locations.

Researchers used spreadsheets to capture, stored, and analyze all data for this project. As relevant documents were discovered in each archive, the team entered carefully selected bibliographic and historical data into a spreadsheet. Because the delicate condition of these unique historic documents proscribed making photocopies, and photographic images were not able to record faded script, the spreadsheets are "the only record of what we found in weeks of research in remote locations." At the end of the discovery process, Archiva concatenated the teams' spreadsheets into a master spreadsheet containing approximately 4,000 records that included approximately 20,000 distinct entities. The data were entirely qualitative, and the master spreadsheet initially contained no cell formulas. (We note that a senior historian did write a lone cell formula that summed a numeric value they appended to each record during their analysis.)

Because a single transaction might be recorded at multiple dates at multiple locations, Archiva's senior historians used the master spreadsheet to perform analysis of the historical data, sorting and searching the data in different ways, seeking to identify the multiple records that referred to the same transaction. They combined records for the same transaction into a single record, resulting in approximately 2,000 independent records.

This spreadsheet, which represented all of Archiva's data gathering and analysis, became the centerpiece of their deliverable to the client. The simple and flexible sorting, searching, and display capabilities of the spreadsheet allowed more rapid analysis than would have been

possible using other means, which was essential to Archiva: “It would not have been possible to do this in the timeframe we had without the spreadsheet.”

13. COMPENG: PROJECT PLANNING AND MANAGEMENT

CompEng is a small engineering firm that designs and manufactures systems that combine advanced software with complex hardware. Its clients include the world’s largest electric utilities and engine manufacturers.

Most CompEng projects have multi-million dollar price tags. Upon project initiation, the project engineer creates a spreadsheet project management tool called “BOM.” The BOM spreadsheet started as a simple bill of materials but evolved into much more. The BOM spreadsheet serves as a project plan, with individual steps, dependencies, schedules, costs, and estimated durations; as a traditional bill of materials, with a detailed listing of the components and sub-components of the deliverable; and as a project notebook, with annotations, manufacturing process information, and embedded notes from project team members to one another.

The product development process revolves around the BOM spreadsheet. Members of the project team use the BOM spreadsheet daily, including individuals from sales, manufacturing, engineering, purchasing, and finance. It is updated constantly, and serves as the basis for weekly project review meetings. The BOM spreadsheet is linked to multiple other spreadsheets, including parts catalogs and engineering calculation worksheets that are in turn updated regularly. The final version of the BOM is maintained as a record of the inputs and outputs of the project, and provides an important resource for accounting and customer technical support.

The project engineer we interviewed felt strongly about the mission-critical importance of this spreadsheet: “Every project here has unique needs, requirements, and relationships. The BOM spreadsheet drives the whole project: many people, many parts, many dependencies. Without a very flexible way to track everything that is happening on a project, we would be at greater risk for more chaos, purchasing and billing mistakes, and embarrassing engineering problems. I really don’t know what we’d do without the BOM spreadsheet.”

14. CLEANCO: TRACKING PRODUCTIVITY

CleanCo is a family business with four locations and 20 employees that provides dry-cleaning, laundry, clothing maintenance, and pick-up/delivery service. The president of the company, like many small business owners, is involved with all aspects of the business. Employee productivity is particularly important to this business, and changes in productivity need to be identified quickly so that the president can quickly take any necessary action.

The president uses the “Weekly/Monthly Management Report” spreadsheet (WMMR) to track and manage employee productivity. He obtained the WMMR spreadsheet from a “cost club” of 12 dry cleaning businesses that provide support and training on best practices. He modified the spreadsheet to fit his specific needs at CleanCo.

Once a week the president manually transfers data from the company’s employee timecard spreadsheet and point-of-sale system into the WMMR spreadsheet. Each month, the WMMR spreadsheet produces a summary of sales by product category; a summary of productivity by category and employee; and the costs associated with recruiting, hiring, training, and terminations.

The president is not a sophisticated computer user and is uncomfortable working in Excel. The president personally updates WMMR and carefully reviews the cell formulas and data. He pays close attention to the output of the spreadsheet because it provides a “track record of where we’ve been and where we’re going,” and a “tool to keep productivity at target level.”

Before WMMR, the president had no mechanism to extract meaningful knowledge from the raw data regarding his business. Without the spreadsheet, he would not have “sophisticated tools” to

measure key metrics that require data from multiple systems, such as pieces per operator hour. If the WMMR spreadsheet were not available, "we would have to build it again right away."

15. NONPROFORG: COST ALLOCATION

NonProfOrg is a small (approximately 30 employees) nonprofit organization that provides services and information to disabled individuals. Private and government grants fund NonProfOrg's operations.

NonProfOrg faces significant financial challenges obtaining funding and complying with the rules governing the grants. Each grant has a different set of restrictions on labor costs, direct program costs and indirect costs, including the period within which the funds must be spent; specific personnel and/or programs toward which the funds must be applied; maximum amounts, types, and/or rates of indirect costs that can be charged to the grant; and billing and reimbursement procedures.

NonProfOrg's controller needed a structured method to decide which expenses to allocate to each funding source across multiple months. He inherited from his predecessor a spreadsheet called the Payroll Recap Sheet (PRS). He extensively reviewed it, finding and fixing many errors, and extended it to meet his needs. He enters information about the allowable uses of each grant. On a regular basis, he enters into PRS all labor costs, direct program costs, and indirect costs. He uses Excel's Solver add-in to perform optimization to allocate costs to funding sources while honoring the grant restrictions. He uses the PRS outputs to populate NonProfOrg's payroll and billing systems.

The controller describes their dependence on PRS: "When I started working on the PRS, I thought it would get used once a month. I now use it two-three times each week to examine the impact of different hiring and funding scenarios and can quickly answer the many 'what-if' questions that our executive director loves to ask. And at the end of each month, we rely heavily on the PRS to close out the books. It provides the basis for our contract billing and our assignment of revenues to costs."

16. TECHSTART: GPS ALGORITHM DEVELOPMENT

TechStart is a small start-up firm with less than a dozen employees. They have venture capital funding to develop advanced algorithms to generate more precise locations using the Global Positioning System. It is essential for TechStart to understand and communicate the performance of different algorithms throughout the development process, because these results drive their relationship with venture funding sources, and potential customers around the world.

The TechStart research team analyzed and compared algorithm outputs using Excel. On a routine basis, they took millions of records of output, pared them down to tens of thousands (sometimes they ran into Excel's 64k row maximum), and read the selected data into Excel. From here, engineers use Excel to perform extensive analysis of the data, and produce graphs summarizing the results.

The output of the spreadsheet was sought after by many people, including engineering, marketing, and potential customers. In some cases, TechStart needed to generate complex results in 24 hours to satisfy a demanding potential customer and could not have done so without the spreadsheet.

17. POWERCO: PRICING POWER CONTRACTS

PowerCo is a large provider of electric power to municipalities and utilities in competitive electricity markets. The sales force generates about 20 proposed contracts a day, and PowerCo's pricing analysis group (PAG) must quickly and accurately price each proposal. If the proposal is

successful PowerCo will be bound by the quoted prices for years to come, so accurate and rapid pricing is essential to their business.

The PAG relies on a "pricing model" spreadsheet to conduct the pricing analysis and determine its recommendations. The inputs to the pricing model include historical data about the customer's energy consumption, prospective data about future consumption, and information about the competitive landscape. After preparing a structured set of inputs that represents the prospective customer's demand, the pricing model spreadsheet calls a commercial application that estimates future generation and transmission costs, which in turn are factored into the final pricing calculations in the spreadsheet. Essentially all of PowerCo's own proprietary knowledge about how to understand customer demand history and translate this into detailed power demand forecasts is captured in the pricing model spreadsheet.

Each pricing model is developed from a spreadsheet template. The template is "constantly evolving" as PowerCo updates the model and its logic. The PAG leader we interviewed is clear about the importance of the pricing model spreadsheet: "We are in a really competitive low-margin business, so the pricing models are critical. If your quoted price is off by $1/megawatt-hour for the wrong customer, you are looking at a financial disaster."

18. GROUPHOTEL: HOTEL REVENUE MANAGEMENT

GroupHotel is a large business hotel in a major North American city that derives the bulk of its revenue and profit from group bookings associated with conventions and hosted conferences. The success of GroupHotel is measured by a combination of room occupancy rates and profitability per occupied room.

