FAILURE AT THE SPEED OF LIGHT

PROJECT ESCALATION AND DE-ESCALATION IN THE SOFTWARE INDUSTRY

Master of Arts in Law and Diplomacy Thesis

Submitted by Thomas B. Hickerson

Spring 2006

http://fletcher.tufts.edu

THE FLETCHER SCHOOL

TUFTS UNIVERSITY

Acknowledgement

This thesis would not have been possible to write without a number of kind, generous and loving people that helped me on my way here at the Fletcher School.

First and foremost, I want to thank my wife and my parents for all their love and support during my time at the Fletcher School. My wife Svitlana has and will always be my support, my love, and inspired me to seek a better life for the both of us. My parents always provided us with a loving home to come back to, even when we were many hours—and time zones—away.

I also want to take a moment to thank my MALD thesis advisor, Professor Bernard Simonin, for all his guidance, advice and support over the past year. Working with him on the thesis and in the classroom has been invaluable and inspiring. Together with Professor Simonin I would like to thank my academic advisor, Professor Paul Vaaler, and the Fletcher faculty as a whole, including Professor Carolyn Gideon, Professor Laurent Jacque, Professor Lisa Lynch and Professor Alan Henrikson.

Table of Contents

1. Abstract

Winning is a habit. Unfortunately, so is losing. –Vince Lombardi

This thesis is about failure—more specifically, failure in the software industry. Failure is an attractive subject because everyone experiences it, allowing people to understand on a personal level what will work in business—and what won't. Failure is also a concept that is difficult to define, as there are many different academic opinions on the subject.

The aim of this thesis is to take a look at notable failures in the software sector, and draw lessons from them in the context of organizational failure, looking at the internal and external factors that caused the projects to fail. It will look at three cases where software projects failed. Two of the cases were very public, involving not only corporations but also governmental bodies, including the City of Denver's Computerized Baggage Handling System project and the Federal Bureau of Investigation's Virtual Case File project. The third case deals with failure within a private company, the ArsDigita Corporation; while the company was still privately held, the information around the company's problems was made public by dissatisfied leaders and employees within the company. We shall look at the three cases through the framework of project escalation, which is defined as "an escalating commitment to the project even after signs of failure are evident". We also discuss how each project de-escalated, and what kind of effect this had on the firms involved.

In taking a look at each project, we will see that software had little or nothing to do with the failure at all. Instead, the internal and external factors surrounding each project contributed to its problems. Escalation and de-escalation are very important tools used to

prove this, as they categorize the internal faults that each firm experienced while trying to execute their goals. However, external shocks to each project also played their part in the project failure. Following the three studies are short descriptions of systems that have been created in response to the problems project managers have encountered, showing us that companies realize that escalation is a problem, and they are proposing new strategies to identify and deal with it.

2. Introduction and Problem Definition

In 1992, California's Department of Social Services started a project to develop an automated child support system. They contracted out the development work to Lockheed Martin under a $75 million contract and set the deadline for the project to be completed by December 1995. The project was stuck with a number of problems, including cost overruns, political infighting, and a flawed procurement process. The California department of Information Technology kept supporting the project, however, until 1997, when an exasperated question was finally asked by an oversight committee's report: "How bad does a particular project need to be before [we] consider termination?" The CIO of the state of California finally pulled the plug on the project in November of 1997, after direct expenditures of $100 million[1].

What is most humbling about this story of a software project, begun with the best of intentions and developed over years of time by a top, well-known firm, but, despite best efforts, did not make the grade, is that it is not an exception. The Standish Group published in 1994 the CHAOS Report, and it cited that 31.1% of software projects were

[1] Keil, Robey, 1999, p. 632.

canceled before completion. Furthermore, it cited that only 16.2% of software projects are on-time and on-budget[2]. Most of the other projects, if they were completed at all, came in over budget and behind schedule, with fewer features than originally planned for. Cost overruns for these projects averaged nearly 200%. In some accounts, at least one in four projects end in failure.

This phenomenon is called project escalation. Like a runaway train, IT projects that get out of control are very hard to stop or redirect. This occurrence has been well-documented in management journals, and there is now a considerable amount of literature that describes what creates escalation and what can be done to avoid it. An often-cited case dealing with escalation was Denver International Airport's (DIA) computerized baggage handling system, which was a project started with high hopes and expectations, but soon went out of control and caused serious problems for the airport and the City of Denver. Another high-profile blunder was the FBI's Virtual Case File, where the FBI finally pulled the plug and built another system to replace it with commercial off-the-shelf software.

In this thesis, we explore the hypothesis that software projects experienced escalation not because of the software itself, but because the business processes behind the project were flawed or found lacking in certain respects. In the case studies below, we look at firms that struggled with organizational change and experienced failure while creating software. Often, internal, badly-executed processes to create software scuttled the project; however, it is also worth noting that not only internal but external forces acting on an organization played a role in damaging the execution and project management processes within a firm. Failure in this particular market—software—is particularly topical for several reasons.

[2] Accessed on September 22, 2005 at http://www.standishgroup.com/sample_research/chaos_1994_1.php.

One: software is a product that is borderless. Since it can be transmitted internationally with little or no barriers, and it is a thing that can be created anywhere and used anywhere else.

Two: software is a product that has been created in a cerebral nature. Estimates to create 'good' software will often depend directly upon the ability of the programmer to think, plan and execute efficiently, otherwise the product has a high rate of failure. This creates a situation where there is unequal knowledge between the developers and management, and can lead to a situation of information asymmetry, which is described in more detail under the Agency Theory framework in a later section.

Three: software is a product which has linked itself directly to the markets. In the recent boom-and-bust episodes, the IPOs of software companies were accelerants in an already fiery market. More recent episodes of market change have occurred thanks to the emergence of Google's IPO, which has created another swell in the "Web 2.0" league of companies[3].

In the case of the ArsDigita Corporation, for example, a private firm was profitable before it accepted external funding from venture capitalists, and then the company was embroiled in internal struggles between the old and new management until the company's sale of assets to Red Hat Linux in 2002. External forces, namely the dot-com bust, aggravated the crisis and hastened the firm's demise, but internal change, namely the influence of new partners, changed the firm through political infighting and weakened the firm's ability to execute effectively.

[3] See Paul Graham's essay entitled "Web 2.0": http://www.paulgraham.com/web20.html, published November 2005, last accessed April 4, 2006.

Faced with the pressures of internal and external events, many firms' efforts to create a software system can end in failure, even in the best of situations. What, then, is to be done? How can this failure be avoided? Can runaway IT projects change course in mid-stream?

In this thesis, we also take a look at frameworks and process models that have been developed in the aftermath of spectacular project failures, not only including escalation but also project de-escalation. Three case studies will be presented, showing the internal and external factors that scuttled execution and proper project management. In this thesis' conclusion, several new processes are presented that have been developed to avoid poor oversight of software projects.

2.1. Methods

This thesis will look at aspects of failure in the software industry, especially in terms of IT escalation. To do this, the rest of this thesis will hold to a structure of inductive inquiry into three separate case studies of firms and failure. The logic of inductive inquiry is useful for understanding and insight to occur when rigorous examination of relevant data is conducted, then compared, then reevaluated[4].

In exploring the frameworks of project escalation and de-escalation, it is worthwhile to review the overall concept of failure in the organization. Mellahi and Wilkinson survey the different definitions of organizational failure and group them into two classes: Industrial Organization/Organization Ecology (IO/OE), and Organization Studies/Organizational Psychology (OS/OP).

[4] Yin, 1994.

While each grouping of theories disagrees on several issues, there is enough agreement to warrant their collection under two ways of looking at failure. Industrial organization scholars argue that industry matters more than the firm, organizations are embedded in their environments, and external factors have more power over the firm and its success/failure than the managers, who have little control over the situation. IO is grounded in economics, for example, and stems from the Schumpeterian thesis of 'creative destruction' where external shocks generate waves of failure and disrupt entire industries through change. Organizational ecology experts have also developed a large body of statistical work to measure failure and adhere to four factors that determine failure in a firm: population density, industry life cycle, organization age, and organization size.

The OS/OP point of view is separated from this for several reasons; first, they argue that managers are the primary decision makers, and their perception of the external environment dictates how they succeed or fail through their management of the firm. Overall, this school of thought states that "failure is linked to internal inadequacies in dealing with external threats"[5] and goes on to categorize the internal problems that can cause failure.

Unfortunately, OS/OP scholars do not agree on one grand theory to explain failure, but instead rely on several "middle-range theories" which relate to one another. Mellahi and Wilkinson criticize the OS/OP scholars on their over-reliance on internal factors, but also state that the IO/OE scholars avoid internal factors entirely. They conclude by stating that a new framework, one that includes internal and external reasons, should be put forth to combine all schools of thought on organizational failure into one.

[5] Mellahi and Wilkinson, 2004, p. 28

Project escalation and de-escalation fits within the idea of the "middle-range theory" as it is an internal problem, dealing with perception by firm managers. However, keeping in mind that external factors are also crucial to understanding each case in this thesis, we include sections on each firm's external problems and the role they played in project escalation.

This thesis begins with two very public cases; the case of the Denver International Airport and the FBI's Virtual Case File. Both these cases are very important because a) there has been a lot of data generated about each case and its context in terms of software and failure, and b) research on that data has generated a number of frameworks that allow us to analyze other failures. These frameworks have been crucial in creating the body of work that studies IT project escalation and de-escalation. The thesis goes on to apply these frameworks to a third case.

The third case, known specifically by the author is the case of the ArsDigita Corporation. ArsDigita was a very private occurrence because the company was privately owned and all litigation was (eventually) settled out of court. Including this case is important because the thesis will analyze the third case, which has not received a lot of attention in the media or the scholarly world, and apply the frameworks that were discovered in the process of researching the first two cases.

2.1.1. Frameworks Used: Introducing Escalation

The frameworks discussed in this thesis focus on escalation because project escalation is part science but also part psychology; faced with a difficult project that has gone wrong, how do managers and decision-makers perceive that there is a problem. Once they determine this, how do they turn things around? Escalation has been defined in many

different ways, including the increase of controllable project costs beyond planned levels[6], or in terms of negative project status, where there are significant performance problems in one or more areas of cost, schedule, functionality or quality[7].

Escalation theory, in a study done on the DIA case study by Mahring and Holstrom (1999), categorizes factors that lead to escalation in four groups. The groups are listed below to show what can lead to an escalation in project resources:

Table 1. Factors that contribute to project escalation

Category	**Description**
Project Factors	• Investment character of the project • Long-term payoff • Large size of payoff • Resources seen as plentiful • Setbacks perceived as secondary
Psychological Factors	• Personal responsibility of failure • Ego importance of failure • Prior success or reinforcement • Prior expenditures irreversible
Social Factors	• Responsibility for failure • Norms for consistency or 'hero effect' • Public identification with a given course of action • Job insecurity
Structural Factors	• Political support • Institutionalization/bureaucracy

There are a number of theories that explain why escalation happens in the first place; while the above factors are certainly a quick list to review after things have already gone wrong, there are also several theories employed to try and prevent escalation before things go wrong.

