MANAGING AGILE E-BUSINESS DEVELOPMENT: HOW INTEL AND MICROSOFT DID IT?

Dien D. Phan
Department of Computer Information Systems
St. Cloud State University
St. Cloud, MN 56301, U.S.A.
ddphan@stcloudstate.edu
320-308-3890

Abstract: Electronic Business (e-business) plays a major role in the world economic growth today. However, management of e-business system development projects has not been easy. Failure to meet requirements, high development costs, unscheduled server shutdowns, and schedule delays have been common. Since the emergence of e-commerce, systems development paradigms have shifted. Many new paradigms have worked well while others have not.

The goals of this study are to gain insights concerning the in-house development of early e-business systems at Intel Corp. in 1997 and an online retailing system at Fingerhut by teams at Microsoft. Lessons learned and critical success factors are discussed.

INTRODUCTION

The history of software development reveals a multitude of problems caused by poor, as well as undisciplined and incomplete development practices. A study by The Standish Group in 1995 reported that the overall success rate of information technology projects in the U.S. was only 16.2 percent and that over 31 percent of IT projects were canceled before completion (Standish Group, 1995). Ten years later, the 10th edition of the annual CHAOS report from the Standish Group reported the project success rates at 34 percent, a hundred percent increase Software Magazine (2007).

The chief reasons for the improved project success rate "…is the projects have gotten a lot smaller. Doing projects with iterative processing as opposed to the waterfall method, which called for all project requirements to be defined up front, is a major step forward." said Standish Chairman Jim Johnson (Ibid.).

Intel Corp. is one of the few major brick-and-mortar companies that developed an e-business system in-house quickly during the early days of e-commerce. In June 1997, Intel studied the feasibility of moving into Web-based e-business. The development began in early 1998, and by July 1998, a pilot system was deployed. The project started with dozens of highly competent developers then expanded to involve several hundreds of employees and customers world-wide. By the end of 1998, the entire e-business system at Intel was completed on time, within schedule, and within budget.

The purpose of this study is to gain insights regarding success factors that contribute to the successful delivery of agile e-business projects. The goals of this study are to determine:

- What are the major issues faced by agile e-business development projects?
- What are the new insights in e-business software development? What has been changed from traditional systems development?
- What are the success factors in agile e-business development?

E-BUSINESS SYSTEMS DEVELOPMENT

To survive and grow, business enterprises frequently redesign their business by building new information technology capabilities. For companies that need to exploit opportunities in e-Business, building software to integrate Internet technology to their business processes is crucial.

Modern Information Systems Project Management

Although the impact of software on the world's business is enormous, software development processes today suffer a multitude of problems. Schedule delays, resource mismanagement, cost overruns, high maintenance costs, security failure, and poor quality have been very common (Phan, Vogel, and Nunamaker J. F. Jr., , 1995). Despite continuous improvements in software engineering and ongoing efforts made to enhance the processes and techniques used in the management of projects at all stages, today's IT development projects, most of which are e-commerce systems, still are not properly managed.

Numerous software engineering and project management studies suggest that the major challenge for any development project is the ability to manage and control development resources and environment (Brooks, 1995; Phan, Vogel, and Nunamaker, 1988,; 1995). With the emergence of new business models such as supply chain management (SCM) and the widespread use of off-the-shelf software, controlling dependence on external resources in the development of e-business software is a major challenge.

There are several reasons for this challenge to exist. First, it is difficult to control and manage end-to-end connectivity worldwide, as much of it falls outside direct control of company's management. Second, managing dependence on vendor software components is also a daunting task. Panko (2003) reports that many companies that use Microsoft software are having difficulties in keeping up with numerous and frequent upgrades, patches, and service packs. Many of them have not been fully tested and thus contain bugs that crash their systems. Third, some experts believe that building software is similar to "craftsmen building cathedrals during the medieval time" - one stone at a time with no interchangeable components- thus making software more difficult to build than modern airplanes or bridges WGBH Television (1991). Fourth, the quality of service from vendors for off-the-shelf software components is poor. Due to competition and the need to increase market share, software makers have slashed the price of software and in many instances have allowed free downloads to registered users. The recent downturn of the software industry has caused many software vendors to reduce or outsource off-shores their support staff. They argued that the low prices or free downloads don't allow them to provide adequate support. Consumer Union (2003) reported that 60 percent of customers who call vendors' technical support numbers encounter problems with busy signals, annoying telephone menus, unreasonable waiting times, and technicians who seem not to know what they are talking about. For customers whose only choice is to use online support, the situation is not better. Hard to find solutions, buggy updates and patches, slow websites (as millions of users download the same update to stop the new worm), and slow email responses from technical supports are commonplace.

As requirements for e-business systems exceed the resource capacity of development projects, managing projects to meet schedules, costs, and requirements becomes more difficult. The typical software engineering phenomena of using limited resources to meet all requirements has been described by Brooks in his well-known book, *The Mythical Man-month* (Brooks, 1995). Compressed schedules to cut development cycles also create extra demand on limited resources.

In responses to the need for short development cycle time, many organizations such as Microsoft and Netscape began to implement some of the agile development processes in the 1980s (Cusumano and Yoffie, 1998). However, not until 2001 that Ken Beck et al. announced the formal agile method which combines a philosophy and a set of development guidelines on their Agile Software Development Manifesto web site (http://agilemanifesto.org/). The philosophy of agile development approach is to have incremental delivery, informal methods, simplicity, and small teams. The development guidelines stress delivery over analysis modeling and require continuous user involvement. Furthermore, since highly competent developers can be up to ten times more productive than less competent ones (Brooks, Ibid.), highly skilled workers are vital in the success of agile development projects.

Pfeffer and Salancik (1978) and Phan et al. (1995) argue that limited resources cause projects to become interdependent with other organizational units, as well as with the external environment, for resources that they do not currently have. As a consequence, projects and the environment make more demands on each other and develop internal and external strategies to minimize uncertainties arising from dependence on the environment for resources. These strategies include buffering, coordinating, contracting, and merging.

