Have the Factors Affecting Software New Product Development (S-NPD) Changed in the Age of Mobile Apps and Agile Methods?: A Position Paper

Samuel A. Ajila
Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario, Canada

***Abstract*--It has been long recognized that Time-to-Market, Cost-of-Delay, and Uncertainty are the three major factors that can impact New Product Development (NPD) process. Software New Product Development (S-NPD) is even more notorious with delay and cost overrun. Time-to-Market is the length of time it takes to get a product from idea to marketplace. Therefore, shortening the production cycle is a major goal for software development companies so that they can get to the market faster and be the "first mover" of a particular product. Delay on the other hand can bring added cost to the product development and this is referred to as Cost-of-Delay. This cost is normally hidden. The opposite of delay is speed but in S-NPD process *speed* has a monetary value and if the cost of speed is too expensive to achieve, then, delay may be a preferred option. Agile method was introduced in the 1990s and the aim is to shorten the time-to-market and reduce delay. Mobile Apps are software applications designed to run on mobile devices such as smartphones and tablets. The constraints attached to mobile devices make the development of Apps more difficult compared to "desktop" software development.**

The goal of this position paper is to examine the three factors: Time-to-Market, Cost-of-Delay and Uncertainty through "observation" vis-à-vis Apps development and the use of agile methods in order to see if something has changed over the years. In addition, a controlled group interview was conducted to support the observation method. The questions this paper seeks to answer include. Are we getting to market on time? Are we delivering software product on budget? Do we have reduced uncertainty when it comes to Apps development and the use of agile method?

I. INTRODUCTION

New product development (NPD) process is defined as a method of idea generation, transmission, concept development and testing, business analysis, market evaluation, implementation (programming in the case of software), marketing, and monitoring and product progress [5], [7], [8], [11]. NPD involves both engineering and marketing aspects, where both together bring a new product into the market. This paper focuses on the engineering aspect of NPD.

In today's highly competitive industrial environment, new products are divided into three major categories: new to market, new to firm, or new to both market and firm [7], [12], [13]. Products from these categories, all share a common cycle defined as Product life Cycle (PLC). In fact, for any kind of product, the PLC is composed of four stages: introduction, growth, maturity, and decline. Every product goes through these stages from its birth to death.

In the last decade, the number of new software applications (i.e. Apps) introductions into the markets has increased exponentially because of the importance attached to the mobile computing industries and the contributions to the new IT economy. At the same time, managing the NPD of these apps has become challenging for small to medium sized industries involve in the development and marketing of these products. In addition, it requires extensive investment in terms of money and human resources. Furthermore, from observations and controlled interview methods, for every twenty five apps products ideas, about eight enter development, three or four are launched, and at best only two will succeed. Even then, in terms of return on investment only one out of every ten apps launched will be viable in the first six months to one year of introduction to the market. The success rate in terms of return on investment is very low although if one software apps succeeded the company can make a lot of money to offset the loss in terms of apps that did not get to launch stage [15], [1].

What is really important in every NPD process is shortening the product development cycle [12]. As a result, taking the advantages of rapid development, would lead to benefiting from being the pioneer and the first mover advantage thus setting better prices, and then have dominant market share and customer loyalty [4]. All these advantages drive software applications industries to compete on time to market and reduce their NPD cycle time.

There are three major factors that may have effects any NPD process [21], [27]. These are: time-to-market, cost-of-delay, and uncertainty (i.e. risk). Among these, time-to-market is the key factor involving cost-of-delay, and analyzing risk. The objective of this paper is to examining by observation and through controlled interviews if the use of Agile methodology in NPD process of developing software applications (i.e. Apps) has positive impact on the three factors especially in the period time-to-market reduction.

The rest of this paper is organized as follows. Section two gives the background and problematics and introduces the foundation for the framework. Section three discusses the framework outlining the problem setting and the procedure. Section four presents the lessons learned from the research results. Section five discusses the outcomes, enumerates sources of bias in the research and gives a conclusion.