Self-Justification Theory (SJT) is one such explanation, stating that individuals tend to escalate their commitment to a course of action in order to self-justify prior behavior.

[6] Orli, 1989.
[7] Brockner, 1992.

SJT is based on the idea that "individuals seek to rationalize their previous behavior… against a perceived error in judgment"[8].

Prospect Theory has been argued to be a better explanation for escalation theory, since it takes into account risk and an individual's attitude towards how a problem is framed. Originally introduced by Whyte (1986), the theory states that individuals will exhibit risk seeking behavior in choosing between to negative alternatives, especially when the choice is between a sure loss—the initial loss on the investment—and the possibility of a larger loss combined with a chance to return to the reference point.

Agency Theory is defined as "a contract under which one or more persons (the principals) engage another person (the agent) to perform some service on their behalf which involves delegating some decision making authority to the agent"[9]. Goal incongruence between principal and agent can create a situation where the agent tries to maximize his or her own utility and not act in the best interests of the principal. Information asymmetry is also a deciding factor in the agency theory question, since agents are assumed to have private information which the principals cannot access[10].

Approach Avoidance Theory is an alternative view put forth by Rubin and Brockner (1975). Under this theory, escalation is a behavior that results when driving forces that encourage persistence outweigh the restraining voices that encourage abandonment. The competing forces create a situation where there is conflict over whether to continue or withdraw. The restraining force is typically overcome by one of three motivations:

- The size of the reward for goal attainment
- The cost of withdrawing

[8] Staw and Fox 1977, p.432.
[9] Jensen and Meckling, 1976, p. 308.
[10] Baiman, 1990.

- The proximity to the goal

Based off of Approach Avoidance Theory is the concept of the completion effect, that the motivation to complete a goal gets stronger as an individual gets closer to that goal[11]. The desire to achieve task closure can have significant influence on bringing about escalation behavior.

2.1.2. Frameworks Used: Introducing De-escalation

The twin to escalation theory is de-escalation theory, or the process how a manager will extract themselves from a project that has already escalated its resources. While escalation has a lot of research and theories attached to it, the science of de-escalation is something that still deserves further treatment. Keil and Robey (1999) attribute de-escalation only taking place when managers either learn to manage existing resources better or changing the level of resources committed to the project. Keil and Robey go on to list a number of things that can change the project and they go on to posit a relationship between these factors and de-escalation. They include:

- Change in top management support
- External shocks to the organization
- Change in project champion
- Organizational tolerance for failure
- Presence of publicly stated resource limits
- Consideration of alternate uses of funds
- Awareness of problems facing the project
- Visibility of project costs
- Clarity of criteria for success and failure
- Organizational processes for evaluating decision makers
- Regular evaluation of projects
- Separation of responsibility for approving and evaluating projects

Keil and Robey went on to describe an experiment where they interviewed seventy-five auditors, who had indicated in a previous survey that they had dealt with both a period of

[11] Conlon and Garland, 1993, p. 403

escalation and de-escalation. In their findings, they measured factors present that contributed to escalation and de-escalation of the project. This included a structured portion of the interview, where factors that contributed to the escalation or de-escalation of a project were present or not. When paired *t*-tests were run on the results, it showed that the profile of the escalation phase was markedly different than that of the de-escalation phase of the project, and that the de-escalation was "associated with significantly less tolerance for failure and significantly more publicly stated limits, more awareness of problems, more clarity of criteria for success or failure, more outcome-oriented evaluations, more regular evaluation of projects, and more separation of responsibility for approving and evaluating projects"[12].

Keil and Robey went on to discover that only through the actions of actors could projects enter the de-escalation phase and turn their nonproductive processes around. Typically the actors were senior managers, internal auditors or external auditors, and it took efforts to overcome two effects, the "mum effect", or the reluctance to announce bad news about the project, and the "deaf effect", or the unwillingness to listen to bad news about the project.

2.2. Data Sources

Primary sources of data for the first two cases were accounts in the media and reviewed publications, which were compiled and written by journalists, scholars, and other investigative personnel. The sources usually took the form of articles that were based on direct interviews with active stakeholders in each project, or press-releases from the

[12] Keil and Robey, 1999, p.73.

stakeholders' firms. Secondary sources include articles on theories and management frameworks that cite the cases specifically as a cautionary tale or major example.

Primary sources of data for the ArsDigita case are court documents from the case *Shaheen v. Greenspun*, which took place in Delaware in the spring of 2001, and individual reports from employees on the activities that went on inside ArsDigita Corporation during that time. Additional sources for that case were gathered from media articles and press releases from ArsDigita and Red Hat Linux.

Analysis on the data was done by first assembling case histories from the sources and presenting each case's history, from inception to demise. The problems that occurred in each story is then grouped into internal and external factors, the criteria for this being what problems with the project, when they occurred, were either under the stakeholders' control or not. After relating each factor, a summary is made to compare factors in the case with the frameworks introduced above, and to consider what could have been done in efforts to de-escalate the project.

3. Case Study 1: Denver International Airport

The first case is not so recent, but still often cited, and is about the software problems that plagued Denver International Airport (DIA) from the period of 1993 to 1995. The computerized baggage handling system of DIA caused the airport to postpone its opening four times, at a total cost of half a billion dollars. The computerized baggage handling system (CBHS) fell far short of all expectations because of a number of factors, several of which are explored below.

3.1. History of DIA

The planning and inception of the DIA came as early as 1987, when the city of Denver together with the Federal Aviation Administration completed a master plan to create the world's most efficient and the nation's largest airport. Construction would begin in 1989 and the projected completion date at that time was October 1993. The original master plan called for each airline to install their own baggage handling system, as was customary in previous airport construction projects. United Airlines, which committed early to the DIA project, was the first to start work on a baggage handling system, aiming for an advanced solution. To this end, they commissioned BAE Automated Systems Inc. to build the CBHS at the new airport. BAE was considered a leading manufacturer of these systems, with a solid track record of past performance.

Due to the planned size of the airport—53 square miles—the initial plans always meant to push the envelope as far as technology and operating an airport were concerned. For the systems that ran the airport, distributed client-server architectures would be put in place to run all sorts of specialized functions, from security to air conditioning. Baggage handling systems would also run on a client-server system, which meant that central control of the system would be placed outside the control of the airport's information systems department.

In 1991, DIA managers began to think about the idea of an airport-wide CBHS at the airport and the benefits that it could bring to the project. Only United and Continental had committed to the project so far as leaseholders and, as one DIA manager explained, "airlines other than United simply were not coming forward with plans to develop their own baggage systems." At that time, airport planners began to develop specifications for an airport-wide CBHS and started to contact firms for bids on the project. One of the

firms was BAE, but fifteen other firms were contacted as well. Only three other firms submitted plans for the system, and a consulting firm hired by the airport recommended against all three, stating that the configuration would not meet the airport's needs and would not be on track for the 1993 opening.

BAE elected not to bid on the program at the time. However, the fact that BAE had started work on the CBHS for United convinced the DIA project team to approach them about building the airport-wide CBHS. After some negotiation, BAE proposed building what one article called, "the most complex baggage-handling system ever built." According to one account, it was to include 3,100 independent "telecars" on 22 miles of tracks and six miles of conveyor belt to route and deliver baggage among 20 different airlines. BAE built a prototype system in a warehouse near its manufacturing plant near Carrollton, Texas, and the prototype systems was enough to convince top management to move ahead with the plan.

In April of 1992, BAE was awarded the contract of $175.6 million to proceed with design and construction of an airport-wide CBHS. Gene Di Fonso, President of BAE, recalled later about the sessions with airport management to nail down deadlines and freezes to the plan; "We placed a number of conditions on accepting the job…The design was not to be changed beyond a given date and there would be a number of freeze dates for mechanical design, software design, permanent power designs and the like."[13]

For a number of reasons outlined below, the schedule for the CBHS was delayed and stopped the airport from opening four times; Mayor Wellington Webb delayed the airport from opening on October 1993 to December 1993, then March 1994, then May 1994. A number of factors caused the delays, including electric power problems, software

[13] Mahring et al, 2004, p. 219.

problems, and the fact that the client-server architecture, in a place so big, made problems in the system hard to track down and eliminate. BAE was also criticized for overreaching in engineering literature as the delays started to become public knowledge. One reviewer pointed out that, while the computer system did know what to do with the telecars that were full, BAE had no experience with empty cart management software, or in other words, how to maximize where the telecars should go after fulfilling a delivery[14].

In late April 1994, BAE prepared the first test of the system, and the City of Denver invited reporters to witness the event. The system experienced such a huge catastrophe that Mayor Webb had to delay the opening once again—reporters observed discarded clothes and other items from luggage under the telecar's tracks, the subject of several previously undiscovered errors of the system. "There is only one thing worse than not opening DIA," stated the Mayor soon afterwards, "that is opening the airport and then having to shut it down because the [CBHS] doesn't work."[15]

3.2. Internal Factors with DIA

During the course of the project, several things happened to place pressure on the project, encouraging several aspects of risk which sent the project down the wrong path.

Drastic Personnel Changes. In October of 1992, six months after the grant had been awarded to BAE, the chief airport engineer of DIA, Walter Slinger, died suddenly. Slinger, who had been heavily involved with the negotiations surrounding the BAE agreement, was an autocratic leader who was very detail-oriented. His replacement, Gail Edmond, a woman who had worked closely with Slinger, was a more consensus-oriented

[14] Anonymous, "DIA Has Happened Before", p. 7.
[15] Ibid, p. 219.

leader, and was not given as much authority. As one United Airlines project manager later explained, Edmond "had a good understanding of how the project was organized and who they key players were, but the City council didn't give her anywhere near the autonomy and authority that Slinger had."[16]

Changes after the so-called 'freezes' had taken place. Each airline began to submit more changes after the system freezes had supposedly taken place. Di Fonso later remarked, "we kept asking the City to take prompt action to assure BAE the ability to continue its work in an uninterrupted manner. Without the City's help, the delays in BAE's work quickly became unrecoverable." Di Fonso stated that soon after the contract had been signed to begin work in April, all the promises about freeze dates evaporated.

Struggles between the airport and BAE after the delays started. Soon after the disaster and the public demo, the City of Denver hired a German consulting company, Logplan, who began to audit BAE's work and issued an independent report on what could be fixed in a short period of time. They eventually recommended an entirely different system, replacing all the work that BAE had done.