When controlling the external environment is costly or not possible, development projects may seek informal and semi-informal interdepartmental linkages to coordinate organizational units via the supply chain management (SCM). Because E-Commerce systems require significant investments in infrastructure and cause changes in the supply chains, development projects tend to use low risk incremental approaches such as pilot systems or prototypes (Turban, Lee, Warkentin, and Chung, 2002)

Furthermore, economic counter-forces are working against building quality in a vendor's software components. Given competitive market places in the software industry, software products that offer reliability are often edged out by products that offer more features. Worse still, the endless treadmills of software releases often cause tight coupling among versions of software products. This phenomenon is called "version skew" in which product A depends on version X of product B but then when B gets upgraded to the next version, product A fails (Lieberman, and Fry, 2001).

Numerous studies have been conducted to study successes and failures of software development projects. Key project success factors are (Phan et al., 2005, Pinto, and Slevin, 1989):

1. Clear goals, directions, and well-defined scope.
2. Top management support.
3. Client/User participation.
4. Good project management and quality control standards.
5. Multiple versions available.
6. User/Client acceptance.
7. Coordination and collaboration efforts..
8. Favorable project publicity.
9. Optimal mix of work force.
10. Detailed plan and schedule
11. Good management of features and milestones

CMMI AND SOFTWARE DEVELOPMENT PROCESS IMPROVEMENT

In order to measure the effectiveness and maturity of organization's software development capabilities, The Carnegie Mellon University's Software Engineering Institute (SEI) has developed the Capability Maturity Model (CMM) that is predicated on a set of software engineering capabilities with focus on process quality management (Paulk et al., 1993). Today the CMM model is enhanced by the Capability Maturity Model Integration (CMMI) which incorporates software engineering, system engineering, and program management. It also includes measurements in project management skills that are crucial to project managers.

Many studies have found that improving process capability and maturity lead to better quality and productivity (Fox, and Frakes, 1997; McConnell, 1993). Certainly moving up the capability and maturity level in the model requires significant investment, efforts, and discipline, these studies showed that moving up in the CMM level is a good investment.

E-business Systems Development

Numerous books and articles have suggested methods and practices for building e-commerce systems. Turban, Lee, Warkentin, and Chung (2002) recommend the following steps:

1. Define Architecture
2. Select Development Option (Buy, Lease, In-house, Outsourcing, or Hybrid)
3. Install
4. Deploy System
5. Operate and Maintain

Major issues in building E-commerce Systems are:

1. *Integration of Front, Back Office, and Supply Chain systems.* On-line applications must be connected and integrated to back office systems, such as ERP and DBMS. Good integration is critical to make sure that customer orders match existing inventory, price, and delivery schedule. The failure of Fingerhut, Inc. in 2000 highlighted the importance of system integration.

2. *Vendor and Software Selection.* Few companies today have the capability and resources to develop an entire e-business system. The majority rely on vendors to supply hardware, software, and expertise to build e-business systems. Companies can select to have servers run on mainframes, mid-range computers, Unix or PC workstations. While mainframes and mid-range computers require more staff to support, they are not vulnerable to the majority of attacks that aimed at Unix or PC workstations. The belief that PC servers require fewer staff to run than those that use mainframe or midrange computers may be false. As mentioned previously, many IT departments lack human resources to install and thoroughly test frequent and numerous patches and updates for PC servers.

3. *Software customization.* Off-the-shelf e-business packages generally offer enterprises to customize the functionality of the software by developing their own code in Java, C++, HTML, XML, ASP, etc. Dependent on the individual situation, each company must decide to build e-business system proprietary or open source software. For example, during the early days of e-commerce when open software was not popular, many organizations, such as CheckFree.com, used proprietary technology for its software and built custom linkages between the front office software and back-end legacy systems. However, their competitors who entered the market later used XML-based solutions built on open Web standards. The open systems can be purchased for much lower costs and are easier to integrate into an organization's legacy systems than proprietary software.

4. *Web Security.* With the increasing number of hacking activities on e-commerce systems, maintaining high quality web systems is a daunting task. In 2002, CERT (www.cert.org) reported over 2,000 vulnerabilities on e-commerce systems, and the annual number of vulnerabilities doubling in recent years. The recent discovery of various security problems in Microsoft's software systems has shown that the security threat to e-business systems remains serious, and more latent security bugs in Windows systems may emerge. While firewalls and routers can protect the systems most of the time, they too are not invincible. For example, on January 26, 2003, router maker Cisco acknowledged that some of its routers may have been taken over by the infamous Slammer worm (Panko, 2003).

The differences between traditional development projects and e-business projects are summarized in Table 1.

Table 1: Common Differences Between Traditional and e-Projects (Kukik and Samuelsen (2000).

Development Process	Traditional Projects	E- Projects	
		Small	Large
Requirements	Rigorous	Limited	Rigorous
Technical Specifications	Robust: Models	Overview	Robust: UML models
Duration	Months or years	Days, weeks, or months	Months or years
Quality and testing	Quality targets focus	Risk control focus	Quality Assurance focus
Release Process	Rigorous	Expedited	Rigorous
Feedback	User involvement focus	Automated feedback	Automated and solicited feedback

In the following sections, the e-business development project is examined to further understanding the challenges and strategies used in managing an agile development project.

E-COMMERCE DEVELOPMENT AT INTEL

Intel Corp., located in Santa Clara, California, is the world's largest producer of Integrated Circuits Chips in the world today with a total revenue for 2002 of $26.8 billion and a net income of $3.1 billion. Founded in 1968 to build semiconductor chips, Intel invented the microprocessor more than 25 years ago. Today, the company has evolved from a processor manufacturer into a supplier of network and server hardware, Internet hosting services, and other e-Business components. Its technological leadership ranges from microprocessor design to advanced manufacturing and packaging, and it maintains production and research facilities around the world. Thanks to its generous salaries, benefits, and working conditions, Intel was able to hire and maintain the best employees in the Silicon Valley.