GroupHotel has a planning and analysis group (PAG) that analyzes historical occupancy and profitability, studies trends associated with future group bookings, recommends price and margin guidelines, and answers ad-hoc queries from the sales and marketing groups. This group guides critical revenue management decisions, such as determining room inventory allocations, prices, and discounts to maximize profitability. They make and regularly update statistical forecasts of uncertain demand, and perform complex modeling to make optimal pricing decisions and predict room utilization.

PAG's most important decision support tool is the Long Range Planning spreadsheet. PAG's director created this spreadsheet. PAG uses it to understand room availability and demand for several months into the future, and to perform optimization to determine the size, pricing, and margins of groups the hotel should seek to book for each week of the analysis period. An analyst regularly downloads bookings data from GroupHotel's reservation system, and historical data from a mainframe archive, then runs an optimization macro to update availability and price recommendations.

The Long Range Planning spreadsheet is tightly integrated with critical business processes. Sales managers, marketing directors, financial analysts, and operations managers routinely use the Long Range Planning spreadsheet to do their jobs. Sales and marketing activities for GroupHotel are driven by the spreadsheet outputs. Finance uses it to "predict profitability for the next quarter." Housekeeping managers use it to "figure out how to staff the next month." According to the PAG team member that was interviewed for this research, "A sustained major problem with this spreadsheet would cause a huge amount of disruption at this property."

V. CONCLUSIONS

We presented 18 mission-critical spreadsheets and organized them into six categories: application software development, financial risk management, executive information systems, sales and marketing business processes, business operations, and complex analytics.

They are in organizations large and small; corporations, private firms, and family business; in start-ups and long-established firms; in a wide variety of industries including electric power, software development, hospitality, management consulting, supply chain management, investment banking, financial services, heavy industry, communications services, engineering, and disability services.

It was very easy to find these spreadsheets and their users. We believe this is the tip of an extremely large iceberg. Spreadsheets are vitally important to business, and merit sustained attention from academic researchers to discover techniques to enhance quality, productivity, and maintainability.

In Part 2 we develop insights regarding spreadsheets' usage, creation, and role in information systems; provide analysis and explanations; present new models to explain how spreadsheets are created and evolve; and provide a detailed agenda for future research.

PART 2: INSIGHTS, ANALYSIS & RESEARCH AGENDA

I. INTRODUCTION

In Part 1, we presented 18 cases of mission-critical spreadsheets and organized them into six categories. Here in Part 2, we analyze the cases to draw attention to important aspects of spreadsheets' usage, creation, and role in information systems. In Section II, we present major findings. In Section III, we provide analysis and hypotheses to understand the findings from Section II, present new models to explain how spreadsheets are created and evolve, and provide a detailed agenda for future research.

We present a multidisciplinary analysis integrating theoretical and empirical insights from the disciplines of information systems, computer science, spreadsheet risks, management science, and ethnology; and an extensive research agenda that we hope will inspire and guide future researchers. Highlights include a new skill-user programming paradigm which generalizes the traditional end-user programming paradigm; and the discovery of accidental legacy systems which are spreadsheets inherited by a person when they take a new job.

Our major findings in Section II are: 1) Spreadsheets are widely used for mission-critical functions; 2) Spreadsheets are an effective application development platform; 3) There is diversity of development skill in creators of mission-critical spreadsheets; 4) Sophisticated programmers sometimes choose spreadsheets over other languages; 5) Spreadsheets are amenable to formal development practices, but such practices seem rare; 6) Spreadsheets play a central role in the evolution of business processes and work systems; and 7) Spreadsheets are a source of "accidental legacy systems."

Our analysis and research agenda in Section III is organized in 10 themes. We discuss the prevalence of spreadsheets in mission-critical information systems. We explore the absence of "intentional spreadsheet applications" in the spreadsheet error literature. We discuss the features of spreadsheet programming languages for application development, and argue that the spreadsheet is a rapid development language, an integrated development environment, and a fourth generation language but raise concerns regarding source code protection.

We explain the factors that can militate in favor of building a custom spreadsheet rather than purchasing a commercial application. We present the options available for performing analytical work and explain why people can prefer a spreadsheet to a custom application by the IT department. We explore the practical managerial advantages of the spreadsheet. We discuss the absence in our data of the information technology department for spreadsheet applications. We frame many of the challenges of spreadsheets in terms of amateur programmers. We present four different perspectives on spreadsheet development. We wrap up our analysis with the skill-user programming paradigm.

This is exploratory research in a new area, intended to showcase the breadth and importance of spreadsheet usage. We have sufficient data to propose hypotheses and theories, but larger-scale work would be required to test those theories. Because of the small sample size, it is not possible to draw demographic conclusions from this data. In Section III we suggest a number of research questions that might be suitable for research aimed at generating findings that can be generalized to a population.

II. FINDINGS FROM INTERVIEW DATA

In this section we present seven key findings from the interviews in Part 1.

To assist the reader in finding company descriptions in Part 1, we provide the company's case study number along with their pseudonym. For example, AnalyCo/2 means the company is AnalyCo, and they are the case study number 2.

1. SPREADSHEETS ARE WIDELY USED FOR MISSION-CRITICAL FUNCTIONS

We presented 18 examples of mission-critical spreadsheets. Each spreadsheet is indispensable to an important aspect of the organization's activities. Without their respective spreadsheets, the people we interviewed would struggle to do their jobs and in some cases would be unable to sustain their companies.

These spreadsheets exhibit great diversity. They are in organizations large and small; public corporations, private firms, a family-owned business, and a non-profit; in start-ups and long-established firms; in industries including electric power, software development, hospitality, management consulting, supply chain management, investment banking, financial services, heavy industry, communications services, engineering, and personal services.

2. SPREADSHEETS ARE AN EFFECTIVE APPLICATION DEVELOPMENT PLATFORM

We saw three examples of spreadsheets being used by professional software developers to create application software for delivery to end users. SmallSoft/1 developed a large spreadsheet application software package that they sell commercially. AnalyCo/2 developed a spreadsheet application with thousands of users. ConsultCo/3 developed a decision support system delivered to a client. All three companies made a thoughtful, informed choice to use a spreadsheet to develop these applications.

3. DIVERSITY OF DEVELOPMENT SKILL IN CREATORS OF MISSION-CRITICAL SPREADSHEETS

Mission-critical spreadsheets are created by a broad spectrum of programmers, ranging from people barely able to modify a spreadsheet to professional computer programmers.

We judged six of our 18 informants to be "unsophisticated programmers," with little knowledge or experience of computer programming. One of these six (CleanCo/14) struggled even to modify an existing spreadsheet. The sole access to computer programming of the six unsophisticated programmers is the spreadsheet. They have no choice.

We judged 12 of our 18 informants to be "sophisticated programmers," defined as possessing the computer programming knowledge and experience to make an informed choice among the spreadsheet, a database, or a traditional language to create the functionality they required. Four of them were professional computer programmers (SmallSoft/1, AnalyCo/2, ConsultCo/3, TechStart/16).

4. SOPHISTICATED PROGRAMMERS SOMETIMES CHOOSE SPREADSHEETS OVER OTHER LANGUAGES

The 12 sophisticated programmers possessed the technical know-how to make an informed choice to use a spreadsheet or something else and chose to use a spreadsheet.

Choosy Programmers Choose Spreadsheets

We asked our informants whether they evaluated alternatives to the use of the spreadsheet. Of the 12 sophisticated programmers, eight answered "no," two "yes but not seriously," and two "yes." The 10 who responded "no" or "yes, but not seriously" were able to determine—with minimal analysis—that the spreadsheet was the right programming environment for their task. The two who answered "yes" performed a careful analysis of alternatives and then chose the spreadsheet.

These sophisticated programmers are not choosing spreadsheets out of ignorance or incompetence. They are choosing spreadsheets for development because spreadsheets are the right programming language for their circumstances.

5. SPREADSHEETS ARE AMENABLE TO FORMAL DEVELOPMENT PRACTICES, BUT SUCH PRACTICES SEEM RARE

Spreadsheet development practices vary widely. Surprisingly, the quality of development practices is independent of the sophistication of the programmer.

Formal Development Approaches

Two of the spreadsheets (SmallSoft/1, AnalyCo/2) were created using formal application development approaches by sophisticated users. The development process included user feedback, testing, documentation, and formal deployment to multiple users at separate locations. It is clear that the spreadsheet computer programming languages are amenable to formal development methodologies.

One spreadsheet (CleanCo/14) was acquired externally and then modified using a formal process by an unsophisticated user.