3.3. External Factors with DIA

The investment funding of the airport. Raising a bond issue for the airport brought in many different parties to the airport, and they did not react well to the news of the airport's rescheduled openings. Brokerage houses sold more that $3 billion in bonds to thousands of shareholders, and when delays began, they turned around and began to push for the U.S. Securities and Exchange Commission (SEC) to get involved. The SEC finally did step in and launched a two-year investigation, centering on whether the city

[16] Ibid, p. 219.

and the underwriters misled investors about the risks of a significant delay in the airport's opening due to the new baggage system.

The costs incurred towards stakeholders involved to keep the airport running but not open. After the 1994 demonstration that showed to reporters that DIA's system was nowhere near ready, the City of Denver had to renegotiate with all the airlines to help carry the cost of not opening the airport. Tenant airlines agreed to carry the costs, but those turned into problems for the consumers later on; airlines levied a $40 surcharge on round-trip tickets to Denver when the airport finally was operational, and generated ill will towards travelers to the area.

The only alternative would have been to turn Denver into an "one-airport town," as one report put it. Many of the airlines, frustrated at the delays, went ahead and reduced the number of flights arriving and departing from Denver. This created problems later on, as Denver International Airport, once touted as the airport that would serve its passengers with direct international flights, would now only serve the U.S., Canada, and Mexico. More problems and costs, such as the $38 average fee to get to the City of Denver from the now-faraway airport, was also an impetus for travelers to avoid DIA as a destination.

An investigation filed by the Securities and Exchange Commission, and then by the Colorado State Government. The CBHS was not the only problem plaguing the DIA. Accusations of financial wrongdoing prompted an investigation first by the SEC and later by the Colorado attorney general's office. The Attorney General specifically looked into things such as dilution of concrete for runways and improper use of funds. More problems came later in 1996, when the FAA was called in to investigate air-traffic

control systems reported failures such as controller's screens going blank while tracking passenger jets.

3.4. Epilogue of the DIA system

Faced with the combination of external and internal pressures, Mayor Webb was forced to abandon his commitment to the CBHS. A system, originally billed as a "back-up" system, based on conventional belt conveyors and carts pulled by tugs, was quickly implemented. This became the substitute system to handle baggage at DIA. By the airport's opening in 1995, the manual system served two concourses and the CBHS served United's outbound passengers only. By 1996, United fired BAE from the job of operating the CBHS, citing among other problems a track record of 500 misdirected bags per day, and both parties sued one another. BAE claimed that United would never certify the system complete and continued to derail the project by cutting testing periods of the system short.

BAE was eventually acquired by a competitor, G&T Conveyor Co., in 2003, shortly after United declared bankruptcy in 2002. Only in 2005 did United finally give up the automated system entirely and return to manual systems for handling baggage at Denver, citing costs of $1 million a month for CBHS maintenance alone.

DIA cost the City of Denver, and the city's taxpayers, close to $1 billion in keeping the airport closed, not counting the number of fees that tenant airlines later tried to extract from its customers. Once called the "most mismanaged place in America,"[17] the airport is synonymous with project failure in some circles, but is considered a success now, in certain circles. Restaurateurs, for example, suffered together with everyone else until

[17] Henkoff, 1995.

DIA opened for business. "The wait was painful," stated a McDonald's franchisee, but quickly added that now McDonald's generates $5 million in sales a year at DIA.[18]

Even now, engineers still use DIA's system as a harbinger for planning new materials handling systems. "Courage is the innovator's most valuable asset," one magazine article in the trade publication *Material Handling Management* read, after it reiterated mistakes learned from DIA and noted that, of the dozen cart systems implemented since Denver, none of them were implemented in the US. "A good idea badly implemented," concluded the article, "is too often only remembered as a bad idea."

3.5. Lessons Learned with DIA

One of the things that daunt organizations is the decision making progress, and how it can lead projects, teams, or entire firms down the wrong path. Brockner (1992) introduced us to escalation theory, which refers to the tendency of decision makers to persist with failing courses of action. Escalation occurs because of a combination of two factors: our rational side that considers values associated with commitment to a certain course of action, and our self-justification side, which reflect our unwillingness to admit that a certain course of action was initially mistaken. In a study comparing DIA with trends in escalation theory, Mahring and Holstrom list a number of factors which contributed to the demise of the DIA project, which included the following:

- Project Factors: With a long-term investment, a large prospective payoff, and a large cost to abandon outright, the DIA CBHS looked like all it would take was enough time and development and the payoff would be worth it. With the quicker turn-around time that a CBHS could offer, other airlines would see DIA as an

[18] Ruggless, 2000, p.6.

impressive place to serve as a base of operations. It also seemed as if near-limitless resources for the project were readily available, and would be efficacious for the foreseeable future. Additionally, it did not seem like there were any feasible alternatives; only at the very end of the project did United finally accept "standard" baggage handling systems in the place of what it had thought would be the "next standard" in baggage handling systems.

- As an aside, the habit of viewing the project expense as an investment, which gave expectations of future gain, generated a subjective opinion about whether or not to commit additional resources. Expectancy theory is one of the tenets of escalation. It states that decision makers assess the probably that additional resources spent on the project will lead to goal attainment, and that subjective notion that affects the decision to commit additional resources[19].
- Psychological Factors: With a prior history of success and the high personal visibility of the project, many of the managers involved convinced themselves that "things did not look so bad" and continued down the same project path despite warning signs. Also, the expenditures spent on the project, since they were in construction costs, could not be recovered, and contributed to the idea that "we can't go back". The sense of personal responsibility (as the election campaigns were based on development of the City of Denver and the DIA) was also a driving force behind escalation.
- Social Factors: Behavioral norms that promoted "staying the course" even after the parties involved did not believe in the project also encouraged the project to continue. The "heroic" ideal to turn a project around contributed to this, as did

[19] Brockner, 1992.

the preference for consistent behavior. The sense that being associated with a failing job also drove decision makers to escalate, as it was an incentive to hope that it could somehow be turned around and "save the day."

- Structural Factors: The social and political context of the project also created high stakes, since several city-wide political campaigns were based on the airport's success or failure. Also a factor to consider was the organizational culture and administrative inertia which came with the project stakeholders involved in the work. The deliverables of this project were embedded into the organizations that worked on it. Thus, nothing could be removed from the 'critical path' to allow the project to succeed without it, creating an 'all or nothing' scenario.

Mahring and Holstrom's study of the DIA story as seen through escalation theory tell the story in four phases, and then link different factors as playing a role in the steps that lead to abandonment of the project. In the Mahring study, for instance, problems began to emerge as the governance style of the project changed with the change in personnel; the death of Slinger and the subsequent replacement by someone with less authority and less credibility made the governance structure less stable. External pressures only came to bear on the City of Denver and the project after the change, and the subsequent disaster of the trial run in April of 2004.

Of course, the twin to escalation theory is de-escalation theory—that is, how managers try to extract themselves from a project that has already escalated its commitment. Only the severe, external shock brought the CBHS project managers to the conclusions that there had to be change at DIA. Other factors that Keil and Robey list, ironically enough, include publicly stated limits of resources, separation of responsibility between approval

and evaluation of a project, and the presence of alternative uses of resources. With a mid-project change in governance style, apparently limitless resources, and no visible alternative, these were things that were simply not seen as factors during the conception of the DIA CBHS project.

Once the project did change, the City of Denver was able to de-escalate the project by hiring an external auditor, Logplan, who then recommended that an interim system be put in place quickly that would be less complex to implement. With the CBHS taken off the critical path for the airport to open, parts of the CBHS could finally be salvaged on a less aggressive schedule and eventually phased out entirely.

4. Case Study 2: The FBI and the Virtual Case File

> *"[In] December they gave us the software, in January we went back to them and said, well, there are some deficiencies in the software. They said, well, there are 17 deficiencies, we decomposed the 17 deficiencies, they turned into 59 deficiencies. Then we had a two-week sit-down with SAIC [Science Applications International Corporation] and those 59 turned into 400 deficiencies… You know, I have a base-line software that I was told that 90 percent was ready, yet they were asking for another $56 million to develop the other 10 percent. Now where's the logic in this one[?]"* —Zalmai "Zal" Azmi, Chief Information Officer, FBI, Feb. 14, 2005

The above quote was part of a series of articles published by *U.S. News and World Report* in June 2005, focusing on the cancellation of the contract to develop a Virtual Case File (VCF) for the Federal Bureau of Investigation. The VCF project had been under scrutiny in 2003 for database design and testing approach, and the FBI Cyber Division returned a report in 2004, identifying specific functional deficiencies. In January 2005, the Aerospace Corporation issued an independent report on the VCF, and recommended discarding the product and starting over with a commercial product[20].

[20] U.S. Congress, House, 2005, p. ii.

SAIC, the original contractor, added insult to injury by reporting that it needed over $50 million more to complete the first phase of the project. In March 2005, FBI Director Robert Muller pulled the plug on the VCF project, running up costs of over $107 million dollars.

4.1. History of the VCF

The original contract for the VCF, a part of a program called Trilogy, was awarded to SAIC in summer of 2001. Trilogy had three components: replacement of workstations and hardware, replacing old infrastructure to provide network connectivity, and a part called the User Applications Component. Originally designed to replace five legacy systems, the UAC was planned to be implemented in three stages. Soon after September 11, 2001, however, the focus of UAC quickly changed, and the plan was radically altered to favor a quicker approach. The pace was quickened on the project especially as the Congressional investigations after 9/11 identified that an inability to share information led to the tragedy. Work on Phase I, which was to enable an aging Automated Case System for the web, was stopped, and the plan for Phase II was skipped entirely. Instead SAIC was told to move ahead with Phase III directly, which was the creation of the Virtual Case File, a "case management system for the enterprise solution," as one government report described[21].

The technical and functional deficiencies described earlier came to light after SAIC's Delivery 1 of the VCF in December of 2003, and over the course of a year and a half, exposed a number of consulting and management weaknesses on the part of the FBI.

[21] Ibid, p. ii.

4.2. Internal problems with the VCF

As in our previous case, we can identify internal and external processes which derailed the project, which were just as to blame as the problems in the software itself.

The changes in the project mid-course were too ambitious. After 9/11, the project's character changed drastically, and a process that was supposed to take hours would now have to be completed in three seconds. The VCF contract became a more ambitious system that would handle millions of case files in a variety of formats with a three-second response time to specific queries. To make matters worse, the system was to be implemented in 22 months and would replace an old mainframe system called the Automated Case Report System in a "flash cutover" strategy, which essentially meant that the system would take the place of the old system overnight.

System requirements were also lacking originally, and then were added too late. When the contract was signed in 2001, the FBI did not identify all the requirements right away. Even when the character of the project changed in 2002, not all the requirements were spelled out yet. To meet the December 2003 deadline, SAIC hired 250 full-time positions to work on the project. The FBI went through 19 IT project managers from November 2001 until the deadline, however, and with each change there was also a change in the requirements. "Each change brought new directions, a different perspective on priorities and new interpretations to requirements,"[22] one SAIC executive later testified to Congress.