By late 1990s, the benefits of e-business for Intel were too great to be ignored. With a world-wide network of business partners, resellers, and original equipment manufacturers (OEM), Intel needed to improve its efficiency by automating its business to business (B2B) processes. Current traditional order processes at Intel were too slow and thus a Web-based order management system was needed.

Most of Intel's business is in the PC market where there is intense competition from other chip makers such as Advanced Micro Devices (AMD), Cyrix (acquired by National Semiconductor in 1997 and then by VIA in 1999), Texas Instruments, Motorola, and IBM. In the past Intel customized its catalogs and sent them to its potential customers along with product availability information. Until the summer of 1998, Intel's business process was done entirely on paper. However, when large customers such as Dell Computers and Cisco Systems started to use the Web to do their business in 1996, they pressured Intel to do business online. Andy Grove, chairman of Intel in 1997 expressed his strong belief in the e-business model and wanted Intel to move quickly into Internet

business (Intel, 2000). The mandate to launch an e-business system within several months sent shockwaves to the newly formed e-business development team: *"A lot of people feel overwhelmed by 'The Task' "* noted by Sandra Morris, Vice President of Sales and Marketing Group, and director of Internet Marketing and E-commerce at Intel (Intel, Ibid).

Defining Goals and Customer Requirements

In order to establish the project scope, Intel's e-business customer requirements and market opportunities were studied. At the end of 1997, Intel began building its worldwide e-business team and conducting a customer needs analysis. As the analysis moved forward, it became clear that the group closest to the customer should drive the project; therefore, Intel's Sales & Marketing Group was assigned to lead the development. The project began with a systems requirements study which included business partners requests, executive meetings, and literature searches. The results were translated into project requirements which were then ranked by their relative importance, and the required resources to develop key requirements were estimated.

Project Resource and Environment Control Strategies

With the mandate to install an e-business system from the top, Intel development teams were able to obtain approvals for the analysis and design stages quickly. With top management commitment and strong supports from all stakeholders, development groups obtained the best developers in the corporation for the project. Realizing that building entire e-business systems from the ground up is a daunting task, the initial project team decided to go ahead with a pilot project to design and deploy a simple e-business solution that worked with existing business processes, and then grow the system from one version to the next. Within a few weeks of brainstorming, the scope and the initial architectural design were approved and dozens of highly competent developers quickly moved into the development of a pilot system.

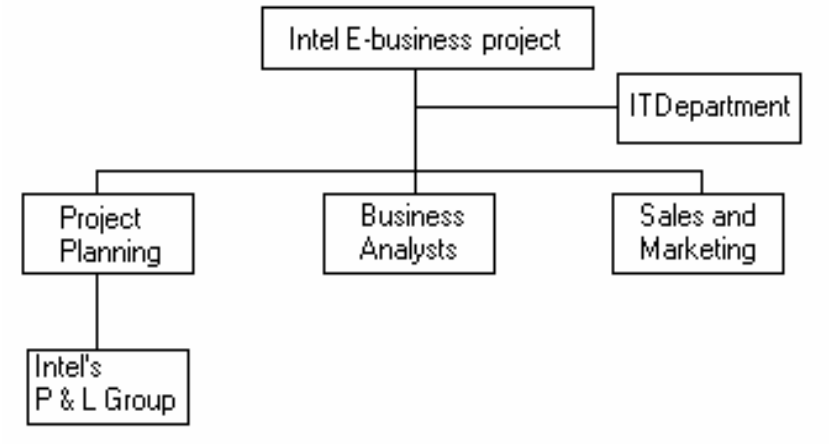

Figure 1: Intel E-Business Initial Project Organization Structure (Than, 2000).

The roles and responsibilities of participants of the initial e-business development teams at Intel (Figure 1) were defined as follows:

1. A planning team which consisted of customer, technical and logistical perspectives to define the scope and objective of the project.
2. Business analysts and consultants to help define the business workflow and assess how information is given to customers.
3. Sales and marketing staff to determine the best ways to work with customers in the new e-business environment.
4. Intel's planning and logistics group to help the IT department build a solution that integrated with existing business systems.
5. The Information Technology (IT) department to act as an "enabler" of business. Its role was to integrate e-business technologies and build the system.

Avoiding Common Software Engineering Traps for Failures

Because there were few people with adequate knowledge and experience in the development of B2B E-commerce systems, the initial Intel development team was extremely cautious. It must avoid common causes of software engineering project management failures,

such as lack of communications, setting ambitious goals that they could not meet, and mismanagement of schedule slippage. Rather than attempt to build the entire business infrastructure from ground zero, Intel development group agreed that an agile approach to build the pilot system using small teams was the best solution. The philosophy was to grow the systems at incremental deliveries. The project scope then was narrowed to a pilot system that served its middle tier customers which covered 25% accounts in 35 countries (Figure 2). *"We picked one thing we could build very quickly and deploy to our customers. The thing to do is ask what makes you a strong company and how do you use Internet to make the company stronger,"* noted by Sandra Morris (Intel, Ibid.).

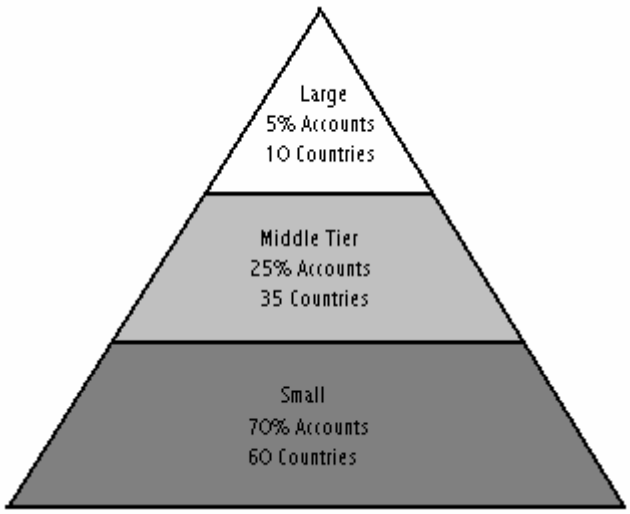

Figure 2: Intel's account structure at the time of development (Than, Ibid.)