II. BACKGROUND AND PROBLEMATICS

A. Software New Product Development (S-NPD) Process

The S-NPD process consists of activities carried out by software development industries when developing and launching software applications (i.e. Apps). A new software application product introduced in the market involves a series

of stages, starting with product idea that is evaluated after consultation with different stakeholders especially the customers then it is developed, tested, and deployed to the market [21], [23]. The sequence of activities for apps S-NPD process is slightly different compare with other NPD process. In addition, it also differs from industry to industry. We present a general sequence used in this research project in Fig. 1 below. The process is not linear in the sense that there is overlap between phases but these overlaps vary between the initial phases that are more or less linear and the later stages with up to 80% overlap [26].

- *Initial market needs survey*: This phase is normally carried out depending on the new product, the targeted customer base, and industry type. In the case of "general needs" apps, the survey is informal and it is done online starting with an idea from a "fan" base. In the case of domain specific apps, the "survey" can be started by either the customer base or the apps industry. In this case, it can start with a complaint or suggestion from a customer or from several customers. The industry can then consult with their customer base if they have similar need or something close to that need.
- *Idea generation based on needs*: Once the need is established from at least 5% of the customer base, the industry then develops a general framework that formally defined the need and application to business process. The framework will include potential benefits to the customer if eventually, the apps are deployed.
- *Stakeholder consultation*: Once the initial framework is developed, the stakeholders are consulted. These include the top management, the marketing department, the R & D section, and the customers. The marketing department will then package the framework and send it to their entire customer. The expectation is that between 5% and 15% of the customer base will return a positive response to the new product. In addition, a "go ahead memo" and seed money is required from the top management.
- *Concept development*: If the go ahead is given, a prototype is developed using either evolutionary prototyping approach or revolutionary prototyping depending on the nature of the new product [1]. If the new product will be integrated with other products and probably used the same database, then an evolutionary approach is used with reduced number of functionalities. If the product will be stand-alone or little interaction with other company products, then the best approach is a revolutionary prototyping approach.
- *Customer Consultation 1*: The prototype is sent to the customer base for feedback.
- *Evaluation and business analysis*: The initial concept and the response from the customers are analyzed and evaluated from business point of view with emphasis on the potential risk (uncertainty) associated with the product.
- *Design & development and testing & integration*: The full implementation commence after the evaluation. There is need to enumerate mitigating factors if there are potential risks associated with the product.