The requirements document that was finally drafted was about 800 pages long. After it was approved, the SAIC contractors began to work on the software that was based on a

[22] Holmes, 2005.

spiral development model, where small, iterative changes were made to the code and then shown to the FBI. The reviewers would then submit a list of desired changes. This set of changing requirements added to the scope of work, and made late schedules even later.

Secrecy, nondisclosure of information, and downplaying the importance of IT was ingrained into the FBI culture. Trilogy program manager Sherry Higgins, who had previously handled programs for the Olympics, AT&T and Lucent, struggled with dealing with the FBI and finally left the project in July of 2004. "They'll throw good money after bad to cover up a bad decision,"[23] she was later quoted as saying. As an example, Higgins recalled trying to find out why the FBI embraced mainframes; it took several meetings and interviews with fifteen FBI IT personnel before one person simply said "It's that way because leadership just spent $10 million on a mainframe, and they don't want it to go to waste". Higgins also recalled that the FBI always downplayed the importance of IT, and saw changes to the requirements document as inconsequential at best to the success of the project. "The culture within the FBI was, 'We're going to tell you how to do it'," she later said[24].

VCF, if it ever worked as envisioned, would turn the agency culture on its ear. FBI agency culture at that time was based on two tenets; sharing information was wrong, and IT was not an important tool for doing their job. FBI's dismissive attitude towards IT was shaped by former FBI Director Louis Freeh, who viewed IT as a back-office support function. Mueller, when he took office in 2001, promised to reshape the policies around IT and promised a fresh start for the Bureau.

[23] Holmes, 2005.
[24] Goldstein, 2005, p. 31.

Part of this fresh start included hiring a new CIO for the FBI. After a number of CIOs came and went, Azmi took the job. Formerly the CIO for the U.S. Attorney's Office, Azmi recalled that one of his first shocks came when he asked how big his IT budget was. The answer was $5,800. The rest of the money was controlled entirely by field offices. "Nothing really shocked me more,"[25] he recalled. Azmi also found out that the FBI had not complied with a law signed by President Clinton, dictating the responsibilities of federal CIOs. There was no basic IT policy, no strategic plan, and no enterprise architecture. Also, Azmi noticed from the outset that there were not enough FBI personnel to monitor the work of the SAIC contractors. The FBI could not keep up with communicating the expanding requirements for VCF to SAIC developers, and SAIC were just filling in the blanks and making too many key development decisions for themselves.

4.3. External Problems with the VCF

Whether SAIC was to blame for asking for more money after the deficiencies were exposed may or may not have been a core reason for program termination, but it is already obvious that a number of mistakes were made in the internal planning and organization of the project. What follows are some of the external problems that plagued the overall arrangement of the VCF project:

There were contracting problems with the VCF at the outset. The FBI, citing lack of contracting manpower, used a Government-wide Acquisition Contract as the method of managing the grant money behind Trilogy. The GAC is a model that removes management control from the government agency. In this case, the FBI had to relinquish control and allow SAIC to run the program. Government reports would later cite that this

[25] Holmes, 2005.

was one of the reasons why the program faltered. Additionally, lack of clear milestones established by the FBI, together with a high turnover rate for the FBI's Chief Information Officer position, added to the problem of clear leadership and definition for the project.

A political dimension also changed the way decisions were made about the project. For example, faced with the number of functional deficiencies but determined to prove that it was producing something, the FBI made the decision to test the workflow capabilities of the system out of a field office in New Orleans. From December 2004 through March 2005, users were entering information into a test system. SAIC's delivery of that part of the system was flawless, thanks to a number of factors: clear definition of requirements, appropriate milestones, and management oversight. The workflow system, however, was scrapped together with the rest of the VCF system since it was not compatible with any other system. The total cost of testing came to $17 million, and involved about 250 people.

The way information from the external contractor about VCF was being handled by the FBI. "We took for granted the message was being moved up the management chain," said one of the SAIC managers of the VCF. As it turned out, the FBI, lacking a sound project management structure at the time, had pushed most of the reporting on the progress of the VCF down to the lower levels of management. One report pointed out that a shortage of skilled program managers and engineers was one of the contributing factors to this problem[26]; another was the overall attitude towards technology. Up until the scandal with the VCF broke, each separate FBI office had its own budget for information technology. That meant that 40 or 50 different legacy systems were already in place when Trilogy was launched, and there was no Chief Information Officer or

[26] U.S. Congress, House, 2005, p. iii.

documentation detailing the overall scope of the FBI's computer systems. In short, the FBI saw information technology as not being a vital solution to getting their job done, and VCF became "an enterprise IT project that fell into the most basic traps of software development, from poor planning to bad communication."[27]

4.4. VCF Epilogue

In December of 2003, SAIC delivered the VCF to the FBI. As the quote at the beginning of the section indicated, a number of functional deficiencies were soon discovered. When SAIC offered to fix the hundreds of changes in spring of 2004 for the extra cost of $56 million, Azmi rejected the proposal. Azmi instead took the workflow system which had been tested thoroughly in New Orleans and used that to test the development life cycle of creating a project. The project rolled out on time and under budget, but according to an internal FBI memo, the application "did not improve the productivity of most users"[28].

As of the writing of this thesis, the FBI has announced that a new replacement of the Virtual Case File, dubbed Sentinel, will be phased in over the next four years. Sentinel will be built using commercial off-the-shelf software, together with a different style of project management that was tested with the roll-out of the workflow system under Azmi. Different legacy systems will be rolled into each phase and tested thoroughly before the legacy systems will be finally retired.

[27] Goldstein, 2005, p. 26.
[28] Ibid, p. 35.

4.5. VCF and Lessons Learned

Looking at the history of the VCF and the project through the lens of project escalation, we can see the factors that we explored originally in the context of DIA.

- Project Factors: The long-term payoff was a huge factor, since a complete record system that all FBI agents could use was seen as highly important and incredibly valuable. The changing character of the project was also something that encouraged escalation, since from Trilogy pre-9/11 to the VCF the schedule and capability of the system had to change.

  The perceived infeasibility of alternatives was also a factor, since SAIC had essentially tried to build the entire VCF system from scratch. Embracing off-the-shelf alternatives, the Sentinel project promises to at least avoid this mistake.
- Psychological Factors: For a high-profile organization such as the FBI, the ego importance of failure was very high indeed. The culture of not using a computer to gather and research intelligence was also a factor. The poor benchmarking and requirements gathering also affected the project, with FBI overseers instead relying upon a "I'll know it when I see it"[29] attitude to determine when the project would be ready.
- Social Factors: The institutional methods which the contractor used were also factors into the downfall of the project. The FBI's own sense of rivalry against other law-enforcement and intelligence-gathering agencies only served to create additional friction. The task of changing that cultural attitude of both secrecy and

[29] U.S. Congress, House, p. 11.

unimportance of IT finally fell to Azmi, after four other CIOs came and went between 2002 and 2003.

- Structural Factors: Changed by the initial shock of activity after 9/11, the Trilogy project quickly had to become something it was initially not intended for, and that was one of the ways that the project was flawed. Since it was labeled the "government's primary weapon to stop terrorism," the pressure for the project to produce results was higher than other software projects handled by the FBI in the past. The project quickly became very political, and arguments about the weaknesses of the project were quickly silenced.

De-escalation of the project could only come at the very end, again in the form of an external shock. The actor who got involved in turning the project around this time was Azmi, since other CIOs and VCF project managers who came and went during the project's lifespan were too institutionalized in the aforementioned "mum effect" in place at the FBI. The workflow portion of the code was developed, tested and released through the New Orleans office using a number of techniques that we earlier listed in the de-escalation section, all factors that Azmi dubbed the Life Cycle Management Directive. This included:

- A strict timeline;
- Clear acceptance criteria;
- Daily meetings of the project manager and Azmi himself;
- Adherence to baseline requirements.

This clearly embraced Keil and Robey's factors of de-escalation, including its clear definition of success or failure, regular updates on project status, and a clear awareness of

the problems facing the project. Only after splitting the workflow project off from the monolithic VCF and testing it under Azmi's management, which was quite different than the rest of the VCF had been managed, was the FBI able to hammer out a repeatable process that could then be used in future projects, such as Sentinel.

5. Case Study 3: The ArsDigita Corporation

In the introduction, we discussed that internal and external forces can affect a firm and its technical capacity. However, how does this affect private firms, especially smaller players in different markets? The following example delves deeper into a case where software creation was one of the issues, but also change management, ownership, and a clash of organizational culture were all issues that decided the fate of not only a project, but the life of an entire firm.

5.1. History of the ArsDigita Corporation

ArsDigita was a profitable corporation in spring of 2000, when it attracted venture capital funding with eventual plans to grow the enterprise towards an initial public offering. The relationship between the directors of the board, the founders, and the new management deteriorated over the course of a year, however, and resulted in a lawsuit over a breach in the Shareholders' Agreement when CEO Philip Greenspun tried to gain control of the company again. The affair was settled out of court, but due to the culture of openness at the corporation, the details of the case were publicized widely. ArsDigita concluded its lifespan with a sale of the assets to Red Hat Linux in spring of 2002.

ArsDigita was an unusual example since the change management issue here was perceived as an initial success, but turned into a serious problem which decided the fate

of the company. The pressure to deliver products "on Internet time" and in a fashion closer to traditional revenue models, eclipsing the non-traditional values that had made the firm noticeable from the beginning, drove a wedge between new management and old. In the beginning of the company's history in 1999, ArsDigita's revenue streams were mainly services and support, and the company forged several important partnerships with hosting resellers so that they could keep their staff small and focused upon their core competence; making websites. Open-sourcing the code for the ArsDigita Community System, their sole software product, turned out to be a smart move for ArsDigita. Their code rode the wave of viral marketing, having developers recommend the code to other developers. The ArsDigita staff also ran bootcamps to teach outside developers how to use the ACS, and this allowed the developers core to the project to find others that they could work with.

Philip Greenspun, as the leader of the company, was not the picture of the traditional CEO; not only did he travel everywhere with his Samoyed, Alex, (a 1999 newspaper article sports them together in a photograph, and his personal website was covered in pictures of them together as well) but he also implemented ideas that created PR for the company in unconventional ways. For example, as the company grew, Philip chafed at the idea of hiring headhunters to find new talent, so the company rented a high-end sports car and the first employee to recommend five new recruits got to drive it for a month.