E-Business Systems Architecture

In order to simplify the e-business system maintenance and support, Intel decided to standardize the e-business architecture to one hardware vendor, one operating system, and minimized the number of database and application vendors. The infrastructure was designed to be flexible and scaleable to help manage the cost of ownership as the system grew.

Servers. Intel's initial e-business system infrastructure was built around three main clusters of Internet servers: web-servers, database servers, and data analysis servers. The reasons for standardizing on one hardware vendor were to simplify maintenance costs, make growth easier, and allow Intel to interchange components as necessary without compatibility issues. Because Intel made CPUs for Personal Computers, it has a strong interest in the use of PCs servers in e-commerce systems.

OS and Databases. Intel's e-business system was standardized on one operating system from Microsoft. DBMS were limited to two vendors.

Application Development. Intel provided a number of applications that served specific needs and made a great effort to use off-the-shelf components. Rather than using proprietary software to develop e-business systems, Intel was one of the few companies who selected common or open system standards such as SSL encryption and Extensive Markup Language (XML) to build its e-business system in the early days of e-commerce. In order to move further toward the widely-accepted open Web standards, Intel joined RosettaNet to develop common XML schemas and definitions to enable automated supply chain interactions within the industry.

Incremental Development Approach and Schedule Management

To build the e-business system, Intel used an incremental approach to migrate business processes to the web with frequent integration testing. Small milestones were set for all teams to meet weekly. This approach is similar to the common "grow rather than build from scratch" and "daily synchronization and stabilization" approaches that leading software makers, such as Microsoft and Netscape, are using [4]. One of the major milestones at Intel after the launch of its e-business website was an *Access Manager* application that automated the creation of account user IDs and passwords to access the unified environment known as the "Landing Zone." In the past, an Intel account supervisor often spent over an hour creating user-ids and passwords for customer, since each user-id was only

good for one application. With the Landing Zone environment, Intel customers could log into one place and access many other intra/extranet sites; this convenience attracted many users. *"The people want to come to the environment because it offers these common services of security and entitlement…"* commented the B2B Platform and e-Content Application manager of Intel's Sales and Marketing Application Group. Other milestones included the 128-bit encryption function that started with the domestic encryption technology, and then extended to include third party encryption applications that were developed outside the U.S. The only problem that caused the delay of the project for many weeks was the high packet loss which we will discussed later. Thanks to early detection, the delay did not cause significant project slippage.

Project Communications and Customer Involvement

In order to communicate project scope and requirements to project stakeholders, initial project communication documents such as project charter, scope, schedule, milestones, status reports, and problems tracking were stored in the project repository, many were available on intranet. For customers overseas had little or no knowledge of e-commerce and had no e-commerce infrastructure, Intel initially made a great effort to educate customers in basic e-commerce and Internet applications. Information on individual customer Internet and network structure, e-commerce background, and customer supply chain and workflow management was collected so that the new e-commerce systems would be able to fit into existing customer information systems.

Once Intel had a grasp of the overall Internet infrastructure and e-commerce background of its customers, development team members contacted customers and performed assessments on their e-commerce system needs.

To gain early support from customers and employees, Intel launched a publicity campaign about its e-business projects and the benefits of e-commerce. During the period of 1998-2000 many Intel employees and executives played the project champion and/or cheer leader roles to business customers and other employees world wide. For example, Mr. Phuc Than, Geeneral Manager of Intel Vietnam, frequently promoted Intel's e-business systems to Intel's customers in Vietnam.

International input proved to be crucial. For example, through customer early involvement in testing connections, the development team detected a connection failure from an overseas customer. This was traced to the inability to decrypt the SSL which used the domestic 128 bit encryption scheme since it was not available for customers outside the U.S.A.. In 1998 the U.S. government still banned export of the U.S. 128 bit encryption. The Intel team fixed this problem by allowing overseas customers to use encryption applications developed by vendors outside the U.S.. *"It is absolutely amazing the amount of work we did with customers oversea to make e-business program a success for them, I remember one time our technical team traced a connectivity issue back from a Customer in Asia, through their ISP, and out across the Net to a bad router board in San Francisco owned by yet another ISP who hadn't realized they had a problem,*" noted an e-Business Customer Marketing Manager at Intel (Intel, Ibid.).

Other customer involvement activities included:

1. Formation of a team specially focused on customer deployment,
2. Running concurrent project tasks so that deployment related activities such as account ids and passwords took place while the system was under development,
3. Made the pilot intranet website available to beta account users,
4. Education, training, and project promotion for customers and users world wide,
5. Upgrading project priority by positioning e-business as strategic imperative for Intel and its customers,
6. Ensure responsive Internet access by working with customers, ISP, and other related organizations in customer site countries.

Through working with customers to install network connections in early 1988, Intel teams were surprised to find that some international customers had no Internet capability and thus were not ready to do business online. For example, one Intel partner actually had to create a new budget to pay for the cost of Internet access. Another connection problem that the Intel development team solved for its customers was the high rate of Internet packet loss during transmission. *"We worked with a deeply committed team of world wide contributors. The inputs on design, deployment, and training came from people who were in those countries around the world. We like to say that the sun never sets on our e-business development, because we have people working on it around the world in every time zone*", noted an Intel Manager.

E-Business Pilot System Delivery

The initial e-business pilot system was one-stop with 240 shopping sites for customers throughout the world. Personalized data and applications were tailored to users' needs to provide an individualized experience. This system basically was a self-service extranet named "E-Business Program" focusing on procurement and customer support for its supply chain partners. Access to the site was restricted to Intel's authorized business partners and customers. Through iterative development, this system eventually grew into a full scale B2B e-business system in 1999.