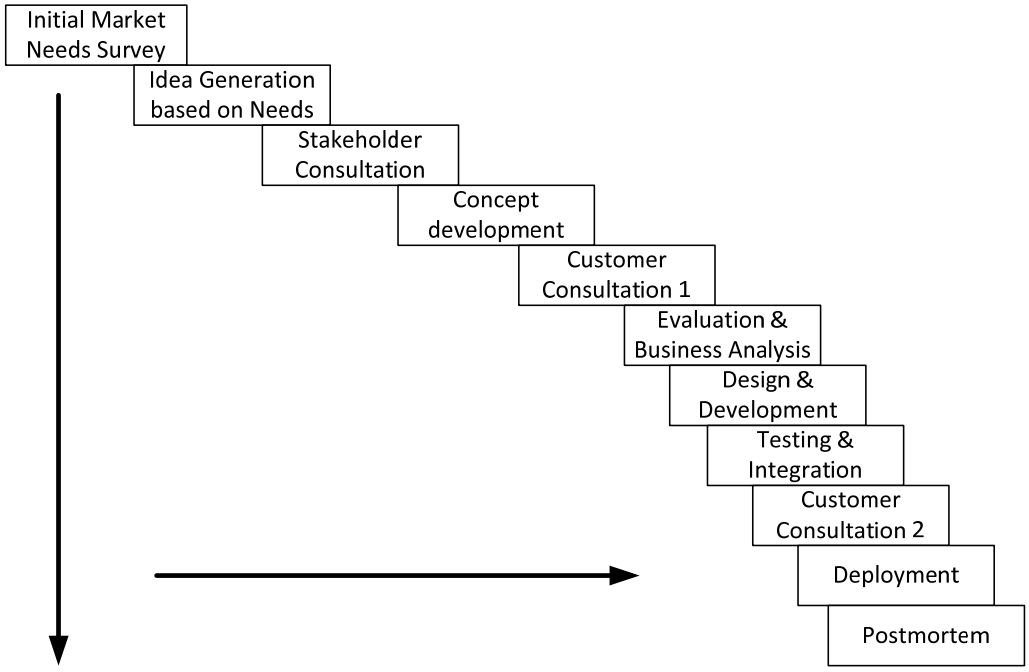

Figure 1: Sequence of Activities in S-NDP lifecycle

- *Customer consultation 2*: In this case, the developer can choose to give a final version of the product to selected customers for feedbacks or give a "reduced" final version with some important functionality to all the customers for feedbacks. In the case of Apps, the general approach is to give out free-of-charge a final version with secondary functionalities and one or two primary functionalities. The hope here is that many clients will like the App and be willing to pay for the full version.
- *Deployment:* Depending on the result of previous step, the deployment is either done at once or gradually. For many commercial Apps, the deployment is generally done gradually.
- *Postmortem*: Finally, a group of experts in the company are tasked with carrying out the postmortem and building either on the success or failure of the project.

B. Time-to-Market

Numerous terms such as cycle time, innovation speed, NPD time, and speed-to-market are related to speed in S-NPD [20], [22]. Among these, the concept of speed-to-market plays an important role in S-NPD cycle time reduction. Time is a determinant factor in every product development process and it is the most important factor in S-NPD. The term "time-to-market" referred to the length of time it takes to develop a product from idea to the marketplace [5]. There are several approaches that could be taken to reduce the duration of the time-to-market period [18]. The most important decision is selecting the right approach, because major S-NPD failures are due to lack of understanding of the requirements or the customers' intention and therefore applying an unsuitable time-reduction approach in order to get to the market quickly will results in failure. Hence, an approach is turned to a major concern rather than a success factor. Indeed, any time-reduction approach, must address three main concerns: skip steps to save time, speed is too expensive, speed really yields productivity [20], [22].

Skipping steps [15] does happen when S-NPD acceleration matters. Obviously, this is accomplished by skipping some of the work to be done which could be in the requirements analysis stage or the design process. Although major skipping occur at these levels, but today's highly competitive market, and continuous customers' demands can make skipping vital activities unproductive and unbeneficial routes especially in software development. A good way of avoiding steps could be in using a flexible process (e.g. Agile methodology) or components reuse and thereby giving more value to product release-time rather than implementing unnecessary features. Therefore, the key observation from skipping steps is to identify the unavoidable details and at the same time ignoring the extra and unnecessary steps.

In S-NPD process, *speed* has its own money value and if it is too expensive to achieve, product managers will not reduce the product cycle-time at the expense of paying for the compressed time. In general, there is always a tradeoff between time and development cost. In the same way, delaying the product development process has a hidden cost called "The Cost-of-Delay." Hence, a balance between cost-of-speed and cost-of-delay is needed and very important [9], [11].

Productivity is another crucial factor in reducing production cycle time. Product managers determine their product cycle time by analyzing the amount of productivity they can get out of their resources. In fact, the more products get developed at a given time duration, the less costly is the production process. One important issue that must be considered is that faster development requires enhanced staff and optimized techniques which in turn will add–in a new cost to development.

C. Uncertainty in S-NPD

Uncertainty is defined as immeasurable probability or unknown probability in economics [22]. It is also known to be the lack of information, lack of knowledge, and inability to assign probabilities on the effect of a given factor on success or failure of a decision. In [15], uncertainty is referred to as the unpredictability of the environment, inability to predict the impact of environmental change, and the inability to predict the consequence of a choice response. Therefore, in each product development, where customers' demand and marketplace play the role of outside environment of a company, uncertainty is involved. This is due to the fact that we can never be 100% sure about what customers' demands or marketplace situation are going to be. Also, suppliers, competitors, distributors, regulator factors, and technology deliver major portion of uncertainty into software product development process.

D. Agile Methodology

Agile approaches are software incremental development methods in which increments are small starting with primary functionalities and new versions are released as time goes on [4], [5], [19]. The software business is operating in a global market and rapidly changing environment with increasing complexity. Economic conditions are changing coupled with competing IT products and services and the demand for on-time and on-budget delivery. So, software has to be developed quickly and on time to respond to the different pressures (competition, on-time delivery, and costs). A solution in this wise could be "rapid development" methodology [28].