Greenspun had other ideas which were unconventional, but also got him and the company noticed. He refunded a day's tuition to his MIT students during one of his lectures on web engineering[30]. He also started several non-profit initiatives, including ArfDigita (a website for finding available pets in shelters), the ArsDigita Prize (a contest for high-

[30] http://www.mensetmanus.net/inspiration/tuition-free-mit/, last accessed March 18, 2006.

school age students to create websites), and ArsDigita University (a free educational program to teach an entire computer science curriculum to returning students). His personality and willingness to "walk his talk" was something that got the company in the press, and also attracted a lot of like-minded talent; many programmers saw ArsDigita as the kind of company they would like to work for.

With the financing, Greenspun agreed to step down as CEO, but remained a majority shareholder and a member of the board. A professional manager, Allen Shaheen, was hired in his place. Additionally, two seats on the Board of Directors now were occupied by Greylock and General Atlantic Partners personnel, and a stipulation was written into the letter of intent that "major" decisions could not be made without the express consent of either director from Greylock or GAP. An entire section on corporate governance and control of the organization was written into the Shareholders' Agreement that was drafted as a result of the funding.

Between March 2000 and March 2001, the relationship between Greenspun and the other shareholders "deteriorated significantly." While the rest of the plaintiff's pretrial brief had been redacted as to the details of the disagreements, Greenspun and his employees were more than vocal about the problems that they state began with the VCs. From their point of view, this could only result in Greenspun trying to reassert his control over the company.

One of the things that led to the implosion at ArsDigita was the clash of cultures, but another thing was that the release, support and development of the flagship product for the firm, the ArsDigita Community System, went astray due to the infighting that started after the arrival of the VCs.

5.2. Internal Factors

Greenspun would later claim that he had left all the decisions regarding the funding to counsel, and that he did not understand the Shareholders' Agreement when he signed it. He also would go on to state in the above article that the new CEO Shaheen, together with the two VC directors, stopped listening to Greenspun's advice and made many bad business decisions, without asking for his advice or input.

5.2.1. One Special Note: Culture Matters

Part of the problem was a culture clash between software engineers and venture businessmen. The notion that culture could change the style of execution within a company or project group is something that has been discussed at length by the likes of Peter Drucker, Samuel Huntington and others, valuing the ideas of culture of different groups and how it can aid or harm development.

For example, different types of technologies come up short when trying new implementations abroad. In a recent study looking at failures of Enterprise Resource Planning (ERP) software in China, it was found that not only language problems scuttled implementation projects; the Chinese managers' ability to grasp the requirements of business process reengineering, together with simple needs, such as the need to support letter-of-credit payments, or update inventory automatically, stymied attempts to consolidate older business processes into one system. While an exhaustive list was produced by the study, one quote from a local manager sums up the source of problems and process with ERP in China: "Understand China's history and culture before you start doing business with Chinese companies"[31].

[31] Xue et al, 2005.

At ArsDigita, and on the World Wide Web, Greenspun was considered an engineer's engineer, having graduated at a very young age from MIT, and having worked as a software engineer for most of his adult life around the Cambridge and MIT area. He wrote that MIT was always perceived as a "no-praise zone", and that injuring the new partners' self-esteem was the reason why they stopped listening to him. "I'm not sure how much time these three guys had ever spent with engineers," he wrote[32], and could not contain his disdain for how the company was being run by the new team. He stated openly that ArsDigita had become profitable by starting small and taking only small risks, not the huge expenditures that the new management had made after receiving control:

> We never signed up for this kind of risk and we don't have substantial other investments. I put 8 years of my life into ArsDigita Community System. Jin put in 4 years. We would be unhappy to see the company spend through its accumulated profits plus $38 million in capital merely so that three guys in suits could learn a little something about what it is like to run a software products company[33].

The internal tensions rose between the VCs, who naturally wanted control to use the company to produce new software, and Greenspun, who had brought a successful company to life on a system that was distributed, freely downloadable and open source, with very little control of the flagship product itself.

With so much of the corporate culture already focused on sharing, helping and giving away work for free, in hindsight it was obvious that there would be a clash with traditional business minds at a venture capital firm, who were more interested in buying—and maintaining—control. In one article about corporate venturing in a technology sector (the example was biotech) the traditional venture capitalist point of view was spelled out for the reader:

[32] Greenspun, 2001, accessed December 2005.
[33] Ibid, accessed December 2005.

> Typically an investor will seek control over a range of commercial matters such as share issues, grants of options, certain capital expenditure, major acquisitions/disposals and material dealings with IP. Through its influence over the investee company a corporate venturer can increase the value of its investment, for example, by encouraging the successful development and exploitation of IP rights….A director may face a conflict between his fiduciary duties to the investee company and the interests of his appointor. Therefore, mechanisms should be in place to deal with such conflict, such as providing for resolution at shareholder level. In practice the chances of any such conflict arising are reduced if the parties involved have clear commercial objectives that are agreed at the outset, which are monitored and updated as necessary[34].

Intellectual Property (IP) was one of the core moneymaking ventures for many companies during this time, but the open-source movement was anything but supportive of the notion of using IP to generate revenue.

5.2.2. Other Mistakes Made in Change Management

To make matters worse, Greenspun was still perceived as the head of the company, and would make uncontrolled statements berating the current CEO and management. One of the classic mistakes of change management, this was the other part of the internal problem between the new and old managers of the company: Philip did not step down entirely, and instead eroded the company's and the public's confidence in Shaheen and the new Board of Directors. This unwillingness to totally give up control in the minds of the employees was one of the things that contributed to the relationship going downhill fast.

The final straw between Greenspun and the new management came, when, according to Greenspun, he made an offhand comment comparing the new management trying to run ArsDigita to "a group of nursery school children who've stolen a Boeing 747 and are now flipping all the switches trying to get it to take off."[35] This enraged the GAP partner who

[34] Bushrod, 2005.
[35] Ibid.

sat on the board, and he took the initiative to remove Greenspun from the company. This was in late March of 2001, however, and by that time Greenspun had already taken the initiative to take back control of the company.

5.3. External Factors

From the point of view of the VCs, Greenspun's management and actions was as much to blame for the company's expenditures as they were. Additionally, not all ArsDigita employees painted a rosy picture about Greenspun's leadership and management:

> To me, this [Greenspun's story] is a gross oversimplification, a symptom of deeply entrenched denial. ArsDigita was a good company, especially in the beginning, and I miss it, the way that it was back then, but I think it's false to claim that ArsDigita was perfectly noble…I also think it's disingenuous to gloss over the fact that, at the end of the day (to borrow one of our second CEO's favorite phrases), we took 38 million dollars of someone else's money.
> --Michael Yoon, *ArsDigita: an Alternate Perspective*

There were several other factors that contributed to the difference between the founders and the VCs. Among them were the following:

Greenspun's own perception of spending and finances. Philip's ideas that made news also cost money. One of those ideas was a weekend house on Cape Cod to serve as a programmer's weekend retreat to work on projects and hold workshops. Philip's previously mentioned ideas also took up valuable space on the budget, and the presence of the non-profit organization was cited in the plaintiff's brief as a point for concern, though the specific reasons for this citation were publicly redacted. As the company grew, former employee Michael Yoon also mentions the problems with finding office space for ArsDigita's growing network, and what Greenspun did about it:

> One story I heard, which may be apocryphal, was that Philip demanded that the Atlanta office be fitted out with real Georgia marble. Two things I know for sure about the Atlanta office are that Philip approved it and that it was *gigantic*, with

> enough Herman Miller cubicles and Aeron chairs to accomodate over fifty programmers. The occupancy of the office never exceeded *four*…

The overall financial situation facing dotcoms and VCs in 2001. Due to the financial downturn, many of ArsDigita's clients, who were VC-funded dotcoms as well, started to close and declare bankruptcy, not paying ArsDigita. "For every Siemens, there was also a PogoPet, a Zzyxx, a Revbox, an ArtMetropolis, and a HouseHoldDirect," wrote Yoon. A serious squeeze on the revenue stream for a company that did not charge for licenses was a problem, and several rounds of layoffs had to be made in spring of 2001.

The pressure was not only felt by dotcoms, but by VCs as well. GAP was one firm that had other problems besides ArsDigita; in 2001, another portfolio company that had already had its IPO, Bindview, asked that their GAP director of the board step down and that GAP publicly distance itself from the company. Greenspun's statements about the VCs' performance did nothing to help their deflating image in the public eye.

The demand for technologies that enterprise-level customers were familiar (and comfortable) with. After releasing four versions of the ACS, under the new management, the company's strategy began to focus on rewriting the product in Java. "Why Java? Because that's what many of our prospective clients were asking for," Yoon writes. Indeed, the ACS did run on several obscure technologies, such as the Tcl programming language and the AOLServer web server. More popular technologies, it was reasoned, would lead to more clients and more revenue.

The first release of ACS under VC management came when ACS 3.3 was released in May of 2000. ACS 4.0, with a new feature set, came rapidly on its heels, coming out in November of 2000. The ACS 4.0 release was criticized because it was only released in a 'core' stage, without additional modules of functionality released with it. This was a

sharp departure from the previous releases, which has the maximum amount of functions included together with it to help developers create their own sites as necessary.

While the official statement was that the release would include modules later in the year, the developers and other firms that started working with the 4.0 release did not hold back criticism. One developer called the code "incomplete, unscalable, untested", and the founder of a small software company commented, "We have lost at least one prospective client because of it, perhaps more"[36].

Soon, the company began a push towards developing an entirely new product in Java, and resources were altered to support the Java version, rather than the previous versions of the company's product. Yoon writes more:

> Not long after, the management team, which still included me, decided that (a) that we didn't have enough resources to maintain a Tcl version of ACS and a Java version of ACS and (b) there was no way we'd make our growth projections trying to sell the Tcl version. Like it or not, it seemed that most prospective customers were reluctant to buy a Tcl-based product. Also, it was around this time that the notion of creating a closed-source product to augment the open-source ACS first emerged.

This was a serious point of contention with the old management and the new: "Don't throw out your prime source of revenue before another one is in place. Fashionable programming languages don't equal useful software", ArsDigita shareholder Eve Andersson stated, after she was let go from the company in 2002. In her online article about ArsDigita, she also voiced her opinion that "without modules, ACS is a useless product."

The drive to rewrite the code base was also a risky plan from a strategic point of view. As frequent columnist Joel Spolsky put it in his website *Joel On Software*, in an article entitled *Things You Should Never Do, Part I*:

[36] Andersson, 2002.

> When you throw away code and start from scratch, you are throwing away all that [product] knowledge. All those collected bug fixes. Years of programming work.
>
> You are throwing away your market leadership. You are giving a gift of two or three years to your competitors, and believe me, that is a *long* time in software years….You might as well just close for business for the duration.
>
> You are wasting an outlandish amount of money writing code that already exists[37].

The plan to gain a license revenue stream with the new closed-source product was a proposal that many of the founders disagreed strongly with, since they had embraced the ideals of open source for so many years. However, the usage of IP and licensing revenue was a strong driver to try and create a profitable product for the company to raise its earning potential.