Deployment Problems and Challenges

For transaction security, Intel sites let customers place and track orders using standard Web browsers with Secure Socket Layer (SSL) using the 128-bit encryption. Since the U.S. 128 bit technology was banned from export by U.S. law, Intel's alternative was to make an exception for business partners overseas to allow the use of the weaker Data Encryption Standard (DES) which uses the 56 bit encryption. However, this solution would compromise Intel's web security. System security strength is measured at the weakest link and thus hackers could easily break the system at the weakest link from overseas. To maintain the strength of the128 bit encryption technology world wide, the solution was to allow customers to acquire third party 128-bit encryption application that was developed outside the U.S. to connect with Intel's B2B system. The decision to use encryption application from third party vendors was a departure from the traditional belief that software components must be "crafted" individually and cannot be interchangeable.

Another challenge was to overcome the high packet loss problem. To increase network throughput, data compressions were used. However, the encrypted file transfers using SSL are very sensitive to packet loss. Early efforts in network problem detection for customers discovered that when the packet loss rate exceeded 15%, the download times for encrypted pages skyrocketed. Intel technicians were surprised that this problem had never been detected by the Network Service Providers (NSP) or Internet Service Providers (ISP). So Intel engineers were left to solve the problem for customers. Through working with customers, Intel also noticed that most of the transmission problems were located in connections between customer sites to Central offices of the telephone companies or local network hubs. The solutions were to reduce the number of elements that needed to be transferred through the network and to upgrade the communications bandwidth. This was done by redesigning the web pages for storing reusable artifacts, such as graphics, menus, and icons, locally at customer sites so that only compressed data would be transferred through the net. With some incentives from Intel, many overseas partners upgraded their network connections to ISP servers with high bandwidth pipes and high speed network and routers.

Quality Assurance

During development, codes were frequently inspected and tested. Intel's development groups anticipated frequent changes in both internal and external design and processes. Its development process was an iterative one that relied on continuous improvement and frequent synchronizations. Customers played a major role in quality control via the on-line feedback and Intel's site visit. During the development process, Intel continued to put beta versions of new features out for customers to use and test, then continued to upgrade its e-business system based on the feedback. This is the modern agile software development approach. During the early days of agile development approach, critics of this process argue that due to a short development cycle, "release-often" software does not go through the rigors of the traditional development process. The counter quality assurance argument for this approach is the exposure of Beta versions to thousands of users worldwide, allowing bugs to be identified faster and fixed sooner with fewer testers.

For a company that markets products directly to business customers worldwide, high quality connection was a big challenge for Intel, especially in parts of Europe and Asia. This was because Intel has no control of the end-to-end connections. Intel frequently tested its connections with customers to keep up with rapid increases in sales volume and communication throughput. Because network performance varies significantly with customers in different countries, Intel constantly motivated and pressured its international customers bring their connections up to the acceptable standards.

RESULTS

On July 1, 1998, Intel officially began taking orders from OEM and distribution customers using a new series of personalized Web sites. The new e-business system enabled approximately 200 Intel's customers in almost thirty countries to place orders for Intel products, check product availability and inventory status, receive marketing and sales information, and obtain customer support - all in real time, 24 hours a day, 7 days a week. The major successes of this project are summarized below (Phan, 2001b):

1. Wave after wave of orders, worth $1 Billion, were booked on its online e-business system in the first 15 days of deployment, surpassing the company's initial launch goal of moving $1 Billion in the first three months.
2. The company was able to eliminate most faxes to and from its customers world wide. For Taiwan partners alone, it eliminated 45,000 faxes per quarter.
3. The company continued to receive approximate $1 Billion of online orders per month for the rest of 1998.
4. Customer survey rated Intel's e-business site at 94% for Overall Value.
5. By June, 1999, over 560 companies in 46 countries were using Intel's e-business system.
6. The time to complete a transaction was reduced by an average of 70%.
7. Customers were able to conduct business with Intel without going back to faxes and telephones.
8. Many of Intel's employees who participated in the development of the e-business program received promotions after its successful deployment. Sandra Morris was promoted to the CIO position.

Entry to Wireless

After the success of the e-business program, Intel expanded its e-business system further by moving into Mobile-commerce. By the end of 2000, close to eighty percent of Intel's computer-using employees moved into wireless technology. Intel upgraded many conference rooms and venues such as the cafeteria with RJ-45 jacks, enabling employees to plug into the local area network. No matter where they are in the building, employees have access to e-mail, important information, or presentations without having to go back to their offices.

Intel claimed that it achieved an *11 hour per week* average productivity gain. Employee satisfaction also soared as a result of the sense of empowerment and personal flexibility accompanying the mobile PC. Today all Intel employees have access to wireless technology.

MICROSOFT'S E-PROJECT AT FINGERHUT INC.

With the competitive business environment, Microsoft is one of the earliest companies that pushed for agile development process. Rather than following common software engineering methodology used in the industry, Microsoft has tried to scale up a loosely structured small-team (hacker) style of product development (Cusumano and Yoffie, Ibid., Microsoft, 2003). The Microsoft Solutions Framework (MSF) is a flexible interrelated series of models that guide it through assembling the necessary project resources to ensure that technology infrastructure and solutions meet business objectives. Great emphasis is placed on the relationship between business objectives. Technology implementations in this framework and allow Microsoft to cope better with change in technology and environment.

1.2 Team Model

The Microsoft's team model is defined as a team of peers working in interdependent and cooperating roles. Each team member has a well-defined role on the project and is focused on a specific mission. This approach encourages ownership and ultimately results in a better product. The leaders of each team are responsible for management, guidance, and coordination, while the team members focus on carrying out their missions.

In order to effectively empower quality at team member level, Microsoft gives all members of the team the ownership of the quality of the component before a product or solution is put into use. The Team Model defines roles and responsibilities, but does not define the management structure of the team. The team may report to a business unit manager, or be formed for a project from several organizations. The organization chart determines who is in charge. The team model defines how the work is divided and who is responsible for seeing that the work gets done. As a consequence, it gives each team member a stake in the success of the product or service and improves accountability for all roles.

1.3 Quality Process Model

Microsoft's process model is a milestone-based model that provides guidelines on planning and controlling results-oriented projects based on their scope, the resources available, and the schedule. According to the official Microsoft's Solutions Framework process

model, software development cycle consists of four phases: envisioning, planning, developing, and stabilization Each phase culminates in a major milestone (Cusumano and Yoffie, Ibid., Phan 2001a).

