2009

Management of Softwrare Product Development, Innovation and Adaptability

Michael L. Harris
Indiana University – Southeast, harris60@ius.edu

Alan Hevner
University of South Florida, ahevner@usf.edu

Rosann Webb Collins
University of South Florida, rcollins@coba.usf.edu

Management of Software Product Development, Innovation and Adaptability

Michael L. Harris
School of Business
Indiana University - Southeast
harris60@ius.edu

Alan R. Hevner
College of Business
University of South Florida
ahevner@coba.usf.edu

Rosann Webb Collins
College of Business
University of South Florida
rcollins@coba.usf.edu

ABSTRACT

This inductive study develops a model of innovation and adaptability in software product development. It is based on a case study of a company that is transitioning from a custom development approach to a product-based solution. The emergent model represents a synthesis of the case study findings and the enfolding literature from traditional product development and software development. The goal of the emergent software product development model is to guide organizations in their selection of development processes.

Keywords

Software Product Development; Innovation; Adaptable Software Development.

INTRODUCTION

In this research we examine the central tension between flexibility and control in the software product development environment. Our focus is a software company that is transitioning to a product-based software approach. Increasingly, software systems development has become a product-orientated endeavor. In the early days, companies needed large development staffs to build custom transaction processing solutions. Each custom solution was tailored to the needs of a single company. The shift to a product-based approach began as new software companies supplied application toolkits, application servers, components, and eventually web services as building blocks for product-based solutions.

Products are fundamentally affected by their need to survive in a competitive marketplace. A successful product must offer a unique value proposition. This value proposition creates a reason for customers to choose the product over competitive offers. To the extent that the value proposition is based on product features, or cost-structure it will impact product design. Furthermore, product delivery options (e.g., packaged software versus web services) can also affect design. As a result, a study of software product development is also a study of software innovation. Furthermore, the marketplace is fluid. Low barriers to competitive entry and changing technology constantly disrupt the market direction. A product concept that looks competitive when development begins may fall flat 12 months later when the product is launched.

The traditional organizational recommendation for an innovative, adaptive environment is an organic structure. However, a purely organic approach may devolve to the 'code-and-fix' development environment that is known to be problematic (McConnell 1996). Our study examines the central tension between flexibility and structure that must exist in order to allow software product development to adapt, yet remain on target. In considering this issue, the study builds an emergent model of software product development. The model is not prescriptive in terms of the software process (e.g., waterfall, spiral, etc.); rather it identifies the interim artifacts that could be used as control mechanisms in any software process.

RESEARCH APPROACH

An emergent model of the software product development for eSoft evolved via an iterative loop between the case study analysis and the literature review. An initial model, based on the case study analysis, was revised based on iterations of (a) comparison with the literature, (b) checking differences with the case study data, and (b) model revision, as appropriate. The entire iterative process of cross-checking between the interviews and the literature continued until the model reached stability, as has been recommended in the inductive research process (Eisenhardt 1989).

For the case study analysis we use an inductive approach to build a model of software product development activities. While the study clearly has many of the characteristics of interpretive research it is more accurately described as an inductive study because the model of software product development is derived from the data (Eisenhardt 1989). The study uses case study data gathered by a researcher who spent over six months at the target company as a participant-observer. The researcher was involved in company meetings and daily activities. The researcher also conducted 29 open-ended interviews of employees. These interviews were initially captured through audio recordings and subsequently transcribed into text files. The result constitutes a rich data set for our analysis.

Initially, two different researchers independently analyzed fifteen percent of the transcribed interviews and coded the discussions according to categories that unfolded organically from the transcripts. Since the process used an emergent approach, it was not possible to calculate an inter-rater reliability using a priori concept categories. However, the researchers met regularly to discuss and consolidate their independent ratings. Subsequently, all of the interviews were analyzed by at least one of the researchers. The findings were summarized continuously throughout the analysis. The emergent model was challenged and modified through the inclusion of each additional interview.

The second source of information to build the model is the literature on software development and traditional product development. Whereas the eSoft case study data revealed the basic constructs for the analysis, the literature, as viewed through the lens of the case data, was also used to understand the software product development environment. An iterative process was used comparing the literature and the case to crosscheck the findings in each area.