Informal Development Approaches

Fifteen of the spreadsheets were created informally, 10 by sophisticated programmers and five by unsophisticated programmers. The programmers gave little thought to the *process* by which they were creating the spreadsheet. They did not follow a development methodology. They performed little or no planning. They did not have a requirements document. They did not have a clear idea of what they were going to create when they started.

Spreadsheet Development Resources Are Misaligned

One would hope that mission-critical spreadsheets would be developed using formal methods to insure accuracy and maintainability. This does not seem to be the case. In our judgment, only two spreadsheets received an appropriate level of development resources. One received excessive resources, and 14 received inadequate resources. For one we could make no determination.

The spreadsheets created by SmallSoft/1 and AnalyCo/2 are of high importance and complexity, and both were created with great rigor and expense, as was appropriate. The CleanCo/14 spreadsheet, which is maintained by a highly diligent individual with little confidence in his spreadsheet skills, is carefully and formally checked on a frequent basis; the level of effort devoted to this spreadsheet seems excessive. For the DevCo/9 spreadsheet, which was created by a since-disbanded development group, we could not make a determination. The remaining 14 spreadsheets were created informally, with what in our judgment is inadequate investment in planning and other quality assurance techniques.

6. SPREADSHEETS PLAY A CENTRAL ROLE IN THE EVOLUTION OF BUSINESS PROCESSES AND WORK SYSTEMS

We found close identification between work activities and the spreadsheets that enables those activities. Our informants sometimes use a spreadsheet to define their responsibilities. For example, the analyst at I-Bank/5 told us that "My job is to get the spreadsheet together to support the deals."

We frequently observed that the development of the spreadsheet is very tightly interwoven with the development of the work system. The results provided by an early version of the spreadsheet support and motivate changes to business processes, which in turn motivate changes to the spreadsheet. Here are two examples.

At ShrinkSoft/8, an internal consultant was directed to develop a tool to track customer attitudes. Development commenced with only a "very high level" sense of what it would do. Development was guided by an "iterative discussion with decision makers"; the spreadsheet enabled new business processes, which motivated recognition of further business process opportunities and therefore further spreadsheet development.

SmallSoft/1 developed their spreadsheet application through a series of consulting engagements. The spreadsheet facilitated each consulting engagement, and was in turn modified to meet the additional needs that emerged during that engagement. At each round of changes, SmallSoft/1 acquired additional business process capability and knowledge as well as a more advanced spreadsheet; the spreadsheet and the business processes were very tightly connected.

7. SPREADSHEETS ARE A SOURCE OF "ACCIDENTAL LEGACY SYSTEMS"

We are observing what seems to be a new category of system, the "accidental legacy system." A legacy system "is an existing computer system or application program which continues to be used because the user (typically an organization) does not want to replace or redesign it" [Wikipedia 2007c].

An **accidental legacy system** is a computer program intended for personal use to accomplish a specific task that remains with the job position when the programmer departs and is used by the programmer's successor(s) in the job. Such a system is not an end-user system, because the user is not the programmer. Neither is it a traditional legacy system, because it is not an application intended for deployment to users.

Three of our spreadsheets (NonProfOrg/15, MajorCorp/6 and BigSoft/4) fall into the category of accidental legacy system. In each, the current user acquired the spreadsheet by inheriting it from their predecessor when they took their current job. In addition, each new user modified the spreadsheet in response to new needs they encountered. We note that the accidental legacy system at BigSoft/4 caused a three million dollar financial loss.

III. ANALYSIS AND RESEARCH AGENDA

1. PREVALENCE OF SPREADSHEETS IN MISSION-CRITICAL INFORMATION SYSTEMS

We showed that spreadsheets are used in a variety of ways in a variety of industries for mission-critical purposes. An important research question is how frequently do mission-critical systems use spreadsheets as a central technology?

We hypothesize such usage is common for three reasons. First, it was easy to find mission-critical spreadsheets, and we are confident that we could find many more examples without difficulty. Second, many of our research informants mentioned other critical spreadsheets used by themselves or their immediate co-workers, or commented on the importance of spreadsheets throughout their companies. Third, informal conversations with executives, managers, business

students, and professional acquaintances consistently indicate that spreadsheets are used for mission-critical business activities.

2. THE SPREADSHEET ERROR LITERATURE HAS NOT ADDRESSED INTENTIONAL SPREADSHEET APPLICATIONS

It is widely perceived that spreadsheet errors are a problem [Panko 1998; 2005] although our knowledge is limited and could be improved [Powell, Baker, and Lawson 2006; Powell, Lawson, and Baker 2007].

Grossman [2007] suggests a taxonomy of three classes of spreadsheet applications, "intentional"—a spreadsheet application that is purposefully developed and deployed to users, "de facto"—a spreadsheet constructed for personal use that is distributed to other people, and "accidental legacy" as defined in this paper.

Error Literature Does Not Consider Intentional Spreadsheet Applications

To date, it appears that the spreadsheet error literature does not consider intentional spreadsheet applications. It is difficult to know with certainty because the empirical literature frequently contains only little detail on the spreadsheets being examined [Powell, Baker, and Lawson 2006]. The error research considers end-user models, models built by business students in a laboratory, and perhaps some de facto applications or accidental legacy applications. Of the 89 examples in the EuSpRIG error corpus [EuSpRIG 2007], only one ("76, Auditor's report: a checklist of testing and maintenance standards") appears to be an intentional application.

An Error-Free Intentional Spreadsheet Application

Our informant at SmallSoft/1 tells us that he believes that their 73MB intentional spreadsheet application is error free. When they ported the spreadsheet to a procedural language, they wrote a software specification based on the spreadsheet, and had programmers code the specification in C++. They then compared the C++ outputs to the spreadsheet outputs, and found that the differences were all from errors in the procedural code. This is only one example, but it suggests that an accurate application can be written a spreadsheet programming language.

Future Research

It is widely accepted that errors are a problem in software, so we should not be surprised that spreadsheet errors are a problem. Unfortunately, there has been no research that compares spreadsheet errors to errors in other languages. It is not possible to gauge the relative risk of choosing to develop in a spreadsheet. We need research on this.

We believe that future empirical research on spreadsheet errors should carefully describe the origin of the spreadsheets being studied. For spreadsheet applications, researchers should distinguish among intentional spreadsheet applications, de facto spreadsheet applications, and accidental legacy spreadsheet applications.

We hypothesize that the categories of de facto spreadsheet application and spreadsheet accidental legacy system are more risky than intentional spreadsheet applications, because the former were not developed with the intention of use by others.

3. SPREADSHEETS FOR APPLICATION SOFTWARE DEVELOPMENT

Spreadsheets are an effective application development platform. This finding may come as a surprise, but as our case studies demonstrate, the spreadsheet is a rapid development language, functions as an integrated development environment, and is a fourth-generation language. However, the quality of source code protection is a concern, and it is not clear if there are inherent limits on size.

The Spreadsheet is a Rapid Development Language

McConnell [1996] places spreadsheets in the category of "rapid-development languages" (RDLs) that "offer speedier implementation than do traditional third-generation languages." He indicates that spreadsheets require on average only six statements to implement a function point that would require 30 statements in Visual Basic, 50 statements in C++, or 110 statements in FORTRAN. Or more simply, "Spreadsheets boost productivity through the roof."

Our informant at AnalyCo/2, an experienced software professional, corroborated McConnell's conclusions. He told us that when developing the spreadsheet it was faster for him to write the spreadsheet code than to write the requirements document necessary to commence development in a traditional language.

According to McConnell, RDLs include fourth-generation languages such as Focus; database management systems; visual programming languages such as Visual Basic; and "domain-specific tools" such as spreadsheets and statistical packages. Because RDLs lack first-rate performance and are limited to specific kinds of problems, they are better suited to in-house business software and limited-distribution custom software than to shrink-wrap or systems software. Our three examples of spreadsheet application software are consistent with McConnell, including in-house business software (AnalyCo/2) and limited-distribution custom software (SmallSoft/1 and ConsultCo/3).

The Spreadsheet Is an Integrated Development Environment

The spreadsheet can be considered an integrated development environment (IDE), which is a software tool that helps programmers create applications by simplifying or automating programming tasks. Ragsdale, Power, and Bergey [2002] argue that the spreadsheet is "highly functional, extendible, easy to integrate, and scalable," and recommend it as a DSS generation language for education.