The traditional software process approaches that are based on complete specification of requirements, then design, build, and test are not geared towards rapid software development [14]. In any case, the modern software business operates in a changing environment and it is seldom practical to have or to derive a complete set of stable software requirements. Often, it is after a software product is delivered and users gained experience using it that the real requirements are discovered.

Rapid software development idea is not new however the notion took off in the early 2000 with the development of agile approaches such as extreme programing (XP) and

scrum. All rapid development approaches share three characteristics: requirement analysis (if any), design, and implementation are interleaved; the software is developed in increments; and the interface is designed and developed using evolutionary approach with 4G programming languages. Given below is the brief summary on two of the most popular approaches in agile methods: extreme programming and Scrum.

Extreme programming (XP) [3], [4], [6], [16], [24], [29] is one of the well-known agile methods. Software requirements are expressed as *scenarios* often called *user stories*. They are implemented as series of tasks. Programmers work in pairs and develop tests before writing source code [4]. In general, there is a short time span between releases. In XP, small increments and frequent releases of software systems are supported; customer involvement is paramount through direct engagement; people, not process is the politic; change is embraced; and simplicity is maintained [19].

Contrary to believe, Scrum is not an agile method in the technical sense of prescribing specific programming practices rather it is a management approach for iterative development. It is seldom used alone. In a sense it can be used in conjunction with XP in order to manage the iterative and incremental development approach [17]. Scrum consists of three phases: planning and architectural design, sprint cycle, and project closure. The first phase is where general objectives for the project, design, and architecture are established. The third phase is where the project is wrapped up and postmortems are defined. The middle phase called *sprint* is very interesting and it consists of four spiral steps: Assess, Select, Develop, and Review, and it continues in the spiral re-starting with assessment (i.e. Assess) and so on. The sprint phase is the central part of Scrum and this is what is normally used to support the management of iteration in other Agile methods (e.g. XP). The central idea in Scrum is that the whole software development team needs to be empowered to make decision, that is, every member of the team is "a project manager." Important characteristics of Scrum are as follow: fixed length sprints of between two to four weeks; the list of tasks to be done serves as the starting point; selection involves all project team members; short daily meetings organized by the scrum master; and review and presentation to stakeholders [4], [10], [17], [18], [24], [28].

Agile methodology is developed to be used by small project teams and for small to medium sized software project. It is very difficult to use agile process for a very large software project. Nevertheless, there has been great deal of interests in scaling agile methodology to develop large software projects. Two of such perspectives are *scaling up* and *scaling out* [25] but the discussion of these perspectives is beyond the scope of this paper. However, there are few problems in the application of agile method. In the first place, customers are integral part of agile approach and it can be very difficult to maintain the interest of customers involved in the process. Secondly, paired team members may be unsuited to the intense involvement that characterizes agile methods. Thirdly, it can be very difficult prioritizing changes when there are multiple stakeholders with conflicting interests. Lastly, maintaining simplicity which is the hallmark of agile method demands extra effort on the part of the project team members and this can delay project take off and extend delivery time.

III. THE FRAMEWORK

A. The Setting

The setting for this research project is based on a final year Software Product Management course (SYSC 4106) piggy-back with a graduate course (TTMG 5006) in Software Project Management at the Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario, Canada. I have been teaching the course since fall 2002 except when I was on sabbatical leave. It is a three credit hour course with five contact hours per week – three hours of theory teaching and two hours of practice otherwise called laboratory assignments. Starting from fall 2007, I developed a set of project assignments for students developing iOS software apps using agile methodology (extreme programing and Scrum) and the classical waterfall model – the so called V-model. Each year we have two apps development projects with one project group using agile method and the other project group using V-model. In addition, we set up two customer groups that will interact with the technical groups. The research method is by observation based on a controlled qualitative approach. In addition interview was conducted using a controlled group of former students. The result of the interview process is presented in section five.

B. The Procedure

- A project meeting was organized during the second week of the semester to discuss the projects and form the various groups (one group for agile methods, one group for V-model, and two customer groups).
- During the third and fourth weeks, we developed the projects specifications, schedules, memorandum of agreement between the customer groups and the technical development groups. In addition, we set out rules of engagements as follow:
  - There should be no communication between the two technical groups during the project period.
  - The technical groups will be working on different project at the same time. For instance, if group one is working on project number two, then group two will be working on project number one.
  - The project duration is four weeks for each project and not more than three hours of programming and discussion per week at a dedicated computer laboratory.
  - The project customer groups will meet with the technical project groups weekly and not more than one hour meeting to discuss progress and problems. Each customer group is in charge of different project.