5.4. ArsDigita v. Philip Greenspun

Amidst the internal fighting and the external pressures on the company, ArsDigita announced that it would begin to sell proprietary modules to work together with its core ACS product. This announcement happened on April 5, 2001. Together with this announcement was the news that Philip Greenspun would step down as Chairman to pursue other interests.

Greenspun did not know about this until the news was announced, but during that time he had been busy; to try and do what he thought he could to save his company, he held a majority shareholder's meeting and elected to rewrite the company's by-laws and re-appoint himself as CEO, also electing two of his employees to the Board of Directors, Eve Andersson and Tracy Adams. On April 6, 2001, a courier delivered a letter to Shaheen, indicating that he had been demoted from "President and CEO" to "President".

[37] http://www.joelonsoftware.com/articles/fog0000000069.html, originally posted April 6, 2000, last accessed March 19, 2006.

On April 11, a lawsuit was filed by Shaheen, ArsDigita Corporation, GAP and Greylock against Greenspun and the remainder of the shareholders who had agreed to change the by-laws, namely, Eve Andersson and Tracy Adams. The Plaintiffs cited the corporate governance clauses in the Shareholders' Agreement, and stated that Greenspun and the rest of the shareholders signed the document of their own free will, and was bound by it. Greenspun found himself in a strange predicament; the company he founded was now suing him, since the money for the plaintiff's retainer came from the company itself. In their own opening brief, the defendants made the argument that ArsDigita should "stand above the fray" and not be listed as a plaintiff together with Greylock and GAP. Additionally, it was not the majority shareholders' wish that the company be involved. The plaintiff's brief countered the argument, and stated that naming the corporation as an additional plaintiff in the suit as it was a signatory to the Shareholders' Agreement as well, and that it had a direct interest in the suit as it "may one day have to access the private capital markets again.[38]"

5.5. From Endgame to Epilogue

Greenspun published the online article *From Start-up To Bust-up* while the trial was ongoing, which made the whole affair very public. "The company's birth and growth were public," he argues in the article, and those "who were kind enough to pay attention and support us are entitled to know how the rest of the story unfolds." This invited everyone to weigh in on the ArsDigita story, and created a court of public opinion, mostly turned against the VCs. Other supporters of ArsDigita found all the court

[38] Plaintiff's pretrial brief, p. 64

documents and posted them online, so that the entire trial was under a microscope for many of the interested parties.

In the summer of 2001, the case was settled out of court. Half of Greenspun's shares were purchased for over seven million dollars, he had to resign from the board, sign a non-disclosure agreement, and agree to never try and take over the company. His tell-all article was taken offline, but naturally, found its way to another employee's web server, and is still online today. With the majority of the shares now in the hands of the VCs, the other founders were pushed out of the company with little or no settlement. The VCs did finally have control of the company, and were through with Greenspun; with the remaining time they had left, they tried to re-engineer the company, product and services as best they could.

Shaheen was relieved of his duties as CEO later that year, to be replaced by a Greylock-appointed manager. After several rounds of layoffs, and attempts to promote the Java product together with closed-source modules, ArsDigita was sold to Red Hat Linux in February of 2002. Other founders also published their take of events in the wake of the sale and closure of ArsDigita; Eve Andersson wrote about it in 2002, and Yoon in 2003.

5.6. ArsDigita: Lessons Learned

In the context of project escalation, the ArsDigita case can be analyzed much like the DIA and the VCF cases before it. Signals that escalation would happen with the change in ArsDigita's strategy from an open-source product to a closed-source product were readily apparent, only all the managers in attendance were probably struck with the "irrational exuberance" afflicting investors at that time, which supports Keil and Robey's "deaf affect".

- Project Factors: As in DIA, short-term resources were seen as near-limitless after ArsDigita Corporation's windfall of $38 million. The long-term payoff also seemed to be big, since there were other IPOs in the news that only encouraged the team at ArsDigita to aspire to work hard, succeed, and eventually create a windfall for all the investors involved.
- Psychological Factors: Ego played a large part here, perhaps more than in the other two cases. Neither Greylock or General Atlantic determined that the firm, being entirely Philip's brainchild, was something he would take significant legal and financial risk to defend.
- Social Factors: Like it or not, Greenspun was identified with ArsDigita, so he felt that its own success or demise reflected on him. Also, he felt very much that he had a community to support with the ArsDigita suite of software, since he cited that writing the account of the lawsuit was in part because of his readers on the World Wide Web, who were "entitled" to find out how the rest of the story would unfold.
- Structural Factors: The factor of control, institutionalized as an end goal in the venture capital community, did not seem like such an important factor to Greenspun until after he accepted their money. The structure of the deal, while supposedly unknown to Greenspun, would clash with the culture of ArsDigita and the culture of the VCs who came to buy a piece of the then-successful company. In many cases institutionalization is described as a factor which erodes quality in so much as the culture that is institutionalized is usually beauracracy or slowness to react. In the ArsDigita Corporation, a

willingness to create and low tolerance of failure were the cultural institutions at the company instead, which clashed with the incoming VCs.

Could the project have de-escalated? Again, many of the things that could generate a gradual de-escalation were not present:

- Since the entire company was private, costs were not visible;
- Resources were seen as plentiful;
- Since the culture of the workplace was affected, criteria for success and failure were in flux and, therefore, unclear.

In fact, a number of the things that support project de-escalation were in flux, including evaluating decision makers, evaluation of projects, and top management support. The drive to create a Java-based product was flawed, not only because of the risk of the switchover (much like the "flash cutover" of the VCF, or the never-before accomplished baggage system of DIA) but because culture had interrupted the firm's execution to such an overwhelming degree. One of the final comments on Michael Yoon's webpage is from a client, whose future was tied to the ACS creating an enterprise portal for them:

> I dare say that what really happened to ArsDigita was somewhere in between Eve and Michael. The company did lie. They lied to my face on at least ten different occasions, and in the end were more concerned about growth, and hiring employees, than their customers. (Can a company really grow that fast?) I doubt that this was the previous culture that had built the reputation we bought, but it was certainly the culture that was there pre-VC money. You could basically say that AD sold, or lost, its soul. It doesn't really matter[39].

6. Is there a solution?

Many researchers and academics in the realm of computer science have contemplated escalation and the avoidance of failure or escalation when software is commissioned for a

[39] Johnathan Brink, quoted on http://michael.yoon.org/arsdigita, accessed March 14, 2006.

project, company, or private person. The short answer to all their research is that there is no “silver bullet”—that is, there is no perfect solution that software can create for a person. Previous business decisions have tried to simplify it into a “build vs. buy” decision, but often the solution can be a “build and buy” instead. Now more than ever, software always needs to be extended, imbedded, controlled, and modified to serve a specific purpose.

In the above cases, we saw a number of recurring themes. Requirements for a project were either lacking, or not on time. In the case of the VCF, the requirements were not in place when the contract was awarded to SAIC, and the requirement generation process was flawed. In the case of DIA, a number of ‘freezes’ were agreed upon, including a freeze to its software system requirements, which was then ignored and revised continuously.

The software was for a system that was billed as the first of its kind, or over-hyped in some way. CBHS was supposed to the largest baggage handling system ever built. The VCF was supposed to become the leading tool to fight terrorism for the FBI. ArsDigita’s Java product would lead to closed-source revenue that would save the company. In each project’s case, there was an element of elevated importance which placed supernatural expectations on the end product.

The management of the project was troubled, or changed significantly mid-project. In each project, there was a significant ‘sea change’ in the management, either through inability to come to grips with the organization’s culture (as with ArsDigita or the FBI) or through other reasons (the untimely passing of the manager at DIA, for example).

The root solution to avoid failure in the software creation for a project, then, is proper planning and management of the project at hand, starting with clear requirements gathering, a best practices testing methodology, and proper management of resources. The term 'resources' in this case does not only apply to number of servers, software products, or even people, but the business processes that create the need for the software and determine its overall place as a revenue-generating item in the first place. Additionally, a repeatable and communicable process should be in place so that the project can withstand change, either in the requirements or the management structure.

6.1. Is it just a problem with software?

Software has been singled out in business cases because there are many myths and legends surrounding software and its creation. The term itself, software, comes from the implication that software is easy to change, whereas hardware is more difficult to change[40]. These misconceptions led businesses into unforeseen pitfalls and created spectacular failures, like the examples we have spelled out above. There is no one reason why business leaders and managers make these assumptions when considering a software project; however, the following myths address some of the ideas that business leaders have when considering how to manage a software project.

Myth #1: Software does not have to be built like other products.

The reality is that there already exists a large body of work about how to build software, especially large products and complex systems, which has been developed over the last twenty years. Software has to be built through planning, requirements gathering and testing phases, just like any other product. It can be developed in a number of different

[40] http://en.wikipedia.org/wiki/CMMI, last accessed March 19, 2006.

styles, and there are many documented and existing patterns of development that software professionals practice on a daily basis. Two ways are the traditional 'waterfall' patterns and the 'iterative' or 'agile' pattern of software development.

In the 'waterfall' pattern, the software goes through a number of stages, but each stage is performed only once; planning takes place before development, development always takes place before execution, execution always takes place before testing, and testing takes place before releasing the final product. The problem that many professionals find with the waterfall system is that the initial requirements are inflexible, and that if there is a change over time, the pattern must usually be repeated from the beginning, creating delays and costing more money.

The 'iterative' approach differs from the waterfall in that there are smaller, cyclical stages repeated over time. That is to say, the software package is released in several releases, with each release building upon the last. Smaller milestones are described in each release, and feedback from users is strongly encouraged so that developers can meet expectations and create a better product for the end user. From this process, more realistic expectations are founded in the project and the developers on the project

Myth #2: New software systems can be found in buying off-the-shelf products, or for free in open-source products.

This is an incorrect assumption that leads many firms down the 'buy' path when faced with a build-or-buy decision. The fact is that many different projects require a number of man-hours to not only install but to configure with a business' already existing data. For Enterprise Resource Planning software, for example, it has been discovered that for every dollar spent on the purchase of a product, six to twelve dollars are spent on

implementation[41]. Open-source products can be even more costly to operate, since the product may come with fewer or nonexistent sources of support and documentation.

Myth #3: Software projects are easy to manage.

Managers have had this misconception that if a project is behind schedule, it is trivial to add additional manpower to make up time, usually measured in number of man-hours spent. Twenty years ago, Frederick Brooks wrote *The Mythical Man-Month*, which stated an axiom now known as Brooks's Law, "adding manpower to a late software project makes it later." This conclusion was presented after taking into account the learning curve of newly added workers, and the overall cost in time to distribute tasks and to communicate with new workers. In an anniversary edition of the book, he pointed to other academics' treatments which supported this theory, first by Abdel-Hamid and Madnick in 1991 and Stutzke in 1994.