The Envisioning Phase

In this phase, product and program management uses extensive customer input to identify and prioritize product requirements. Once a new product (or in the case of infrastructure deployment, a new service) gains interest and approval, a project team is assembled to define the product. While a vision statement articulates the ultimate goals for the product or service and provides clear direction, the project scope defines the limits for a particular version of the product or service at a particular version. Further improvement of the software may be added later in future versions.

The Planning Phase

The project plan contains the functional specification—the combined plans of each team member and a schedule. The functional specification provides the project team with enough detail to identify resource, and quality requirements within schedule and budget. Based on vision statement, program management and development groups define feature functionality, architecture, and component interdependencies. Then project management coordinates schedule and arrange development teams that each contains approximately 1 manager, 3-8 developers, and 3-8 testers (Cusumano and Yoffie, Ibid.).

The Developing Phase

In this phase, program managers coordinate all design, code, and testing tasks in the project. An approved functional specification and associated project plan provide the baseline for focused development to begin. The development team sets a number of interim delivery milestones, each of which involves a full test/debug/fix cycle. Customer's involvement includes assessment of product's functionality and verification of support plan availability. They also verify that deferred functionality is documented for the next release.

The Stabilization Phase

In this phase, comprehensive internal and external tests are performed before delivery. To reduce cycle time, testing activities are set to be performed concurrently with code development. During the stabilizing phase bug finding and fixing become the primary focus. At this milestone, the product is formally turned over to the operations and support groups. Typically, the project team members either begin work on the next release or are reassigned to other development projects.

Microsoft Quality Strategies and Processes

"Doing everything in parallel with frequent synchronization and periodic stabilization" highlights Microsoft main strategy in software development. Because there are many teams working in parallel who are given some freedom to introduce new concepts and technology, they must synchronize their innovation frequently so that product components all work together before the integration test stage. Teams and individuals periodically synchronize and stabilize the product in increments as project proceeds, rather at the integration testing stage. This technique is also known as "sync-and-stabilize", "milestone", "daily build", or "nightly build".

1. *Daily Build Process:* Source codes in development are stored in the project repository where programmers check out a copy to work on their own as a private copy. If the private copy worked correctly, the master source code and the private copy are synchronized. The programmers use the "diff" tool to find the differences in the master source code since the time they checked out to make sure that they use the most recent centralized version of the master source code before the deadline time each day, say 3:00 PM. Each project has its own daily deadline time. The Source Library Manager (SLM) tool then update the private copies automatically to incorporate changes from all developers made to the source code. This update process is called merging. Inconsistencies or conflicts in the merged code detected by SLM tool must be resolved by the programmers. The build and test of merged code are performed daily. After the unit test and the "quick test" (a highly automated test which performs to make sure the new features do not conflict with other features), the private copy is checked in to the master

source code in the repository. Here the private copy is integrated and is tested with the entire system. In order to prevent overriding changes that were made by other developers on the same source code during the day, synchronization with changes performed by other developers is performed again at check in time. A developer who is designed to be the "build-master" for the day then generates the complete build of the product after the check in deadline each day. After the build, the "build-master" executes a series of automated tests. The daily build and synchronization help developer to maintain high quality of their new code developed by many interrelated teams. The daily build process also provide a code control mechanism that requires all project members to work as an integrated team in order to produce a daily version of the product that works.

One of the major problem that teams often encounter is the "breaking the build", an architectural error caused incompatibility among various functions when the build progressed overnight but no one discovers the breakdown until the next morning and the team lost valuable time in waiting for the fixes. In order to encourage developers to carefully test and synch their private copies, whoever breaks the build will become the next "build-master" and must wear a funny hat during his build-master duty.

2. *Grow Rather Than Design and Build*: Besides parallel development, Microsoft uses the iterative development process to grow existing software rather than build it from scratch. This innovation in software engineering process has long been promoted by Frederick Brooks Jr. but Microsoft is one of the pioneer software makers with a great portion of its software is grown from the previous version. For example many of the file system driver software of Windows NT 3.0 were growing from the old driver software.

3. *Change or Replace Approximately 50% of Code after each Release*: This is the "Short Half-Life of Code" strategy used by Microsoft. It is estimated that Microsoft teams change or replace 50 percent of code in their products every eighteen months or so. Because Microsoft releases a new version of a particular product every twelve to twenty four months, this suggests that 50 percent of code in a product changes or upgrades between releases. In order to comply with "short half-life" strategy, at least 50% of code in the new version must be new or upgraded. Due to rapid change in technology and the highly competitive software market, frequent change in source code is necessary for Microsoft to catch up and prevent competitors from making inroads into Microsoft products.

4. *Change in Design Must Be Justified by Features That Are At Least Twice as Good*: Learned from the OS/2 development project in which the benefits of change were less than the damage inflicted to the integrity of the software architecture, Microsoft implemented this strategy to tightly control changes. Changes that only bring a little improvement but require lots of efforts and cause inconsistencies in system software are not allowed.

5. *Build Multiple Versions Simultaneously*: Because of the global diversified market, it is critical for Microsoft to have versions of each product simultaneously ready all over the world for different hardware platforms. Microsoft Office 2000, for example, are built in numerous versions to accommodate many different languages in different operating platforms (Windows and MacIntosh) and in both Beta and production versions. Internally, Microsoft also had numerous in-debug versions that were used internally. Only versions that are stable enough are released to the market.

6. *Single Site Development*: The biblical story of Tower of Babel is one of the earliest major construction projects that failed . The main reason for this fiasco was poor communications. Microsoft in the past had a bad experience when it co-developed the OS/2 software with IBM in a multi-site development environment and attributed the failure of OS/2 product to poor communications.