  - The instructor will meet with the groups separately bi-weekly to discuss mainly problems encountered and possible solution and to collect "lessons learned" during the period.
  - The total duration for the two projects is eight weeks.
  - The time to market was set to the last week of the semester (i.e. week 15) when all the class members gathered for five hours of presentations, discussions, selling, buying, and to develop postmortems for the projects.
  - Each technical development team must use the S-NDP lifecycle defined in figure 1, section 2.1 starting with the "concept development phase." The three initial phases of the lifecycle are eliminated because these were done together as a class during the first two weeks.
- Different students groups work on the same projects for three consecutive years before changing to a new set of projects. This is done to collect enough lessons learned and to try as much as possible to reduce bias.
- In order to introduce technical uncertainty, the instructor organizes a weekly meeting with the customer groups to discuss project requirements with them. The idea here is that when they meet with the technical project teams, they can either change a project requirement or introduce a new requirement in order to see how the project team will react. The instructor is an observer during these meetings.

IV. LESSONS LEARNED

In this section the lessons learned are divided into three parts: Time-to-market, cost-of-delay, and uncertainty.

A. Time-to-Market

Time is a determinant in every product development and software apps are not exception. The quicker the product gets to the market the larger the size of market share a company gets vis-à-vis the new product or technology. The number one lesson from this research project is that the use of agile methodology in developing apps can take the product to the market quicker compared to using the rigid linear process and in our case the waterfall V-model. Over the eight years period (2006/07 to 2014/15 academic years), the Agile project groups using the Agile methods were able to either complete the implementation of the apps or complete at least all the primary functionalities of the apps compared to the group using waterfall V-model.

Now, if time-to-market is broken down to two aspects: *productivity* and *beating the competition* to market, the winner is still the agile methods group. Firstly, the agile method is suited to our definition of S-NPD lifecycle phases. Secondly, the agile methods groups were able to interact very well with the customers (i.e. the project customer groups) because one of the important characteristics of agile methodology is customer involvement which is privileged over process. The V-model groups suffered from rigid process approach that gives more importance to process compared to people. Although we did not take the product to a "real" market but, from the controlled laboratory observation we can conclude that the agile groups will get to the market faster than the V-model groups. However, we could not measure the impact of *increase sales* which is a part of beating the competition to market. However, in terms of "product likeness", the agile groups also obtained the best results. It was very difficult to measure productivity by observations. Productivity aspect is taken into account as a factor in compressing product cycle time and the more products get developed at a given time duration, the better the productivity and the less costly is the production process. Our problem is twofold in that the two groups are using the same S-NDP lifecycle phases and they have to follow a rigid project schedule. So, in terms of product cycle, there is no difference between the two teams. However, if we used speed to market as a measure of productivity then the agile groups will be on top.

B. Cost-of-Delay

The cost-of-delay in our case is measured by inviting 20 students to check the apps developed by the two groups and judging by "likeness" that is, intention to buy the apps we are able to determine which apps "sells" the most. In general, the cost-of-delay is the amount of profit a company loses when its product is delayed by an amount of time (a day, a week or a month). So, we created a simple profit-and-loss statement that says that if a product sells (i.e. liked by the invited students) at least ten copies the cost of production is covered. Using this subjective measurement, the agile method was the winner because the agile groups were able either to complete their apps or complete more functionalities compared to the V-model group. In this case, the cost of delay is the profit lost by the V-model group compared to the agile method group profit. However, it is very important to note that the cost-of-delay varies widely for different projects within a company. For example, an open source apps developed by a company to source interests and gauge the marketability of the product will have a different cost-of-delay compared to a product going into the market for profit. In addition, the cost-of-delay affects the following [5], [7], [8], [13], [27]: product introduction delay, development budget overrun, product cost overrun, and product performance shortfall. In our case, we only measure "product introduction delay" and part of product performance. We assume that there is no budget or cost overrun.