In the sphere of international project management, it should be mentioned that there is a real need for developing an international framework for project classification and management is also a necessity; in a 2003 Global Survey of the International Project Management Association, the authors noted in the course of collecting data that "developing an agreed, world-wide system of classifying projects is urgently needed…for the discipline of project management." They also concluded that applying a 'one size fits all' model causes many project failures and that best practices should be identified for each type of project, so as to avoid inappropriate methods or systems.

[41] *Build vs. Buy*, 2001.

6.2. If it's not only software, what/who is to blame?

What the case studies have shown above is that, even though the failure occurred at the software level, there were other factors, both internal and external at work to exacerbate the situation and lead the management down the road to escalation. Holding to the aforementioned myths of software development is one thing that leads companies down a primrose path; belief in bad organizational patterns is another.

6.2.1. Software Management Processes

"Deficient software engineering is often blamed for project failures when the blame might more properly be placed with business managers who ignore lessons already learned by Software Engineers," reads one of the passages about software engineering in the website Wikipedia[42]. While there are many lessons that have been wrapped into usable systems of managing projects, each of them has their good and bad points. Three systems that manage the process of software creation are presented below.

6.2.2. Capability Maturity Model

The Capability Maturity Model (CMM) is one of the most established set of rules and methods for defining a software project. Originally designed for avionics software by the Software Engineering Institute at Carnegie Mellon University, CMM rates each software project on a scale of 1 to 5. This score is based on a number of different factors, and determines how an organization becomes more predictable about its software creation processes and quality as it moves up the scale[43]. Level 1 is considered for organizations that have an ad hoc method of program management with no real processes in place. Level 2 is already termed 'Repeatable', where software creation successes are easy to

[42] http://en.wikipedia.org/wiki/Software_engineering, accessed January 25, 2006.
[43] http://en.wikipedia.org/wiki/Capability_Maturity_Model, accessed January 25, 2006.

replicate. Level 5 is 'Optimizing', and considered the highest level, with continuous improvement of process performance. Many CIOs have used the level 5 as a benchmark to decide whether or not to do business with a vendor. "Level 5 was once a differentiator, but now it is a condition of getting into the game," one CIO from a large life insurance company said, "[Other levels] would be at a disadvantage."[44]

Now renamed as the CMMI (Capability Maturity Model Integration), the model is criticized often for its use in large, bureaucratic organizations, and inflexible for use in smaller nimble organizations. Since the CMMI rating is often used not only by large companies to grade vendors, but also used by the U.S. Government to award contracts, the pressure on auditors to deliver a high CMMI rating is very high. Some contractors will go to unreasonable lengths, either lying about their score or pretending that the entire company was rated at a certain CMMI level, when only one department was audited.

Despite the criticism, the CMMI has been instrumental in opening outsourcing markets in India, China, Ireland and Egypt, because companies are able to determine their standards for quality through a CMMI audit. It has also been important to firms in the U.S., to prioritize organizational improvement, and allow for internal review and implementation of process improvement.

6.2.3. Rational Unified Process

The Rational Unified Process (RUP) is an iterative process that was defined originally by the firm Rational Software, now a part of IBM. An effective process management system, RUP is a model much like CMMI in the fact that it is effective for large-scale projects and is extensible and flexible for different firms and organizations. The originators of

[44] Koch, 2004, accessed online March 19, 2006.

this system analyzed project failure themselves, looking at the root causes, and their end result was a system of best practices that made up the first draft of the RUP. Some of the things that they found to be serious problems included parts of the frameworks discussed in Section 2:

- Ad hoc requirements management
- Ambiguous Communication
- Brittle architecture, that is, architecture that does not perform well under stress
- Overwhelming complexity
- Undetected insufficiencies in requirements
- Insufficient testing

RUP is best characterized by a number of phases contained in a release. These are termed Inception, Elaboration, Construction, and Transition. Each iteration has deadlines, while each phase has certain objectives. RUP encourages other best practices which are specifically tied to specific technologies, including use of the Unified Modeling Language, object-oriented programming, and componentization of software.

As with each system, there are limitations to RUP; RUP is often thought to be expensive, and is linked with the purchase of a number of products on the market today, including the Rational Rose design tool. The fact that RUP is a product that can be purchased also implies that it could be used "straight out of the box" which is not the case; each organization should modify the process to suit their needs best[45].

6.2.4. eXtreme Progamming

The converse of a conventional and large-scale system like CMMI is XP, or eXtreme Programming, a style of project management which is also often called Agile Programming. XP was created about a decade ago to serve the need that two of our above case studies suffered from—changing scope of requirements later in the project

[45] http://en.wikipedia.org/wiki/Rational_Unified_Process, last accessed March 19, 2006.

cycle. XP characterizes itself as 'agile' and 'iterative' because it relies upon short, focused releases to create working parts of an application and then enforces frequent integration of different parts of the software into the main application. Other aspects of XP include *pair programming*, where developers would work together on a program, *test coverage*, where developers would write code to test for the actual code to be correct before writing the actual code itself, and *minimal documentation*, believing that the source code should serve as the primary source for a developers' knowledge about how the system works.

In a recent study, as an example of the effectiveness of XP, four Motorola software engineering teams wrote their projects using XP instead of the more traditional 'waterfall' methodology that is accepted practice. Each application was described as mission critical and expected to have a operating lifespan of 30 years, during which engineers would continuously upgrade and improve them. During the pilot program, Motorola engineers reported a rise in team morale, a shortening of the learning curve, an increase in engineer productivity, and excellent test coverage of the code overall. While they did note some challenges in moving to a new system (especially in terms of pair programming), they reported that "XP's project structure lets the team make rapid progress, even when facing volatile requirements"[46].

6.3. Remarks on the hypothesis and lessons learned

In the beginning, we hypothesized that not only software was a problem when analyzing failure and project escalation in software creation projects. We reviewed three cases where software creation was key, but the project organization experienced significant

[46] Drobka, Noftz, Raghu, 2004.

internal and external changes that affected the project overall, and led the project leaders into a condition of escalating resources to complete the project and avoid the stigma of failure.

In reviewing the cases and frameworks surrounding IT escalation, we discovered that escalation, in these specific cases, had nothing to do with the software, but that the processes and perception of the project itself were flawed. As further proof that escalation and de-escalation was independent of software, we have reviewed three different software development frameworks—CMMI, RUP and XP—and found that each framework depends not entirely on software also. CMMI and XP are entirely independent of any software, having to do more with process improvement and clear communication, and RUP, while associated with the Rational suite of software products, also contains milestones and phases which are process-based, and can be organized separate of the software itself.

6.4. Limitations and Future Possibilities

The author acknowledges that this is merely the tip of the iceberg as far as a study of IT project escalation is concerned; the subject is certainly important enough to warrant further investigation. The next steps from a researcher's perspective would be to conduct interviews with the actors involved in escalation and de-escalation of projects. Even Keil and Robey's study, where forty-two internal IT auditors were interviewed about their experiences, had its own limitations. They suggest that the internal auditors could possibly have introduced bias into the process by relating stories that were of most interest to them, and state that there is more work that could be done to review project de-

escalation, especially the process and timing of how managers try to turn projects around after bad news about the project has been sent and received.

Future research in this thesis can begin with interviews with the actors involved in project de-escalation, and encompass more case studies where projects did not end but instead continue after the project was deescalated.

7. Conclusions

The way to success is to double your error rate. –Thomas J. Watson, IBM

While arsdigita.com is up for sale, one website that is doing well today is OpenACS.org. Sporting the logo of a white Samoyed, this community software site—based upon the original Tcl source code of the ArsDigita product—is alive with activity, including discussion boards, downloads and sample websites created with the new product, now dubbed the 'Open Architecture Community System' instead of the ArsDigita Community System.

Now more than ever, failure is recognized as a necessary step towards success. DIA has gotten past its problems with the CBHS, and is currently building new runways to handle increases in traffic. The FBI, in its debacle with the VCF, gained a lot of necessary experience in working with software projects, and its next steps are seen as more credible, with solid expectations to have a working system. Many would-be webmasters and entrepreneurs have based their careers on open-source software thanks to the efforts of the OpenACS website, and the system powers websites for many well-known organizations (including the United Nations Industrial Development Organization, the Berklee College of Music, and Greenpeace)[47] as of 2006.

[47] http://openacs.org/community/sites/, last accessed March 17, 2006.

IT project escalation is one of the many frameworks that can be used to determine where projects have gone wrong in hindsight, but more important is the idea that this thesis introduces de-escalation, where managers can try to turn the project around and get a runaway project back under control. This thesis can be seen as an introduction to the challenges, concepts and tools to manage complex software and IT projects, and as a reminder that failure, with all its problems and negative connotations, also brings experience, and the humility to avoid the mistakes made the next time a project is started.

8. Bibliography

Ambler, Scott W. "Debunking eXtreme programming myths." *Computing Canada* 27(25), November 30, 2001, p. 13.

Ambler, Scott W. "The 'three Rs' of software development." *Computing Canada* 24(25), June 29, 1998, pp. 21, 23.

Andersson, Eve. "Diary of a Start-up," located online at http://www.assureconsulting.com/articles/arsdigitahis.shtml, last accessed December 13, 2005.

"Another headache at DIA." *ENR* August, 29, 1994, p. 8.

Archibald, Russel D. and Voropaev, Vladimir, "Project Categories and Life Cycle Models: Detailed Report on the 2003 IPMA Global Survey." 18th IPMA Project Management World Congress, Budapest, June 18-21 2004.

Arnoult, Sandra. "Back from the edge." *Air Transport World,* March 1, 2001. pp. 48-50.

Baiman, S. "Agency Research in Managerial Accounting: A Second Look." *Accounting, Organizations and Society* 15 (4), 1990, pp. 341-371.

"Be gone, foul fiends. " *Economist,* February 25, 1995, p. 31

Bozman, Jean S. "Denver Airport hits systems layover." *Computerworld*, May 16, 1994, p. 30.

Brockner, J. "The Escalation of Commitment to a Failing Course of Action. Toward Theoretical Progress." *Academy of Management Review* 17(1), 1992, pp. 39-61.

Brooks, Frederick P., Jr, *The Mythical Man-Month—Anniversary Edition.* Addison-Wesley, Reading, MA, 1995.

"Build vs. buy: A fresh look & other 'gotchas'." *Wall Street & Technology* 19 (10), October 2001, pp. 71-72.

Bulkeley, William M. "Venture Capitalists Invest $35 Million In ArsDigita Corp.," *Wall Street Journal*, March 29, 2000, p. B7

Bushrod, Lisa. "Corporate Venturing: Different models and their implications." *European Venture Capital and Private Equity Journal* 1, September 1, 2005, http://www.proquest.com/ (accessed March 17, 2006).