An IDE "normally consists of a source code editor, a compiler and/or interpreter, build-automation tools, and (usually) a debugger" [Wikipedia 2007b]. The spreadsheet fits this definition. It provides a seamlessly integrated source code editor and interpreter. Build-automation is generally not an issue since compilation, object libraries, and object linking do not apply to spreadsheets. However, the Excel link editing feature does provide the only build-automation feature that a programmer would seem to need.

Spreadsheets make all calculations visible at all times. This powerful feature of the spreadsheet computer programming language is typically not available in other languages, and eliminates much of the need for a debugger. Excel provides powerful debugging tools such as Trace Precedents/Dependents, Display Formulas, R1C1 cell referencing, as well as traditional debugging tools such as Watch Window and Evaluate Formula.

In addition, the spreadsheet itself provides a simple graphical user interface (GUI), and Excel's Forms toolbar provides convenient tools for building more advanced GUI features.

The Spreadsheet Is a Fourth-Generation Language

There is little agreement about what exactly is a fourth-generation language (4GL). Definitions tend to be overly general, inductive (a series of examples), or vague. The definition in [Wikipedia 2007a] is overly general ("a programming language or programming environment designed with a specific purpose in mind, such as the development of commercial business software"); under this definition both the spreadsheet and the third-generation Python language would qualify.

An early textbook definition in Laudon and Laudon [1991] defines a fourth-generation language to be "a programming language with easy-to-use features that can be employed directly by end users or less skilled programmers," adding that "such languages can develop computer applications more rapidly than conventional programming languages and promise dramatic

boosts in productivity." The spreadsheet clearly fits this definition, with its well-known usage by end users and less skilled programmers, and its rapid development capabilities discussed earlier.

The [FOLDOC 2007] definition starts with "An 'application specific' language, one with built-in knowledge of an application domain, in the way that SQL has built-in knowledge of the relational database domain. The term was invented by Jim Martin to refer to non-procedural high level languages built around database systems. Fourth-generation languages are close to natural language and were built with the concept that certain applications could be generalized by adding limited programming ability to them." Spreadsheets can be considered specific to business and financial applications, as discussed under rapid development languages earlier. They are certainly nonprocedural, and their ease of use by untrained people suggests they have natural language features.

Another definition [PC Magazine 2007] indicates that, "Any computer language with English-like commands that does not require traditional input-process-output logic falls into this category. Many fourth-generation language functions are also built into graphical interfaces and activated by clicking and dragging. The commands are embedded into menus and buttons that are selected in an appropriate sequence." The spreadsheet clearly fits this definition of 4GL.

A useful definition [Hutchinson 2007] is a "type of programming language designed for the rapid programming of applications but often lacking the ability to control the individual parts of the computer. Such a language typically provides easy ways of designing screens and reports, and of using databases." Under this definition, the spreadsheet clearly qualifies.

The spreadsheet has the previously discussed benefits of being a rapid-development language and an integrated development environment. It has a built-in GUI (the grid) that can be very easily extended and customized. It has excellent report-generation features in the form of the grid; in fact, the spreadsheet grid functions effortlessly as the report generator for financial analyses. Excel's Pivot Table feature enables drag-and-drop programming of complex multi-dimensional analysis and reporting that would take hundreds if not thousands of lines of FORTRAN, with extraordinary flexibility and speed; likewise the AutoFilter tool. Importing data is highly automated, with no coding required. The spreadsheet has a higher level of abstraction than do third-generation languages, with automatic definition of variables by grid location, automatic variable typing, and no concept of registers or storage locations; indeed the underlying hardware is invisible.

The Issue of Source Code Protection

Grossman [2007] indicates that a weakness of spreadsheets for application development is poor source code protection. Users of spreadsheet application software are often given access to the source code of the spreadsheet which allows them to modify the programming statements of the application while they are using it. This seems unique to spreadsheets, and compromises the integrity of the application. However, he indicates that the dominant Excel spreadsheet has "pretty good protection" of source code provided that the programmer has the skill and commitment to properly use the available protection features. He argues that software-as-a-service spreadsheet languages such as Google Spreadsheet have potential for excellent source code protection should the creators of the spreadsheet language choose to provide it.

Are there Limits on Spreadsheet Size or Spreadsheet System Size?

Excel 2003 has a practical limit to size. Although it permits 65,536 rows and 256 columns, some spreadsheet developers tell us that there are problems with Excel stability at 60-70 MB file size. This is consistent with the experience of SmallSoft/1 who encountered problems at 73 MB.

Excel 2007 allows 1,048,576 rows by 16,384 columns. Any practical limitations of size in Excel 2007 are likely to controlled by hardware memory and the quality of the implementation of the spreadsheet executable. It will be interesting to see how large spreadsheets become in Excel 2007. There does not seem to be a conceptual limit to the size of a spreadsheet.

Spreadsheets can be combined into larger systems using a technique called “linking” where the programmer writes a cell formula that references a cell in a different spreadsheet. We did not examine linked spreadsheets, except for the three-spreadsheet system used by SmallSoft/1. We are hearing of systems with hundreds of spreadsheets linked together into large systems. The emerging spreadsheet management software industry will soon be providing extensive data on such spreadsheet systems (fasten your safety belts!).

Research Questions

How common is the use of spreadsheets for application software? Under what circumstances are spreadsheets appropriate for application development? We note that the owners of two of the spreadsheet applications chose to port their successful spreadsheet applications to third-generation languages. SmallSoft/1 ported their application because the application was so successful that it became too large for Excel to handle; Excel would simply disappear during editing. This does seems like poor memory management in Excel, and we hypothesize there should not be any inherent limit to the size of problem a well-designed spreadsheet language can handle. ConsultCo/3 ported their application because of concern over limited client willingness to pay a six-figure price for a spreadsheet plus performance issues.

It may be that spreadsheets are appropriate—and perhaps even under-utilized—as a powerful rapid prototyping language, with the prototypes in some cases being suitable for deployment to end users. SmallSoft/1 told us that the task of porting a spreadsheet to a procedural language is difficult. We should know more about how to manage the conversion process from spreadsheet to third-generation language. It is interesting to speculate whether any applications have been ported from third-generation languages to spreadsheets!

It would be very helpful to see a thoughtful review of the spreadsheet as a computer programming language that examines the spreadsheet as one would examine any other computer programming languages, identifying strengths and weaknesses.

4. COMPARING CUSTOM SPREADSHEET DEVELOPMENT TO A COMMERCIAL APPLICATION

Why might someone choose to do custom spreadsheet development when a commercial application might suffice? We are aware of five reasons that programmers might sensibly choose a spreadsheet over a commercial application: no suitable commercial application, lack of awareness, business process changes, expense, and search cost.

A suitable commercial application does not exist. Commercial applications only exist when many users in many companies require similar functionality. When new functionality is required, commercial software simply will not exist. One might well ask “Why write a piece of software in C++?” as ask “Why write a spreadsheet?”

People do not know about a commercial application. Some of our informants work in areas where commercial software might have fit their needs, but they did not know about it. (We did not explore this possibility, and for none of our examples are we aware of a commercial application that would provide the essential functionality.) It is possible that spreadsheet users are expensively reinventing a tool that could easily be purchased.

Business process changes are too costly. Firms are often faced with a choice between adapting their business processes to one-size-fits-all commercial software, or creating custom software that incorporates their existing business processes. This dilemma is well known to companies that choose to implement ERP systems and are forced to undergo the painful experience of changing the way they operate in order to fit the requirements and limitations of the ERP. With ERP, the economics of custom development or retention of legacy software are often so unattractive that firms choose to invest in ERP software and implementation. For many other smaller applications, the choice of custom development can be reasonable and appropriate.

Commercial software has lifecycle expenses. Commercial software requires the purchase of software licenses, which can be very expensive. These licenses often require annual renewal, so the expense is ongoing. With spreadsheets, once the application is built there are no further costs. Commercial software may have a limited number of licenses that make sharing of models or results difficult or impossible. In contrast, spreadsheet software is universally available on personal computers, so licenses are not an issue and sharing is easy. Commercial software can be difficult to learn and require expensive training course and concomitant travel expense. The firm may be vulnerable to losing the only person who knows how use the application. In contrast, Ragsdale, Power, and Bergey [2002] indicate that the spreadsheet is "ubiquitous and easy to learn." With spreadsheets, there are many people (including new hires) who might be able to figure out what a spreadsheet does. The cost of doing this will be employee time rather than a cash outlay for a training course.