C. Uncertainty

In NPD, uncertainty can be divided into two major types: technological and market [9], [11], [20]. Technological uncertainty refers to the degree of unfamiliarity with a given technology or the change in the technology used for developing product which is new and rapidly evolving. The market uncertainty refers to the ambiguity about customers' demand and satisfaction. The market uncertainty increases

when there is a fast-changing market or a new market, in which customers and their needs are not clearly known. However, the categorization of S-NPD uncertainty is generally based on risk factors and they are potential problems that might affect the successful completion of a software project. Dealing with software risks involve risk identification, analysis, planning (or decomposition), and monitoring (or reduction). In S-NPD, it is very important when dealing with a risk factor to have a balance between the impact, the probability of occurrence, and management concern (i.e. costs and deadlines). In this research project, the variations in risk factors [25] is divided into **known risk factors** (or the so called **Acceptable** risks) which can be effectively managed thorough a comprehensive project management plan; **less known risk factors** (or the so called **As Low As Reasonable Practical ALARP** risks) which can be handled thorough risk management plan; and **unknown risk factors** (or the so called **Intolerable** or **Unacceptable** risks) which are very difficult to mitigate and the best plan is to have contingency budget or try to eliminate the risks if possible. Known risk factor has little or no uncertainty and if any, the uncertainty is specific. Less known risk factor falls between specific uncertainty and general uncertainty. The unknown risk factor has unknown information and with the risk uncertainty realm falling into the complete uncertainty spectrum and very dangerous.

In our case, we are dealing with technological uncertainty and it moderates the relationship between speed-to-market and new product (i.e. the apps) success. Studying S-NPD processes of different companies have shown that market uncertainties are strongly associated with longer development times than are technological uncertainties [11]. This is due to the fact that it is much easier to adapt to the new technology rather than fully understanding customers' needs [11], [12], [20]. To resolve technological uncertainties only internal interactions are required. In the case of market uncertainties interactions are external and are often performed in trial-error manner.

The results of our study show that:

- Agile development approach does not necessarily bring the expected commercial success, and in fact, for the situation where the development of apps is involved, rapid development could results in failure if proper plan is not in place for the evolution of the product. XP and Scrum development methods focus on the "now" that is, getting the product to the market quickly and getting a share of the market. But, software business is never certain because the requirements are not fully known until the product is used and after the release of many versions of the same product. So, in S-NPD, evolution is as important as the market share [1], [2]. A company can lose its market share very quickly if the quality of its product suffers adverse effects. This is more serious if there is no effective product evolution mechanism in place.
- If the risk factors are known, agile methods are better placed compared to linear methods of waterfall models. The only problem here is that all the risk factors in software development project are never known in advance and there is still need to have contingency plan.
- In the case of less known or unknown risk factors, the rigid processes (i.e. waterfall type processes) are better to handle such situations compared to agile method methods.

V. DISCUSSION AND CONCLUSION

In this research project we have studied the impact of agile methods and a linear process on the three factors (time-to-market, cost-of-delay, and uncertainty) affecting Software New Product Development (S-NPD) vis-à-vis the development of software apps. Time as a critical factor affects product development significantly by giving the first-mover advantages such as: setting higher prices, garnering dominant market share, and acquiring customer loyalty. Companies set their number-one goal being the early-producer to benefit from these advantages. Some set of strategies such as: time-based competition, first-mover strategy, fast product development cycle time and on-time schedule performance are implemented by companies to achieve advantages over competitors. The important point is selecting the right strategy which best suits the company resources and financial goals. Thus, managers follow a policy which shortens their production cycle and therefore gets them to the market sooner to benefit from a competitive advantage and a longer sales life. The concept of cost-of-delay as a second factor gives value to time and discusses the importance of early or on-time production.

Product development is affected by four types of uncertainty: Market Newness, Market Turbulence, Technological Turbulence, and Technological Novelty. Studies have shown that if a company is not familiar with technology or market uncertainty; it will require greater effort and commitment on the part of the company; and it would also demand that the firm works harder to get individuals and organizations closer to each other to reduce the uncertainty. Our focus in this paper is on the technology uncertainty. Judging by the lessons learned (section 4.0) from the research project using control observations over a period of eight years, we can conclude (or propose) that something has changed in S-NPD through the use of Agile methods in the development of software apps and this change is somehow positive.