The software development literature has a strong emphasis in the area of custom development; however, several recent studies have examined product issues (e.g., Hanssen and Faegri 2006). Taken together, the custom and product software researches have many insights to offer the current study, but it is clear that the area of software product development is still a relatively new area for research. Some of the holes in the software product development literature can be filled from the rich research stream in traditional product development (Nambisan and Wilemon 2000). Although the integration of these disparate research sources may not be immediately obvious, when viewed through the lens of eSoft's experiences, a pattern does emerge.

Site Selection

The company selected for the case study was an entrepreneurial software development company named here as eSoft. A previous study of eSoft (Hevner, Collins and Garfield 2002) had examined the tradeoffs between custom projects and software products. The current study focuses more exclusively on the emerging product development process that was used for the core product.

At the time of the study, eSoft was in a transition from a purely custom development company to a company that sold customized solutions based on a core software product. eSoft had been in this transition phase for over one year and the upheaval created by the shift to a product-orientated solution is apparent throughout the case. Each interviewed employee revealed issues related to the evolving process. eSoft was clearly not chosen because it is a product development exemplar; rather its value to this analysis is that it was in the midst of defining its product development environment. As a result, the underlying issues and tensions were visible throughout the interviews. In contrast, at a mature product organization this product development knowledge may have become tacit and may have been less accessible.

The model's evolutionary development in five major iterations is explored in next sections.

FIRST ITERATION - UNDERSTANDING THE PRODUCT CONCEPT

Software product development is a highly uncertain environment, and the traditional answer to uncertainty is an organic organization (Robey 1986). This type of organization has ambiguous responsibilities, defines jobs broadly, and relies on capable employees to achieve success.

At eSoft, however, it is not just reporting relationships that are 'organic' and ambiguous. Employees espouse personal visions of the products, but there is no evidence of a common vision. The following quote from marketing demonstrates the lack of a product vision:

> "I'd be told, … can we have a collateral piece [for a new product] … I never had a design spec or requirements documents, anything from which to build this collateral piece. So what I'd typically do is kind of fantasize what I think it should be … "

Not only is there no shared understanding of the product concepts, but even the corporate vision appears to be missing.

> "It all maps back to the CEO vision about what do you want to be when you grow up. What's our business, what should it be, and how do we get there and I think from top to bottom there's a great amount of consternation and confusion and concern over that."

It is clear that eSoft has an unstructured approach. The organizational structure is fluid, and the vision is a self-constructed reality that is different for each observer˙ However, a software development team without guidance becomes a code-and-fix development team, and in many cases this means that disaster is imminent (McConnell 1996). The eSoft employees are not blind to these issues. There is a lot of discussion about the uncertain environment. One employee talks about the difficulty of bringing in new employees.

> "Right now, the project relies on the expertise and experience of a single individual. And you could not, as a junior person in the group, hope to get anything positive out of it."

The literature supports the finding that organic solutions are not as effective as previously believed. Consider this quote from Eisenhardt and Tabrizi (1995):

> "… we call into question the traditional link between organic processes and uncertain situations. … organic processes fail to capture the importance of focus and structure that emerges here."

While the case doesn't speak to a specific resolution of these issues; there are some hints from the existing literature. On the surface the product and software research streams report similar findings. Both streams of research emphasize the need to manage rapidly changing markets. The characteristics of these markets are short product cycle times, quickly moving competition, and rapid technological change (Cusumano and Yoffie 1999). From the product development research come suggestions of an adaptive development approach using small teams to quickly probe the market space (Eisenhardt and Tabrizi 1995; Brown and Eisenhardt 1997). Likewise, the software development literature offers the thought that a large project can be broken into small teams that can adapt to changing needs (Cusumano 1997; Cusumano and Yoffie 1999).