Search costs: Consideration of commercial applications entails time and risk. People must research commercial application alternatives with no guarantee that one can be found. If they can find one or more, they must investigate in detail what it can do and then select an application, have its purchase approved (which can prove time-consuming, frustrating, or impossible), acquire it (which can be costly and may require a commitment to annual payments), install it (more cost and risk), receive training, configure and customize it (more expense), and then finally commence useful work. This can be a daunting prospect compared to simply writing the code that is needed, with the primary cost being one's own time.

Research Questions

We believe it is very important to understand the reasons that people choose to use spreadsheets to solve their problems. For unsophisticated programmers it is simple: they have no choice. But why do sophisticated programmers sometimes choose to develop in spreadsheets? What are the characteristics of their problem that makes the spreadsheet an appropriate choice? We simply do not know, and we need research to understand why.

We presented four reasons that can militate in favor of doing development in a spreadsheet. Each deserves careful study and evaluation, particularly empirical investigation and testing.

We must not discount the possibility that programmers might be making a mistake. By choosing a spreadsheet, Chan and Storey [1996] note that managers can become so comfortable with spreadsheets that they are reluctant to adopt other software packages even if they are more suitable for a particular application. Does this happen, and if so how frequently? In our opinion, the burden of proof resides with those who believe that users are making a mistake, not with the people who choose to write spreadsheet programs.

It is possible that programmers are trading off short-term gain for long-term pain by using a spreadsheet. McConnell [1996] discusses the risks of rapid development languages (RDLs) including spreadsheets. He points out that RDLs are not suitable to some projects and do not scale up well to large projects. He argues that programmers tend to use sloppy design and coding practices with RDLs which can cause significant problems later. He argues this is the programmer's (poor) choice rather than an attribute of the RDL and that programmers using RDLs should manage this risk by employing careful design and coding standards. The existence of design and coding standards could be studied empirically, and there is much work to be done in developing, testing, and promulgating such standards (discussed in Section III.9).

5. SPREADSHEET AS A SMART COMPROMISE FOR PERFORMING ANALYTICAL WORK

How can a typical business person perform analytical work? There are essentially three alternatives.

The first "idealized" (some would say "gold-plated") alternative is to do formal development using the IS department. This can deliver high quality, but has well-known problems with cost, delivering what the user really needs [Tire Swing Cartoon 2007], inflexibility, and the deal-breaker

problem of being too slow to be helpful. However, as we discuss in Section III.7, the information technology department may be incapable, unacceptable, or nonexistent.

The second "do almost nothing" alternative is to do only pencil and paper analysis with the support of a calculator, which has obvious gross limitations and can incur substantial opportunity cost from inadequate analysis.

The third "do it yourself" alternative is to develop an analytical tool in a spreadsheet and obtain the benefits of the analysis that the user genuinely needs, when he needs it, but incur the risks of errors.

Each of the three alternatives provides a different portfolio of cost, flexibility, speed, and risk. Practical-minded business people might well choose the third alternative—do their own development in a spreadsheet—as compromise between the extremes of pencil-and-paper and working with the IT department.

There is much research to be done to understand the forces and economics that encourage people to develop their own spreadsheet computer programs rather than do nothing, or wait for the IS department to do it (if they have an IS department). How could these people be assisted? These questions are likely to be answered by spreadsheet development methodologies which we discuss in section III.9.

6. PRACTICAL MANAGERIAL ADVANTAGES OF THE SPREADSHEET

A spreadsheet is one of thousands of computer programming languages. Why might a programmer select a spreadsheet? There are practical managerial considerations In a business organization that might not exist in a traditional programming organization. These include acceptance by decision makers, limited software licenses, training, flexibility, and modeling.

Acceptance by Decision Makers. Historian Campbell-Kelly and Aspray [1996, p. 191] discusses the rise of COBOL. He indicates that a benefit of English-language coding is a "comforting illusion for administrators and managers that they could read and understand the programs their employees were writing, even if they could not write them themselves. While many technical people could see no point in such 'syntactic sugar,' it had important cultural value in gaining the confidence of decision makers." Many of our informants indicated that the transparency of the spreadsheet was important to their organization, and we wonder whether Campbell-Kelly's argument holds for spreadsheets as it did for COBOL.

Flexibility: The Swiss Army Knife Argument. Powell and Baker [2004] argue that the "spreadsheet is the *second* best way to do many kinds of analysis and is therefore the *best* way to do most modeling . . . the spreadsheet is the Swiss Army knife of business analysis." Or as one of our informants puts it "Excel is pretty good just-enough-of-a-tool for a lot of purposes."

The availability of a "good enough" spreadsheet solution can eliminate the need to invest time and money in acquiring and learning a new language or database package. Smart managers know the Pareto Principle and will often choose "second best" over "best" unless there is good reason to invest in "best."

Modeling: Spreadsheets are valuable for discovering what is to be programmed. Grossman [2002] explains that spreadsheets are well suited for "exploratory modeling" of poorly-understood business issues, where the act of modeling the issue in a spreadsheet teaches the programmer about his problem. This can be thought of as the discovering the software specification by constructing the software.

This concept of learning what needs to be accomplished by programming is well known in computer science. Knuth [1973, p. 188] suggests that after completing a program "it is often a good idea to scrap everything and start over again . . . for we have learned a great deal about our

problem." DeGrace and Stahl [1990, p. 82] discuss the difficulty of programming a "wicked problem," which is a programming problem that is "fully understood only *after* it is solved.".

Research Questions

We propose several hypotheses for adopting spreadsheets for adopting spreadsheets for writing application software. There is a need a much broader empirical study of spreadsheet applications with a special focus on why spreadsheets are chosen as the underlying technology. It may be that spreadsheets are uniquely appropriate for certain classes of poorly defined, ambiguous, or fast-changing business situations. It may be that spreadsheet applications are more easily accepted by management than applications written in other languages. We note that there is rich and well-established literature on technology adoption and productivity (e.g. Katz and Shapiro 1986, David and Paul 1990) that may be useful in examining these questions.

7. THE ABSENCE OF THE INFORMATION TECHNOLOGY DEPARTMENT FOR SPREADSHEET APPLICATIONS

One interesting finding from our case studies was the lack of involvement of the Information Technology Departments (ITD) in the development of mission-critical spreadsheets. First, smaller organizations simply may not have an ITD, as was the case of DealCo/11, NonProfOrg/15, and CleanCo/14. Second, many of our respondents (including I-Bank/5, MajorCorp/6, ConsultCo/3, and CompEng/13) noted that their companies' ITDs are focused on hardware, software, networking, and/or telecommunications acquisition and support, and do not view spreadsheet application development as part of their domain.

While some ITDs potentially had the capability to develop spreadsheet applications, our respondents chose not to use them. One reason was a prohibitive internal cost (GroupHotel/18, ShrinkSoft/8) to using the ITD's services. Another reason was because the ITDs lacked domain knowledge and thus required a much more detailed specification than was available at the outset of the project (CommCorp/10, Archiva/12, AssetFund/7). In some cases the timeframe was too short to involve another part of the organization (PowerCo/17, ConsultCo/3).

We did find four instances where the ITD got involved with a mission-critical spreadsheet. Two instances were to port the spreadsheet to another platform after it had been developed and utilized (ShrinkSoft/8, ConsultCo/3). Two instances were to enhance the quality of the data flow into the spreadsheet but not to touch the spreadsheet itself (AnalyCo/2, AssetFund/7). The ITD did not get involved with the spreadsheet because the spreadsheet model and its outputs would continue to evolve over time, and therefore needed to remain in the flexible spreadsheet platform.

We hasten to point out that our survey methodology and sample size do not allow general conclusions to be drawn regarding the presence or role of the ITD. Are our informants simply unlucky in that they have no IT support or their organization has no information center [Guimaraes et al. 1999; Essex et al. 1998] to provide support, or are information centers rare? Do ITDs generally provide meaningful support to people who program mission-critical spreadsheets? We believe that this topic merits additional investigation.

8. THE ISSUE OF AMATEUR PROGRAMMERS

In a classic book from the 1970s, Weinberg [1998] makes an important distinction between amateur and professional programmers. He first describes how amateur programmers react to a programming challenge: "The amateur, being committed to the results of the particular program for his own purposes, is looking for a way to get the job done. If he runs into difficulty, all he wants is to surmount it—the manner of doing so is of little consequence."

A professional has a different reaction: "Not so, however, for the professional. He may well be aware of numerous ways of circumnavigating the problem at hand . . . But his work does not stop there; it begins there. It begins because he must *understand* why he did not understand, in order

that he may prepare himself for the programs he may someday write which will require that understanding."