Charlier, Marj. "Denver plans backup baggage system for airport's troubled automated one." *Wall Street Journal,* August 5, 1994, p. B2.

Christopher, Alistair. "General Atlantic Embroiled in Portfolio Problems," *Venture Capital Journal,* June 1, 2001. pp. 5-6.

Conlon, D. E. and Garland, H. "The Role of Project Completion Information in Resource Allocation Decisions." *Academy of Management Journal* 36 (2), 1993, pp. 402-413.

"Denver Airport Investor Names City in Lawsuit. " *Wall Street Journal,* February 28, 1995, Eastern edition.

"Denver Airport Probe Ordered After Report Of Repeated Flaws. " *Wall Street Journal* January 8, 1996, Eastern edition.

"Denver International Airport Has Happened Before." *Material Handling Engineering*, August 1, 1994, p. 7.

"Denver official ends task force probing building of airport." *Wall Street Journal,* May 17, 1996, Eastern edition: A9D.

Dichter, Carl. "Is it Designed Right?" *UNIX Review* 11(6), June 1993, pp. 51-58.

Dubie, Denise. "Never-fail business services." *Network World* 19(38), September 23, 2002. pp. 56-58.

Erdogmus, Hakan, and Williams, Laurie. "The Economics of Software Development by Pair Programmers". *The Engineering Economist* 48(4), 2003. pp. 283-320.

Ettlie, John E. "The seven myths of new product development." *Automotive Manufacturing & Production* 109(6), June 1997. pp. 18-20.

Flynn, Kevin. "Denver Airport Probe Dropped." *ENR*, May 27, 1996, p. 17.

Flynn, Kevin, "Denver's bag system finally nears finish." *ENR*, November 13, 1995, p. 15.

Fox, David. "In bed with a venture capitalist." *Intheblack*, October 1, 2005, pp. 32-35. http://www.proquest.com/ (accessed March 17, 2006).

"GAO Finds No Reason To Suggest Airport Won't Pay Its Bills." *Wall Street Journal* February 16, 1996, Eastern edition.

"GAO Issues Outlook On Financial Health Of Denver Airport. " *Wall Street Journal*, October 17, 1994, Eastern edition.

Gallagher, Kevin M. "Love thine enemy." *Software Magazine* 18(13), October 1998, pp. 62-65

Gittlen, Sandra. "It's a dog's life at Internet start-up ArsDigita," *Network World* 16(26), June 28, 1999, p. 44.

Goldstein, Harry. "Who killed the Virtual Case File?" *IEEE Spectrum* 42(6), September 2005, pp. 24-35.

Gowigati, Benoit, and Grenier, Benoit. "The winds of change: change management takes off at Bombardier." *CMA Management*, 75(8) November 2001, pp. 34-38.

Greenspun, Philip. "ArsDigita: From Start-up to Bust-up," located online at http://www.waxy.org/random/arsdigita/, last accessed December 13, 2005.

Hall, Mark. "Banking on open-source ASPs," *Computerworld* 35(12), March 19, 2001, p. 38.

Havelka, Douglas, "A user-oriented model of factors that affect information requirements determination process quality." *Information Resources Management Journal* 16(4), Oct-Dec 2003, pp. 15-32.

Hayes, Frank, "FBI on the Move." *Computerworld* 39(22), May 30, 2005, p. 46.

Henkoff, Ronald. "Smartest & dumbest managerial moves of 1994." *Fortune*, January 16, 1995, p. 84.

Jensen, M. C. and Meckling, W. H. "Theory of the Firm: Managerial Behavior, Agency Costs, and Ownership Structure." In *Organizational Economics*, J. B. Barney and W. G. Ouchi (eds.), Jossey-Bass, San Francisco, 1986, pp. 214-275.

Keil, Mark, Mann, Joan, and Rai, Arun. "Why software projects escalate: An empirical analysis and test of four theories." *MIS Quarterly* 24(4), December 2000, pp. 631-664.

Kiel, Mark and Robey, D. "Turning Around Troubled Software Projects: An Exploratory Study of the Deescalation of Commitment to Failing Courses of Action." *Journal of Management Information Systems* 15 (4), 1999, pp. 63-87.

Khalifa, Mohamed and Verner, June M. "Drivers for software development method usage." *IEEE Transactions on Engineering Management* 47(3), August 2000, pp. 360-369.

Knill, Bernie. "New baggage handling solution ... sort of." *Material Handling Engineering*, September 1, 1994, p. 43.

Koch, Christopher. "Bursting the CMM Hype," *CIO*, http://www.cio.com/archive/030104/cmm.html, last accessed January 25, 2006.

Kontzer, Tony. "Under Pressure." *InformationWeek* 1021, January 10, 2005, pp. 18-19.

Kopel, Dave. "Pena's new airport still a failure." *La Voz*, February 26, 1997, p. 5.

Lane, Randall. "Thanks, Secretary Pena." *Forbes*, April 10 1995, p. 22.

Mahring, Magnus, Holmstrom, Jonny, Keil, Mark and Montealegre, Ramiro. "Trojan actor-networks and swift translation: Bringing actor-network theory to IT project escalation studies. " *Information Technology & People* 17(2), 2004, pp. 210-238.

McCartney, Scott. "Denver Airport Baggage System Is Canceled by United Airlines." *Wall Street Journal* June 7 2005, Eastern edition, p. D.5.

Mellahi, Kamel and Wilkinson, Adrian. "Organizational failure: a critique of recent research and a proposed integrative framework." *International Journal of Management Reviews* 5-6 (1), 2004, pp. 21-41.

Montealegre, Ramiro and Keil, Mark. "De-escalating information technology projects: Lessons from the Denver International Airport." *MIS Quarterly* 24(3), 2000, pp. 417-447.

Neumann, Peter Gabriel. "On Hierarchical Design of Computer Systems for Critical Applications." *IEEE Transactions on Software Engineering* 12(9), September 1986, pp. 905-921.

O'Brian, Bridget, and Charlier, Marj. "Travel: Denver's new airport, already mired in controversy, won't open next week." *Wall Street Journal* March 2, 1994, p. B1.

Ortega, Bob. "Travel: More good reasons not to fly into Denver." *Wall Street Journal* March 31, 1995, Eastern edition, p. B1.

Orli, Richard J. "Battling Project Escalation." *Computerworld* 23 (14), April 3, 1989, pg. 64.

Parker, Dana J. "Chronicling the myth of the killer app." *E Media Professional* 11(7), July 1998, p. 62.

"Pena admits DIA mistakes." *ENR*, May 22 1995, p. 19.

Pena, Federico. "It would've been folly not to build the DIA." *Wall Street Journal*, March 8, 1995, p. A21.

Plyler, Robert W, and Kim, Young-Gul. "Methodology myths: Four tenets for systems developers." *Information Systems Management* 10(2), Spring 1993, pp. 39-45.

Ragavan, Chitra, "A Q&A with the FBI's data czar." *U.S. News and World Report*, June 9, 2005. Last accessed June 26, 2005 at http://www.usnews.com/usnews/news/articles/050609/9azmi.htm

Rifkin, Glenn. "What really happened at Denver's airport." *Forbes*, August 29, 1994, p. 110.

Rubin, J. Z. and Brockner, J. "Factors Affecting Entrapment in Waiting Situations: The Rosencrantz and Guildenstern Effect." *Journal of Personality and Social Psychiatry* 31, 1975, pp. 1054-1063.

Ruggless, Ron. "Denver airport: From baggage-eating turkey to concession eagle." *Nation's Restaurant News* March 13 2000, p. 6.

Scott, William B. "Denver Defies Critics: Eight years after its debut, DIA has become a benchmark for modern airports." *Aviation Week & Space Technology*, June 23, 2003, p. 51.

Shaheen v. Greenspun, Public Version of Plaintiff's pre-trial brief, online at http://unicast.org/stuff/arsdigita/043.pdf, last accessed December 13, 2005.

Shaheen v. Greenspun, Defendants' Opening Brief in Support of Their Motion for Partial Judgment on the Pleadings, online at http://unicast.org/stuff/arsdigita/021.pdf, last accessed December 13, 2005.

Shenkland, Stephen. "Open-source firm reverses strategy", originally published at CNet.com, on April 5, 2001. Available online at http://news.com.com/2100-1001-255418.html, last accessed December 13, 2005.

Simonson, Shai. "A post-baccalaureate undergraduate-level program in computer science," *Association for Computing Machinery. Communications of the ACM* 45(7), July 2002, pp. 21-24.

Skok, Walter, and Scarre, Gina. "Computer Project Management: An Information Systems Development Methodology as the Critical Success Factor - Myth or Universal Truth?" *Management Research News* 15(2), 1992, pp. 6-14.

Sommerville, Ian, "Software process models." *ACM Computing Surveys* 28(1), March 1996, pp. 269-271.

Spolsky, Joel, *Joel on Software: And on Diverse and Occasionally Related Matters That Will Prove of Interest to Software Developers, Designers, and Managers, and to Those Who, Whether by Good Fortune or Ill Luck, Work with Them in Some Capacity.* Apress, Berkeley, CA, 2004.

Staw, B.M. and Fox, F. V. "Escalation: The Determinants of Commitment to a Chosen Course of Action." *Human Relations* 30 (5), 1977, pp. 431-450.

Szyliowicz, Joseph S, and Goetz, Andrew R. "Getting realistic about megaproject planning: The case of the new Denver International Airport." *Policy Sciences* 28(4), 1995, p. 347.

U.S. Congress, House, Committee on Appropriations, Surveys and Investigations Staff. *A Report to the Committee on Appropriations, U.S. House of Representatives on the Federal Bureau of Investigation's Virtual Case File.* April 25th, 2005.

"U.S. is probing allegations about new Denver airport." *Wall Street Journal* August 26, 1994, Eastern edition, A12.

"UAL Corp.: Denver Forgives Airline's Debt From Failed Baggage System." *Wall Street Journal* December 14, 2005, Eastern edition, p. B.2.

"United Parcel Service: Plans for a regional hub in Denver are postponed." *Wall Street Journal* April 5, 1996, Eastern edition.

Weiss, Edmund H. "Why We Still Have So Little Technical Documentation." *Infosystems* 30(5), May 1983, pp. 88-89.

Xue, Yajiong, Liang, H., Boulton, W, and Snyder, C. A. "ERP Implementation Failures in China: Case Studies with implications for ERP vendors.", *Production Economics* 97, 2005, pp. 279-295.

Whyte, G. "Escalating Commitment to a Course of Action: A Reinterpretation." *Academy of Management Review* 11 (2), 1986, pp. 311-321.

Yin, R. K. *Case Study research: Design and methods, 2nd Edition*. Sage Publications, Thousand Oaks, CA, 2004.

Yoon, Michael. "ArsDigita: an Alternate Perspective," located online at http://michael.yoon.org/arsdigita, last accessed December 13, 2005.