The product research stream recommends multiple teams working in parallel on different versions of the product (Krishnan and Ulrich 2001). These versions, called market probes, are tests to map the market space and identify the most attractive product concept. A market probe might be a quick prototype shown at a trade show, a custom solution for a lead customer, a competitive analysis, or even a market projection by a futurist. On the other hand, the software articles describe a serial development approach: only one concept is under development. However, the development team is not locked-in to the concept. They are allowed the freedom to modify the product during the development cycle as new information becomes available.

A further reading of the literature indicates that the parallel versus adaptive-serial distinction is common when comparing traditional product research versus the software literature. Product development articles favor the multiple parallel probe approach (Dahan and Mendelson 2001; Krishnan and Ulrich 2001). Furthermore, once product probes are completed, the product development literature has less to say about flexibility in design. On the other hand, the software literature advocates adaptive development approaches in projects (Boehm 1988; Baskerville and Stage 1996). In most cases, the development of the concept or vision is ignored, or only briefly mentioned.

This leads to a question of whether there are fundamental forces differentiating the traditional product approach from the software product approach. Let's consider the heritage of the two research streams.

Traditional Products: Manufactured goods often have a large capital cost structure. At some point in the development cycle, a company must invest in manufacturing tooling to produce the proposed product. Once this capital investment is made, changing course is expensive. Market probes let these companies consider different concepts before making an investment. As a result the cost structure does not allow adaptability at the later stages, and requires a correct decision upfront.

Software Products: The software development literature derives its heritage from research on custom development products where the customer hands the developer the general development vision. Even the product-orientated software research (Carmel and Becker 1995; Cusumano and Yoffie 1999), assumes the development team begins with a product vision. However, once the vision is determined, these studies deal with the reality that software development is intrinsically more adaptable than manufacturing. The research concentrates on methods to manage that flexibility.

Combining both sets of literature brings new insights to software product development. The manufactured product approach would recommend a wide set of market probes to select the best product vision. However, those probes do not have to lock-in every product detail. The software literature recommends an adaptable approach that evolves as the development team learns the product and as the market itself evolves. However, the software literature also describes the need for controls to contain this adaptability. The hybrid approach is illustrated in Figure 1.

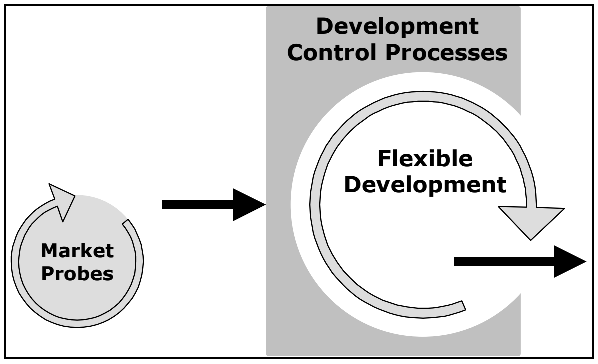

Figure 1. Hybrid Approach: Market Probes inform Flexible Development

SECOND ITERATION – TRANSITION FROM PRODUCT CONCEPT TO PRODUCT DEVELOPMENT

The initial model reveals several questions that can be asked of the case study data. Are eSoft's current projects market probes or flexible development projects? Is there a clear product concept entering into the development project stage? The interviews do show that product probes are being conducted. One of the longtime senior developers noted that:

> "For the last four years we've gone to the trade shows with prototypes and things."

But prototypes are not the only market probes that eSoft conducts. Market probes can be any initiative that learns about the market. For example, implementation instances and competitive analyses can help probe the market space. Following are some of the types of probes found in the eSoft case:

- **Customer Related Projects:** ongoing projects for customers help the company understand the market from the customer's view.
- **Product Technology Explorations:** a balance of organically grown technology and newer, standards-based efforts are used, such as EDI and XML.
- **Partnerships:** operating with trading partners who help introduce eSoft to corporate purchasing departments.
- **Prototyping:** eSoft has developed a prototype of a service-based product to gather prospect feedback.
- **Futurists:** The lead Architect, CTO, CIO, VP-Sales, VP-Development, CEO, a Marketing Strategist and a Product Manager all scan the industry with an eye towards developing the future vision.