7. *Continuously Test the Products as Software Grows*: There are several tests used at Microsoft for developers during the development: structured test scripts (scenario tests), gorilla testing to break the product, bug bashes for everyone to find bugs and usability testing. However as schedule pressure increase, some of the products were not thoroughly tested. Codes that are under debugging are also tested but are not included in the product code. As the project approaches the shipping date, developers will do weakly releases of production version. Because there are several versions and releases of software on hand, should latest version or release got delayed, the next best version will be used for shipping. Once the product is available, it is put to use by Microsoft employees before shipping to customers.

Microsoft e-Project at Fingerhut

Fingerhut Inc. is a catalog retailer in MN. In 1999, when Federated Department Stores wanted to enter e-commerce, it acquired Fingerhut for $1.7 billion. Unfortunately, problems with supply chain management caused Federated to suffer heavy losses from Fingerhut's operations. In the spring of 2002, Federated decided to liquidate the entire Fingerhut business.

During the summer of 2002, FAC Acquisition Inc., led by Tom Petters of Petter Groups and former Fingerhut CEO Ted Deikel, completed the repurchase of Fingerhut's headquarter in Minnetonka, MN, the Information Systems Center and customer data base in Plymouth, MN, the main distribution center at St. Cloud, MN and the Tennessee distribution center from Federated.

In November 2002, Fingerhut Direct Marketing was launched. Fred Argir, CIO and Senior VP of Petters Group, found his IT team owned $250 million worth of legacy hardware and software from the merged companies and wanted to quickly restart the online retailing business. The timing was critical for Fingerhut because in the past up to 80 percent of the company's sales were made in December. Many thought it was impossible for Fingerhut to convert their business to the web on time for the Christmas 2002 sales.

Having no formal software development team available, Fingerhut sent Request For Proposals (RFP) to major vendors including IBM, Oracle, SAP, Microsoft, and EDS. While most offered the completion date after the new year of 2003, Microsoft offered an agile development solution which allowed Fingerhut to enter the Christmas season on time. After Fingerhut phone Microsoft the winning of the bid on Wednesday of the first week of November, the development process moved quickly. A contract was drafted and the deal was signed by Friday. On Monday, a small team of 13 Microsoft employees started their development work with a Agir IT team at Fingerhut site (Microsoft, Ibid.).

Agile Development Process:

By the first week of November 2002, the agile development team quickly breakdown into small teams of 2 to 3 employees to build and integrate Web components based on common and standardized Microsoft applications such as .NET technology, XML, and SQL Server. This is an object oriented approach similar to that in XP programming approach. Each team created a set of features and functions to be built then set priority and deadline. Since Microsoft teams had a lot of experience in the e-business development using off-the-shelf components, the design, refactoring was minimized and teams moved quickly to coding by the end of the first day. As coding progressed, Microsoft's daily-build and daily-synchronization process was performed. However, unlike the very large daily synchronization at Microsoft headquarter, there was no worker assigned as build-master to fix integration problems. Since the development teams worked on common focus, collaboration, decision making freedom, mutual trust, they were self-organized and were able to solve fuzzy problems quickly.

The newly revived Fingerhut kicked off the holiday season with the release of a 216-page catalog in early November 2002. Days later, the Fingerhut web site was launched and ready for orders.
Commenting about agile development success at Fingerhut, Tom Petters said:

"When we started Fingerhut in mid-2002, we had all but written off the 2002 holiday season. But then our IT group, led by Fred Argir (now Nazca Solutions CIO) built a next generation Direct Commerce Automation system which allowed us to go online in six weeks, do over 90,000 transactions, and save 89% over traditional systems to boot. It was a great holiday season at Fingerhut, and we couldn't have done it without Fred and his team. " (Microsoft, Ibid.)

For the year of 2003, Fingerhut mailed 5,000,000 specialty catalogues to its customers, employed more than 400 employees, and sales revenues reached $140 million. In 2004, Fingerhut Direct successfully secured additional $63 million of equity through venture capitals and named Paul Hanson new CIO. In 2006, the company has 1.6 million customers, and total sales exceeded $300 million (Peake, 2007).

According to a former Fingerhut executive whom the first author interviewed, the new Fingerhut company is still pursuing the goal of serving customers for their entire buying life cycle, from the time of low-income and low-credit through their move into better income and credit level. Fingerhut could make decisions about when to make the buying transition for customers, just in time to retain a great number of customers who would move out Fingerhut's reach because of better income and credit.

LESSONS LEARNED AND NEW INSIGHTS

The e-commerce evolution in the past several years has made many changes and/or reinforced existing software strategies and development processes. The following changes are observed:

1. *Short development cycle.* Even before agile development approach was formally announced in 2001, the long development cycles that last two to five years, such as those in the OS/360, IMS, and OS/400, were long gone. The rapid advance of Internet technology and the competitive market have made long development cycles of e-commerce application unacceptable. For companies that seek "first-mover-advantage" from their e-commerce systems, a short development cycle is critical. Thanks the availability of Web Services and plug-and-play components, most major e-commerce applications can be completed within a short time.

2. *Software development projects no longer require the craftsmanship with proprietary or non-interchangeable components.* By adapting and supporting XML and Resetta Net, SSL, third party encryption products, and other Web Services, Intel used interchangeable components in its e-project to reduce time and resources required for the early e-project.

3. *Incremental or iterative component-based rather than sequential development process.* Short development cycle requirements have made traditional waterfall development processes and monolithic multi-year cycles obsolete. Companies find it impractical to build everything from scratch. After IBM's early success with multiple-version development strategies (Phan, 2001a), other dominant software makers, such as Microsoft, went further with the "grow rather than design and build (from scratch)" including Netscape with the "Internet time (incremental) development" (Cusumano and Yoffie, Ibid.).

4. *Systems do not have to be perfect to be deployed.* Short development cycles translated into deploying the system after the major bugs that caused the system to crash were removed rather than seeking close to zero defects.

5. *Personalization and customization.* Users prefer to go to the web site that knows them by their login name. Traditional software project management measured project success on meeting three major indicators: user requirements and quality, scheduling, and budget. Once the system is installed, except for different access authority, all users will access to the common screen with the same interface. Today on-line e-business systems must be personalized and customized to individual users according to each user profile. Furthermore international Web sites must be tailored to overseas customers by having information and interface conform to the national language, currency, and laws. In many global companies programs, databases, and files must also support Unicode.