Weinberg continues to define the key distinction between amateur and professional programmers: "The amateur, then, is learning about his *problem*, and any learning about programming he does may be a nice frill or may be a nasty impediment to him. The professional, conversely, is learning about his *profession*—programming—and the problem being programmed is only one incidental step in his process of development."

What is important for our purposes is that Weinberg is suggesting that the task of increasing one's *domain* knowledge is a separate activity from increasing one's *programming* knowledge. The act of writing software to solve a problem is neither a guarantee of learning about the problem, nor about writing software; *learning is dependent on where the programmer focuses his attention*. The amateur programmer (who is, one hopes, a business professional) is focused on learning about the business. The professional programmer is focused on learning about programming.

The question of amateur programmers is particularly important and could be an area for fruitful and interesting research. Alavi and Weiss [[1985] discussed "potential" risks from end-user computing. (These are actually risks from amateurs writing software, and we will speak of amateurs rather than end-users.) We find that some amateurs/end-users are building mission-critical programs despite being quite unsophisticated.

This raises two important questions. First, what are the real risks of amateur programming? What is the nature of these risks, and how serious are they? These questions can and should be explored empirically.

Second, what might be done for amateur spreadsheet programmers? Even if one believed that they should stop building spreadsheets or purchase a commercial application, we can be sure that amateur programmers will persist in writing spreadsheet software for reasons discussed in Sections III.4, III.5 and III.6. The important question is: How can amateur programmers be supported and made more effective at developing spreadsheet applications and mission-critical spreadsheets? The answer likely lies in the creation and promulgation of development methodologies.

9. PERSPECTIVES ON DEVELOPMENT APPROACHES

As discussed in Section II.5, we saw wide diversity in spreadsheet development practices. We present four different ways to think about these approaches. Some approaches which seem strange or even foolhardy from the perspective of traditional software development might be sensible when viewed from a work systems or organizational perspective.

Formal Spreadsheet Software Development with Requirements Document

AnalyCo/2 used an approach based around a set of written requirements devised by users that were implemented by professional programmers. The programmers built a prototype based on the specifications, sought feedback from users on quality, workflow, and enhancements. They revised the prototype, performed formal testing, and created detailed documentation before deployment to users.

This approach corresponds to the "staged delivery" lifecycle model of software engineering [McConnell 1996, ch. 36]. This indicates that professional programmers can build an application using a spreadsheet programming language the same way they build an application using a procedural programming language.

Formal Spreadsheet Software Development without Requirements Document

SmallSoft/1 built and delivered a series of spreadsheet applications over multiple consulting engagements with different clients. Each engagement taught them something about what the product needed to do and motivated changes to the software. Each round of changes to the software was carefully designed, and sometimes extensively recoded, using disciplined coding practices. Each consulting engagement delivered a working application to a client, tantamount to the release of a new version.

SmallSoft/1's development approach is closely analogous to the "evolutionary delivery" lifecycle model of software engineering [McConnell 1996, ch. 20]. This again indicates that professional programmers can build an application using a spreadsheet programming language the same way they build an application using a procedural programming language.

Spreadsheet Integration and Co-Evolution with Work Systems and Business Processes

We frequently observed that programmers changed spreadsheets, sometimes almost continually. This recalls Nardi's observation that spreadsheet programmers "discovered needs as they went along, and the interaction with the spreadsheet package directly supported this vital process" [Nardi 1993, p. 83]. The work systems model [Alter 2002] allows us to explain these observations, viewing the spreadsheet as the technology that links together business processes, people, and information to meet the demands of internal or external customers.

To the extent that there are changes over time in the business processes, or the programmer's understanding of the work system, it is desirable for the supporting technology—the spreadsheet—to evolve over time, and it might well be undesirable to incur the time lags associated with more formal development approaches.

Spreadsheet as Organizational Artifact

When a spreadsheet is inherited as an accidental legacy application, it can be thought of as an "organizational artifact" [Nardi and Miller 1990]. In the language of Activity Theory, such artifacts are critical to the activities (or work systems) with which they are associated; [Kuutti 1996, p. 26] observes that "artifacts themselves have been created and transformed during the development of the activity and carry with them a particular culture—a residue of that development." The DevCo/9 spreadsheet is a particularly strong exemplar of this.

In this sense, accidental legacy system spreadsheets represent important work system knowledge that may not otherwise be recorded. We wonder whether spreadsheet software (with its user-friendly rapid development and collaborative capability) might be uniquely suited to this important purpose.

Research Questions

The co-evolution of mission-critical spreadsheets with their associated work systems suggests a number of research questions. What are the best practices for managing spreadsheets that are developed organically within a work system to address an unmet need? How can organizations create conditions to foster efficient, successful co-evolution of spreadsheets and business processes? How can this co-evolution be connected to managerial decisions regarding development resources and risk of errors?

Mission-critical spreadsheets are a valuable organizational asset [Grossman 2002], but thoughtful stewardship of these assets seems rare. The appropriate level of resources to devote to the development and quality control of a mission-critical spreadsheet is a managerial decision, but it is not clear that this decision is receiving conscious attention: For the 14 spreadsheets that had inadequate resources (Section II.5), we saw no evidence of managerial interest. Do managers take inappropriate risks, due perhaps to ignorance, or a false belief that their employees will request appropriate resources? Or do managers make thoughtful risk tradeoffs by choosing not to

invest in careful development? How do the answers change when the manager is also the spreadsheet programmer?

Many errors presumably originate during the creation of a spreadsheet, but little attention has been paid to the prescriptive issue of how people should develop spreadsheets. This topic, called "spreadsheet engineering," requires extensive research. Grossman and Özlük [2004] provide a framework for analyzing and developing spreadsheet engineering methodologies, consisting of modeling, development parameters (planning), design, programming, quality control, debugging, documentation, usage, and modification. They argue that each element needs research attention, and that the question of how to deploy research results to spreadsheet programmers requires careful consideration. Based on our interviews with users of mission-critical spreadsheets, we believe that issues of development parameters, design, quality control, and documentation are particularly important. Using other terminology [Alter 1996, p. 366], modularity, compatibility, and reusability are especially important, particularly in light of accidental legacy spreadsheets (Section II.7). Spreadsheet programmer productivity has barely been considered, with Panko [2000a, 2000b] providing insight into productivity in the context of error research.

Appropriate development methodologies, particularly those likely to be adopted by unsophisticated programmers, are urgently needed. Because of the high diversity of purposes, organizations, and people who develop mission-critical spreadsheets, it is unlikely that we will see "one size fits all" recommendations for the creation, maintenance, and use of spreadsheet information systems. Any prescriptive research on spreadsheet development will need to be connected to specific development domains and deployed in such a way that spreadsheet programmers, who in our experience are very busy people, will be willing to adopt them. As discussed in [Grossman [2002], a prerequisite to methodology development is a taxonomy of spreadsheet information systems that can be used to carve up the spreadsheet universe into taxa containing sufficient commonality to admit strong, detailed prescriptive guidance.

The first spreadsheet development methodology paper we know of is Kumiega and Van Vliet [2006], in the domain of financial trading applications. They describe in detail a stage-gate approach where they construct a prototype in a spreadsheet, test the prototype to determine its value to financial traders, and then decide whether to kill it (no or minimal value), deploy it as a spreadsheet (low value), or port it to VBA or C++ (high value).

10. THE SKILL-USER PROGRAMMING PARADIGM

It is tempting to view spreadsheets as an instance of end-user programming, and use insights into spreadsheets to formulate larger hypotheses about end-users in general. Implicit in the classic end-user programming paradigm is an assumption that professional programmers develop software for other people, not for themselves, and that amateur programmers develop software only for themselves, not for others. We find this framework to be inconsistent with our empirical observations. We require a richer paradigm to explain what we are seeing.

Extending the End-User Programming Paradigm

We propose the "skill-user programming paradigm," summarized in Figure 1. It considers two dimensions, with "skill" describing whether the programmer is an amateur or a professional (discussed in Section III.8), and "user" indicating whether the person(s) who will use the software are the programmer or other people.

The lower left quadrant 1 is an amateur programmer developing software for himself; this is classic end-user programming. In our data, this includes spreadsheets by NonProfOrg/15, CleanCo/14, BigSoft/4, and PowerCo/17.

Quadrant 3 in the top right is a professional programmer developing software for other people; this is traditional application development. In our data, this includes spreadsheets by GroupHotel/18, SmallSoft/1, DevCo/9, AnalyCo/2, and ConsultCo/3.