Yet with all of these efforts underway, there does not appear to be a clear product concept emerging. One developer described the situation in comparison to other companies he had worked for:

> "And I think one of the things that I saw missing … is the why, the business sense of what it should accomplish. What are the real goals? … those meetings were turning into decisions on implementation. … Do we really know what type to build?"

This feeling of confusion seems to be common throughout the company. Even with many different probes no product concept seems to be emerging. Two indicators to the problem are mentioned in the interviews. In one case a senior manager is discussing changes in the product direction:

> "CTO will be out there pitching in front of a partner or customer, something like that, and he'll get some feedback. And then he'll use that single point to say, … we've got to go totally disrupt everything we're working [but it may only] be a specific need for that particular customer"

And another employee noted that:

> "Before it's been conflicting strategies coming from CEO, coming from CTO, coming from Architect, coming from VP Sales and whoever it might be. And there wasn't anybody pulling them together, getting them in sync, and making sure everybody agreed."

The problem appears to be that all of these market probes are independent initiatives. There is no effort to gather together the learning and establish a single product direction. The sense that comes from the literature is that the market probes should work together to describe the market space. Whereas in this instance it appears that there is no effort to integrate the learning.

The literature contains guideposts to a potential solution. First, the presence of a strong, autocratic leader is suggested in high velocity environments (Bourgeois and Eisenhardt 1988; Eisenhardt and Tabrizi 1995). How strong should this person be at eSoft? The CTO, VP-Sales, VP-Marketing, and VP-Development are already participating in the processes; often with confusing results. Yet despite this executive involvement there appears to be a general disconnect between product strategy and corporate strategy.

> "However, I don't think we've done a very effective job as to fostering those ideas and putting them down to paper and mapping those against the overall corporate vision and strategy"

The leader needs to be someone who can put the various executives on the same page; who can establish a strong tie between corporate strategy and product strategy; and who can act autocratically in the interest of speed. Taken together, these factors suggest a more central role for the CEO, but does he have the time to devote to this process? It is suggested (Bourgeois and Eisenhardt 1988) that although general strategic decisions should be made by a powerful, autocratic leader, *execution* decisions should be delegated to other team members. Since the execution workload is delegated, the eSoft CEO could conceivably play the role as the strong leader. Alternatively, the company could hire a COO who may provide a better match in terms of time available to the process.

A strong leader is not the only factor that helps pull together disparate market probes. The other recommendation is for "rhythmic time-paced transitions" (Brown and Eisenhardt 1997). There should a clear demarcation between the search for a product concept and the start of development. At this demarcation, the company should formally gather evidence from all of the market probes and consolidate its learning into a single product concept. This transition should not only be scheduled; it should be rhythmic, i.e. the transitions should have predictable intervals. For example, a company might schedule new product releases every September, preceded by a concept lock-down every January 31. The entire company automatically understands to gear its activities for a January development kickoff. Similarly, in the software product literature, Jordan and Segelod (2006) argue for formal coordination of knowledge collection at project inception, as well as sharing of valuable external linkage knowledge.

Another factor in using market probes effectively is the need to communicate the chosen product concept. One difficulty in new product development occurs because various team members do not share common vocabulary or goals (Carlile 2002). Therefore, boundary-spanning objects that establish a common vocabulary and common goals need to be created, and communicator and integrator leadership of the team is appropriate (Nambisan and Wilemon 2000). The role of the product concept is to summarize the learning into a form that is actionable by the development team, and that locks-in the agreed-to product direction.

As a result of these thoughts, our picture of the product development model has come into clearer focus as seen in Figure 2. The bottom line is that there must be a conscious effort to manage the independent market probes and to push them to a conclusion. Development of a coherent transition process from product concept to product development appears to be missing from eSoft. There needs to be a boundary-spanning product concept that grounds and informs the development team.