In addition, the research conducted a controlled telephone interview with 50 former students now working for IT companies with at least 30 of them working in the software apps development related companies. The results of the interview are as follow. About 80% of the 50 students confirmed that they are using hybrid version of agile methods to develop software apps; about 90% of the 30 former students working in software apps related industries confirmed that they are using hybrid agile methods that include XP and modified Scrum process. The Scrum is modified to account for evolution and maintenance [1], [2] of

the products. Out of the 50 participants in the telephone interview, 74% confirmed that they were able to get to the market quicker compared to when they were using rigid linear process. Finally, 55% indicated that handling risks is very difficult when using agile methodology especially when the risk factors are either less known or unknown.

This research project recognizes the fact that time-to-market, cost-of-delay, and uncertainty are three important research areas for improving new product development process especially S-NPD. The following are possible future research opportunities. A longitudinal research approach can be used in the following cases:

A. Time-to-market

Longitudinal research can be used to study the effect of rapid production on profitability and other advantages of being the first-mover. A possible research criterion in this area would be studying two or more competitor companies during two separate periods (let's say two production cycles i.e. two new products) where at any point in time only one of them gets the chance to be the early producer with other playing the role of fast-follower. The purpose of this study would be to analyze the effect of quick production (i.e. the use of agile methods in S-NPD) on current and future success of developments. The question here is how early can the production of a new product brings in success for next projects at hand even though the company will not be able to be the first-mover for the next productions.

B. Cost-of-delay

Aside from monetary loss of late production, we can study other aspects of delaying production. For example, lateness in production could have a direct impact on customers when the customer is another manufacturing company that uses our product as raw material for its assembly lines. That is, developing software apps for another IT company to be included in a larger software project. Studying the effect of delayed production on customer's reliability can be a possible research area.

C. Uncertainty

Different types and levels of uncertainty have different effects on every S-NPD. Size of the company is also a negotiable characteristic when looking at the effect of different uncertainties on new product success. Therefore, performing studies using longitudinal approach on large and medium sized samples would provide an interesting set of results.

This paper will not be completed without addressing sources of bias. There are many sources of bias in this research project. The first is that the experiments (i.e. product requirements) are well-defined and the sample (i.e. number of projects and the interview sample) is very small. Secondly, it was conducted in a laboratory setting with well-defined boundaries so the influence of environmental external factors is limited or nonexistence. Thirdly, it is a one-man show in the sense that the course instructor is the sole arbitrator and judge of the projects. Fourthly, the project did not consider factors such as market turbulence, technology turbulence, technology novelty, integration of multiple technologies, newness to customers, staffing teams adequately, and the complexity associated with product and customers. Finally, everything (i.e. project requirements, project team formation, processes, and the interviews) in this project is controlled.

In conclusion, this paper proposes the following hypothesis: "The factors affecting software new product development (S-NPD) vis-à-vis the development of software mobile apps and using agile methodologies have shifted positively towards getting to the market early and reducing cost-of-delay but uncertainty is still a concern."

REFERENCES

[1] Ajila A. Samuel and Badara A. Kaba, Evolution Support Mechanisms for Software Product Line Process, *The Journal of Systems and Software*, Volume 81, Issue 10, October 2008, pp 1784-1801

[2] Ajila A. Samuel, The Impact of Knowledge Delivery Factors on New Product Development Teams – A Quantitative Analysis of Software Development Efficiency, *International Journal of Product Development (IJPD)*, Vol. 6, No. 2, July 2008, pp 98 – 117

[3] Arisholm E., H. Gallis, T. Dyba, and D. I.K. Sjoberg, Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise, *IEEE Transactions on Software Engineering*, 33 (2) 65 – 86, 2007

[4] Begel Andrew and Nachiappan Nagappan, Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study, *ESEM '07 Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, Pages 255-264

[5] Bhuiyan Nadia, A framework for successful new product development, *Journal of Industrial Engineering and Management (JIEM)*, 2011 – 4(4):746-770 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423, http://dx.doi.org/10.3926/jiem.334

[6] Blom Martin, Is Scrum and XP suitable for CSE Development?, *Procedia Computer Science* 1 (2012) 1511-1517

[7] Chen Jiyao, Reilly R.R., Lynn G.S., The impacts of speed-to-market on new product success: the moderating effects of uncertainty, *IEEE Transactions on Engineering Management*, Volume 52, Issue 2, May 2005 , pp. 199 – 212.