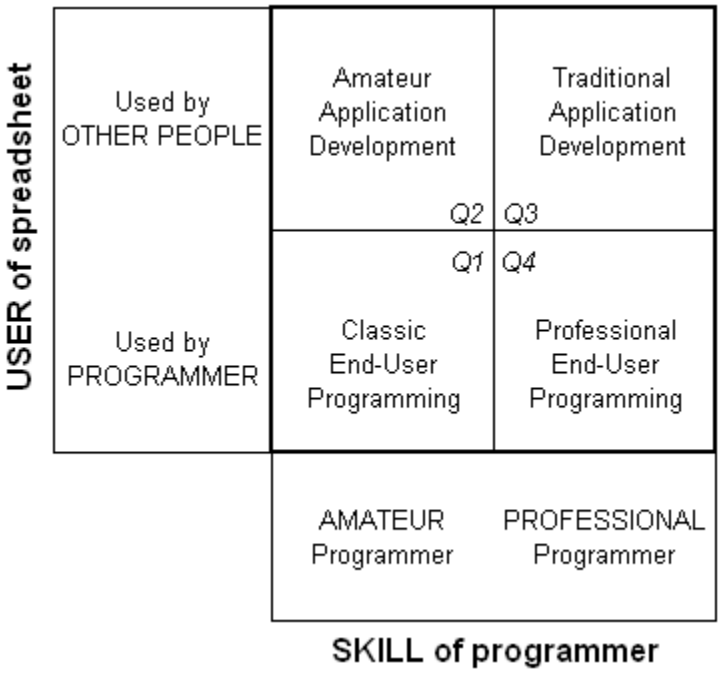

Figure 1. The Skill-User Programming Paradigm

Quadrants 1 and 3 represent the established end-user programming paradigm, with traditional application development (by professionals) on the one hand, and classic end-user programming (by amateurs) on the other.

Quadrant 2 on the top left corresponds to an amateur programmer writing software that is used by other people. Because it is used by other people, the software functions as an application. Therefore, we call this "amateur application development." In our data, this includes spreadsheets by Archiva/12, CommCorp/10, CompEng/13, DealCo/11, MajorCorp/6, I-Bank/5, ShrinkSoft/8, AssetFund/7. As discussed in Section III.2, these can be an accidental spreadsheet legacy system or a de facto application. An application intentionally written by an amateur programmer would also be in quadrant 2.

Quadrant 4 in the lower right corresponds to a professional programmer writing software for his own use; we call this "professional end-user programming." In our data, this includes the spreadsheet by TechStart/16.

Quadrants 2 and 4 represent aspects of programming that are not captured by the classic end-user paradigm. Quadrant 2 provides a "theory home" for the phenomena of accidental legacy spreadsheets and de facto spreadsheet applications. Quadrant 4 provides a "theory home" for personal development by professional programmers.

Evolution of a Spreadsheet in the Skill-User Programming Paradigm

In Figure 1, a progression from quadrant 1 to quadrant 2 represents a transition from classic end-user programming software to an amateur application. We are aware of two ways that this can occur. The first is an accidental legacy spreadsheet (Section II.7), where a spreadsheet is inherited with a job. The spreadsheet moves from Quadrant 1 to Quadrant 2 as a result of a new person rotating into a job position and using the spreadsheet. We saw this in three spreadsheets (Section II.7), and the EuSpRIG error corpus [EuSpRIG 2007] contains several examples. The second is a de facto application where a spreadsheet intended for personal use (quadrant 1) moves to quadrant 2 by means of its distribution to other people.

Research Questions

We believe it is essential to decouple the skill of the programmer from the person who uses the software: not all end users are amateur programmers, and not all applications are written by professional programmers. Much work remains to fully validate and explore this model. Although we developed this model based our experience with mission-critical spreadsheets, we believe it may be applicable to other software such as the Microsoft Access database. To evaluate this possibility we need to know the prevalence of different computer programming languages used by amateur programmers. In our experience spreadsheets are everywhere, databases are not uncommon, special purpose software (for example, decision tree software) is rare, and procedural programming languages are unknown. However, we need data on this.

Limiting the discussion to spreadsheets, how common are they in each of the four quadrants? How do spreadsheets in Quadrant 2 (amateur applications) arise? Is it solely inadvertent mechanisms (accidental legacy spreadsheets and *de facto* spreadsheet applications) or do amateurs intentionally create application software in spreadsheets? How do the recipients of accidental legacy spreadsheets decide whether to adopt the spreadsheet, create their own spreadsheet, or initiate a traditional application, and how does the work system knowledge embedded in the spreadsheet affect their decision? How do their managers think about this, and do they even recognize it as a decision?

IV. CONCLUSIONS

We analyzed 18 mission-critical spreadsheets used for application software development, financial risk management, executive information systems, sales and marketing business processes, business operations, and complex analytics. They are in organizations large and small; corporations, private firms, and family business; in start-ups and long-established firms; in a wide variety of industries including electric power, software development, hospitality, management consulting, supply chain management, investment banking, financial services, heavy industry, communications services, engineering, and disability services.

We find that spreadsheets are widely used for mission-critical functions. Spreadsheets are used for application software development by professional programmers who could choose to use other programming languages. We argue that the spreadsheet is a powerful fourth-generation computer programming language suitable for rapid development, and functions as an integrated development environment.

We find there are important misalignments between spreadsheet importance and resources devoted to development. Professional programmers can use a spreadsheet like any other computer programming language, but many spreadsheet programmers are amateurs. We found both a large spreadsheet that is believed to be error-free, and a spreadsheet with an error that cost millions of dollars.

The state of research knowledge regarding spreadsheets stands in marked contrast to research knowledge regarding “traditional” procedural languages and object-oriented programming environments. Information systems and computer science researchers have intensively studied traditional programming, and have developed vast theoretical and empirical knowledge about programming practices, software engineering, productivity, quality assurance, and many other issues. However, similar work in spreadsheets remains to be done, and such research has the potential to impact the practice of millions of spreadsheet programmers.

We believe that the way that people create and use spreadsheets merits broad, sustained empirical and theoretical research attention. This research is inherently inter-disciplinary, with relevant work and tools in information systems, computer science, management science, spreadsheet risks, and ethnology. We propose a rich array of questions that we hope will stimulate future research.

REFERENCES

Editor's Note: The following reference list contains hyperlinks to World Wide Web pages. Readers who have the ability to access the Web directly from their word processor or are reading the paper on the Web, can gain direct access to these linked references. Readers are warned, however, that
1. these links existed as of the date of publication but are not guaranteed to be working thereafter.
2. the contents of Web pages may change over time. Where version information is provided in the References, different versions may not contain the information or the conclusions referenced.
3. the author(s) of the Web pages, not AIS, is (are) responsible for the accuracy of their content.
4. the author(s) of this article, not AIS, is (are) responsible for the accuracy of the URL and version information.

Alavi, M. and I. R. Weiss. (1985). "Managing the Risks Associated with End-User Computing," *Journal of MIS* 2(3), pp. 5-20.

Alter, S. (1996). *Information Systems: A Management Perspective*, Menlo Park: Benjamin/Cummings Publishing.

Alter, S. (2002). "The Work System Method for Understanding Information Systems and Information System Research," *Communications of the Association for Information Systems* (9)6.

Bach, P. (2007). "Awareness and Market Changes in the Control of End User Computing Risk Management," *European Spreadsheet Risks Interest Group 8th Annual Symposium*, London, UK, July 2007, verbal comment.

Campbell-Kelly, M. and W. Aspray. (1996). *Computer: A History of the Information Machine*, New York, NY: BasicBooks.

Chan, Y. E. and V. C. Storey. (1996). "The Use of Spreadsheets in Organizations: Determinants And Consequences," *Information & Management* Vol. 31 (3) pp. 119-134.

Croll, G. (2005). "The Importance and Criticality of Spreadsheets in the City of London," *European Spreadsheet Risks Interest Group 6th Symposium Proceedings*, London, pp. 83-93. http://sprig.section.informs.org/sprigfiles/Croll05.pdf

David, P. A. and A. Paul. (1990). "The Dynamo and the Computer: An Historical Perspective on the Modern Productivity Paradox," *American Economic Review* 80(2), Papers and Proceedings of the Hundred and Second Annual Meeting of the American Economic Association (May, 1990), pp. 355-361.

DegGrace, P. and L. H. Stahl. (1990). *Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms*, Englewood Cliffs, NJ: Prentice-Hall.