6. *Frequent synchronizations and periodic stabilizations.* This is a popular strategy for developing e-commerce systems used by many companies such as Microsoft, Netscape, and developers of open systems. The long wait for the integration testing phase in order to conduct integration tests of new features has been dispensed with. Because many teams are racing the clock to work in parallel with new features, they must synchronize their innovation frequently so that product components all work together before the integration test stage. Teams and individuals periodically synchronize and stabilize the product in increments as the project proceeds, rather than at the integration testing stage.

7. *Built-in Internet security and trust mechanisms.* Stand-alone systems are also gone. All e-business systems today must have a built-in mechanism for encryption, authentication, and must comply with government privacy laws. Some companies have gone further by using Virtual Private Network (VPN) to connect with B2B partners wherever possible. Others comply with standards set by Truste and Verisign to display their seals (or trustmarks) on the Web site.

Success Factors

From the study above, key success factors in agile e-projects are:

1. *Well-defined goals, objectives, and scope*. Management was clear and explicit with respect to defining project problems and opportunities, global interfaces, goals, and objectives. To avoid scope creeping, it is important for a project manager to clearly specify not only what should be done but also what should NOT be done. At the beginning, the Intel development team made it clear that the project goal was not to build the entire e-business system from scratch and not to convert the company into an on-line one. Rather, Intel's early goal was to serve its existing market using Internet technology.

2. *Highly competent and agile-project-experienced employees*: In order to build software in agile projects, it is necessary to have highly competent team workers who are motivated with good problem solving skills to allow teams to be self-organized as projects evolve throughout development stages. As a reputable high-tech company in the Silicon Valley, Intel was staffed with highly competent workers. Furthermore, since Intel employs short development cycle in the development of its micro-chips, experience in short development cycle has increased the chance for Intel teams to succeed in agile projects.

3. *Good system security and partner trust*. To maintain B2B inter-organization trust and to protect system integrity, e-business system must strive for the best security scheme available. In this global e-commerce environment, hackers are likely to be ahead of software vendors in finding system vulnerabilities, so the best security is needed. Systems security is a continuous improvement process and will never end. In addition to the best encryption available, the system must be protected by many layers of firewalls, routers, and proxy servers when appropriate.

4. *Meeting user requirements and personalized needs*. Beyond meeting user requirements in traditional software, web based systems must offer customization and personalization to all levels of users. To meet user requirements, developers must design the system according to what customers need in an e-business system. In web based systems, developers must visit customer sites worldwide to see how the connection works in terms of speed, encryption standards, native language translation, Unicode availability, etc. Because e-business systems development deals with the latest technology, developers must be prepared to learn as a system progresses. Web content must be accurate, current, and appropriate for each individual customer. Because management, procurement, sales and marketing, and engineering functions of value chain partners and customers all have different informational needs, personalized information online allows multiple levels of customers to save time in finding necessary information.

5. *Involvement from top management and stakeholders*. One of the key factors for Intel's e-business project success was the support from Andy Grove, Intel's CEO. Top management involvement made it easier for project managers to gain cooperation from stakeholders for resources. This also facilitated negotiation for resources and headed off delays induced by bureaucracy and lack of inter-departmental cooperation.

6. *Emphasis on environment and resource dependence control*. Limited resources required the project to rely heavily on internal and external linkage, coordination, merger, and contractions to manage and control resource dependence. In order to keep up with frequent patches and updates from vendors and to maintain good system security, extra human resources in system support and maintenance are needed.

7. *Assuming customers will not be able to connect and access the system if the system requires special capabilities*. One of the fatal mistakes that some online companies make is to assume that customers have proper infrastructure for e-commerce. In the early days of e-commerce many e-commerce projects failed because developers assumed that customers had capabilities such as broadband or T1 connection to use special features of their sites, or that they had the appropriate SSL 128 bit encryption. In order to allow all mid-tier account customers to access its pilot system in 1998, Intel engineers and technicians went to great length to detect and fix problems in network connectivity prior to deployment.

8. *Modular architecture*. A modular system will allow an e-business system to grow quickly. By separating front-end functionality of the web site from the back office systems, frequent updates in front-end applications can be achieved without affecting enterprise type-applications that commonly require longer release cycles.

Room for Improvement

Despite the successes mentioned above, there is room for improvement in the system infrastructure and development approach. Although there is no indication that Intel used the SEI's CMMI for improving its development process, upgrading its current software process capability and maturity would improve the efficiency and productivity of the development project. For example, Genuchten et al. (2001) have examined corporate use of groupware for software inspections, building upon the work of Fagan (1986) and Humphrey (1989, 1995). Humphrey has also developed a *Team Software Process (TSP)* that extends his *Personal Software Process (PSP)* towards a more systematic approach to software engineering that would further enable consistent development of higher quality systems (1999). With this as a foundation, the opportunity to create ever more complex, yet reliable and more easily maintained, systems becomes increasingly possible. This further enables efficient creation of more effective inter-organizational systems to support e-business.

CONCLUSIONS

With the rapid advance in technology, it is important that companies redesign business processes and systems to handle the shift in the business paradigm. In agile e-projects, developers face new issues that may not have existed in traditional software development. These include using short development cycles with frequent synchronizations and stabilizations, more frequent software maintenance and updates, modular and component-based development, open system software, encryption, third-party authentication, and user

personalization and customization. This study has provided some new insights into the development of e-business systems and confirmed the validity of agile paradigms for e-commerce systems. As exemplified by the problems faced and the success achieved by Intel's e-business project, the strategies and success factors discussed in this paper can serve as suggestions and guidance for project managers seeking to improve their e-business project management.

ACKNOWLEDGEMENTS

The authors would like to thank Mr. Phuc Trong Than, General Manager of Intel Corp. Vietnam and Intel Corp. for the availability of information on this development project and former employees at Fingerhut Corp. for the information of the agile project at Fingerhut.

REFERENCES

"References available upon request from [Dien D. Phan.]"