[8] Clark K. B. and T. Fujimoto, Product Development Performance: Strategy, Organization, and Management in the World Auto Industry. Boston, MA: *Harvard Business School Press*, 1991

[9] Crawford C. M., "The hidden costs of accelerated product development," *J. Proc. Innovative Management*, vol. 9, no. 3, 1992, pp. 188–199.

[10] Demarco T. and B. Boehm, The Agile Methods Fray, *IEEE Computing*, 35 (6), 90 – 92, 2002

[11] Emmanuelides P. A., Determinants of product development time: A. framework for analysis, *Acad. Manage. Proc*., 1992, pp. 342-346.

[12] Flint, Daniel J, Compressing New Product Success-to-Success Cycle Time: Improving New Product Ideation Through Deep Customer Value Understanding (2002), *Industrial Marketing Management*, 31, pp. 305-315.

[13] "Fast-Cycle Product Development" Preston G. Smith, *Engineering Management Journal*, June 1990, pp. 11-16

[14] Hopkins R. and K. Jenkins, Eating the IT Elephant: Moving from Greenfield Development to Brownfield, Boston, Mass., *IBM Press*, 2008.

[15] Langerak, F., Hultink, E.J, The impact of new product development acceleration approaches on speed and profitability: lessons for pioneers and fast followers, *IEEE Transactions on Engineering Management*, Volume 52, Issue 1, Feb. 2005, pp. 30 – 42.

[16] Moe Nils Brede, Torgeir Dingsøyr, Tore Dybå, A teamwork model for understanding an agile team: A case study of a Scrum project, *Information and Software Technology* 52 (2010) 480–491

[17] Nageswara Kuda, Rao G., Kavita Naidu, Praneeth Chakka, A Study of the Agile Software Development Methods, Applicability and Implications in Industry, *International Journal of Software Engineering and Its Applications* Vol. 5 No. 2, April, 2011

[18] Pathak Kaushal and Anju Saha, Review of Agile Software Development Methodologies, Volume 3, Issue 2, February 2013 ISSN: 2277 128X *International Journal of Advanced Research in Computer Science and Software Engineering*

[19] Preston G. Smith, The Business of Rapid Prototyping, *Rapid Prototyping Journal*, October 1999, pp. 179-185.

[20] Preston G. Smith, From Experience: Reaping Benefit from Speed to Market, *Journal of Product Innovation Management*, May 1999, pp. 222-230.

[21] Preston G. Smith and Don Reinertsen, "Shortening the Product Development Cycle," *Research-Technology Management*, May-June 1992, pp. 44-49.

[22] Roman Pichler and Preston Smith, Developing Your Products in Half the Time, *CriticalEYE*, December 2003.

[23] Reinertsen Don and Preston G. Smith, The Strategist's Role in Shortening Product Development, *The Journal of Business Strategy*, July/August 1991, pp. 18-22.

[24] Schwaber Ken and Jeff Sutherland , Software in 30 Days: How Agile Managers Beat the Odds, Delight Their Customers, and Leave Competitors In the Dust, John Wiley & Sons, 2012

[25] Sommerville Ian, Software Engineering , 9th edition, Pearson, 2011, ISBN 10: 0-13-705346-0

[26] Taft, J., and Barclay, I.: 'Trends in new product development and the establishment of simultaneous engineering', *Proceedings of the International Conference of Product Development Management Association*, Chicago, pp. 119-122.

[27] Takeuchi, Hirotaka, and Ikujiro Nonaka. "The New New Product Development Game." *Harvard Business Review* 64, no. 1 (January–February 1986), pp 137-146

[28] Turk D., R. France, B. Rumpe, Assumptions Underlying Agile Software Development Processes, *Journal of Database Management*, Volume 16, No. 4, pp. 62-87, October-December 2005

[29] Vlaanderen Kevin, Slinger Jansen, Sjaak Brinkkemper, Erik Jaspers, The agile requirements refinery: Applying SCRUM principles to software product management, *Information and Software Technology* 53 (2011) 58–70