Essex, P., A. Essex, S. R. Magal, and D. E. Masteller. (1998). "Determinants of Information Center Success," *Journal of Management Information Systems* Volume 15 , Issue 2 pp. 95 - 117

EuSpRIG. (2007). "Spreadsheet Mistakes—News Stories," http://www.eusprig.org/stories.htm (current June 25 2007).

FOLDOC. (2007). "Fourth Generation Language,"" „"*The Free On-Line Dictionary of Computing,"* http://foldoc.org/index.cgi?query=fourth+generation+language (current June 25, 2007) .

Gorgone, J. T, J. S. Valacich, D. L. Feinstein, G. B. Davis, H. Topi, H. E. Longenecker Jr. (2003). "IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems," *Communications of the Association for Information Systems* (11)1. http://cais.aisnet.org/articles/111/default.asp?View=Journal&x=93&y=6

Grossman, T. (2002). "Spreadsheet Engineering: A Research Framework," *European Spreadsheet Risks Interest Group 3rd Annual Symposium*, Cardiff, July 2002.

Grossman, T. A. (2007). "Source Code Protection for Applications Written in Microsoft Excel and Google Spreadsheet," *European Spreadsheet Risks Interest Group 8th Annual Symposium*, London, UK, July 2007.

Grossman, T. and Ö. Özlük. (2004). "A Paradigm for Spreadsheet Engineering Methodologies," *European Spreadsheet Risks Interest Group 5th Annual Symposium*, Klagenfurt, Austria.

Guimaraes, T., Y. P. Gupta and R. R. Rainer Jr. (1999). "Empirically Testing the Relationship between End-User Computing Problems and Information Center Success Factors," *Decision Sciences*, 30 (2) pp.393-413.

Hutchinson. (2007). "Fourth Generation Language," *The Hutchinson Dictionary of Computers, Multimedia and the Internet*, http://www.tiscali.co.uk/reference/dictionaries/computers/data/m0005805.html (current June 25, 2007).

Katz, M. L. and C. Shapiro. (1986). "Technology Adoption in the Presence of Network Externalities," *Journal of Political Economy* 94(4), pp. 822-841.

Knuth, D. E. (1973). *The Art of Computer Programming, Volume 1: Fundamental Algorithms, 2nd Edition*, Reading, MA: Addison-Wesley.

Kumiega, A. and B. Van Vliet. (2006). "A Software Development Methodology for Research and Prototyping in Financial Markets," *European Spreadsheet Risks Interest Group 7th Symposium Proceedings*, London, pp. 107-127.

Kuutti, K. (1995). "A Framework for Human Computer Interface Research," in B. Nardi (ed) *Context and Consciousness*, Cambridge, MA: MIT Press pp. 2-44

Laudon, K. C. and J. P. Laudon. (1991). *Management Information Systems: A Contemporary Perspective, 2nd Edition*, New York: Macmillan, p. 901.

Laudon, K. C. and J. P. Laudon. (2006). *Management Information Systems: Managing the Digital Firm*, 9th Edition, Upper Saddle River, NJ: Prentice Hall, p. G1.

Liebowitz, S. (2002). *Re-thinking the Network Economy: The True Forces That Drive the Digital Marketplace*., New York: Amacom Books.

McConnell, S. (1996). *Rapid Development*, Redmond, WA: Microsoft Press.

Nardi, B. (1993). *A Small Matter of Programming: Perspectives on End User Computing*, Cambridge, MA: MIT Press.

Nardi B. and J. Miller. (1990). "An Ethnographic Study of Distributed Problem Solving in Spreadsheet Development," *Computer Supported Cooperative Work Proceedings: 197-208*. ISBN 0-89791-402-3.

Panko, R. (1998). "What We Know About Spreadsheet Errors," *Journal of End User Computing* 10(2), pp. 15-21

Panko, R. (2000a). "Errors in Spreadsheet Auditing Experiments," http://panko.shidler.hawaii.edu/ssr/auditexp.htm, (current July 25, 2007)

Panko, R. (2000b). "Errors during Spreadsheet Development Experiments," http://panko.shidler.hawaii.edu/ssr/devexpt.htm (current July 25, 2007)

Panko, R. (2005). "What We Know about Spreadsheet Errors," http://panko.shidler.hawaii.edu/ssr/Mypapers/whatknow.htm (current July 25, 2007)

Panko, R. (2006). "Spreadsheets and Sarbanes-Oxley: Regulations, Risks, and Control Frameworks," *Communications of the Association for Information Systems* 17(29). http://cais.aisnet.org/articles/default.asp?vol=17&art=29

PC Magazine. (2007). "Fourth-Generation Language," *PC Magazine Encyclopedia*, http://www.pcmag.com/encyclopedia/ (current June 25, 2007).

Powell S. and K. Baker. (2004). *The Art of Modeling with Spreadsheets*, John Wiley & Sons.

Powell, S., K. Baker, and B. Lawson. (2006). "A Critical Review of the Literature on Spreadsheet Errors," Tuck School of Business Spreadsheet Engineering Research Project Working Paper.

Powell, S., B. Lawson,, and K. Baker. (2007). "Impact of Errors in Operational Spreadsheets," Tuck School of Business Spreadsheet Engineering Research Project Working Paper.

Ragsdale, C. T., D. J. Power, and P. K. Bergey. (2002). "AMCIS 2002 Panels and Workshops Ii: Spreadsheet-Based DSS Curriculum Issues," *Communications of the Association for Information Systems* 9(21).

Reichheld, F. F. (2003). "The One Number You Need to Grow," *Harvard Business Review* (81)12.

Senn. J. A. (2004). *Information Technology: Principles, Practices, Opportunities (3rd Edition)*. Upper Saddle River, NJ: Pearson/Prentice Hall.

Tire Swing Cartoon. (2007). http://www.uoregon.edu/~ftepfer/SchlFacilities/TireSwingTable.html (current July 25, 2007).

Weinberg, G. (1998). *The Psychology of Computer Programming*, Silver Anniversary Edition, Dorset House.

Wikipedia. (2005). "Executive Information System," http://en.wikipedia.org/wiki/Executive_information_system (current October 27, 2005).

Wikipedia. (2007a). "Fourth-Generation Programming Language," http://en.wikipedia.org/wiki/Fourth-generation_programming_language (current June 25, 2007).

Wikipedia. (2007b). "Integrated Development Environment," http://en.wikipedia.org/wiki/Integrated_development_environment (current June 22, 2007).

Wikipedia. (2007c). "Legacy System," http://en.wikipedia.org/wiki/Legacy_system (current June 25, 2007).

ABOUT THE AUTHORS

Thomas A. Grossman is Associate Professor of Finance and Quantitative Analytics at the University of San Francisco. He studies how people use spreadsheets and analytics to run their companies. He is President of the Spreadsheet Productivity Research Interest Group, and on the Board of Advisers of Compassoft Inc. He received his PhD in Management Science and Engineering from Stanford. He spent many years at the Haskayne School of Business at the University of Calgary, and was a visiting faculty member at the Tuck School of Business at Dartmouth. He used to make his living in a consulting firm building large-scale optimization

systems for the airline industry. In his spare time he serves as President of a volunteer mountain rescue team.

Vijay Mehrotra is an Assistant Professor in the Decision Sciences Department at San Francisco State University and a management consultant with a specialization in customer service operations. Prior to joining the faculty at SFSU, Vijay enjoyed a very successful career as an operations management consultant, entrepreneur, and executive. From 1994 to 2002, he was co-founder and CEO of Onward Inc., an operations management consulting firm based in Mountain View, CA. In 2002, Onward's customer service consulting practice was acquired by Blue Pumpkin Software, where Vijay served as Vice President for Solutions until returning to academia in 2004. Vijay's academic research focuses on mathematical models, business processes, and organizations, and he has published papers in several leading journals, including *Interfaces*, *Production and Operations Management*, and *California Journal of Operations Management*. He holds a Ph.D. in Operations Research from Stanford University and a B.A. in Mathematics and Economics from St. Olaf College.

Özgür Özlük is an Assistant Professor in the Decision Sciences Department at San Francisco State University. His research interests are in business applications of mathematical programming, practices of revenue management and spreadsheet engineering. He obtained his Ph.D. in Operations Research from UNC Chapel Hill in 1999. After getting his Ph.D. degree, he spent three years working for an airline scheduling company and at a revenue management consulting firm.

Communications of the Association for Information Systems

ISSN: 1529-3181