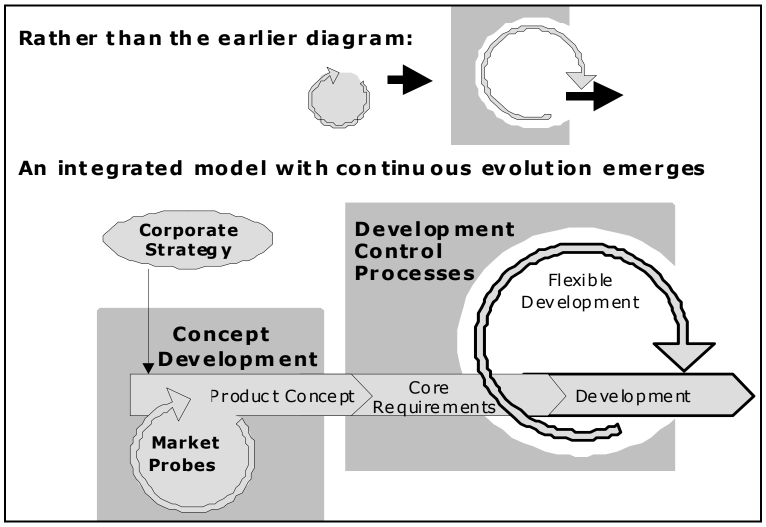
Rather than the earlier diagram:
An integrated model with continuous evolution emerges
Corporate Strategy
Development Control Processes
Flexible Development
Concept Development
Product Concept
Core Requirements
Development
Market Probes

Figure 2. Transition Process from Product Concept to Flexible Development

THIRD ITERATION – BUILDING REQUIREMENTS

eSoft has been making progress in getting its product concept process under control. However, all of eSoft's problems are not solely related to the lack of a product concept. One of the most common employee concerns at eSoft is the lack of a requirements document. In fact, the word "requirement" is used over 500 times in the interviews. Obviously, the requirements and the product concept are associated and fixing the concept selection will help the requirements generation.

However, the issue is too important to the process to assume that product requirements will magically spring from the concept document. The generation of requirements is central to all of the software development processes. Although some of the flexible development processes (Boehm 1988; Cusumano and Yoffie 1999; MacCormack, Verganti, and Iansiti 2001) allow requirements to unfold over time, the generation of requirements is of primary concern. There needs to be a clear linkage between the concept and the requirements and a clear evolution from broad requirements to detailed specification. The case evidence has already established that there is no shared product vision. However, even the development of detailed requirements appears to be an unmanaged process. One newer employee described his requirements experience as follows:

> " … from the time I came in October I've already been assigned to [produce] design documentation. … I did not have much on my plate. … it was quite a bit of time to spend … but there was not much of a context in doing it. "

As opposed to this somewhat haphazard process, requirements should unfold from initial concept to product concept to requirements, and finally to design. This continuous development of requirements grounds the entire development process. The product development will be flexible, but it will spring directly from the product concept explicated through the previous stage. Given the concept of a continuous evolution there is a need for a 'shepherd' to manage the evolution from concept to specification. For example, Hanssen and Faegri (2006) describe a software development company's move from a non-iterative waterfall-like process to a more evolutionary, agile development process to overcome previous problems from getting customer feedback late in the development process.

FOURTH ITERATION – FINDING THE SOFTWARE PRODUCT DEVELOPMENT CONTROL POINTS

In the analysis of the eSoft case study we identified several key control points essential for software product development. Here we briefly review the case study data and literature on these control points.

Software Architecture

One of the most important pieces of the discipline is the development of an architectural design that allows flexibility while establishing an underlying structure that insulates software components from changes and problems with other components. The architecture becomes central to the entire development structure. The architecture describes the interfaces for the components and it describes how the pieces fit together. However, the architecture should not over-specify the functionality of individual components. Each component team is allowed to evolve their component as long as they retain compatibility with the architectural interface.

eSoft has a lead architect but his role appears to be more one of a technical product planner. It appears that the eSoft architect is concerned with individual implementations and features, not with the component backbone for the software:

> "I don't know why [Architect's] worried about individual implementations … [unless] they impact the architecture"

Although eSoft has tried to launch a common architecture effort it does not appear to be productive:

> "Six months ago they formed an architecture team but they only had one meeting"
>
> "We … make … changes to architecture [for each of] these custom solutions right now."

Testing

Testing is an essential point of control during software development: unit testing, integration testing, and field testing. In software products, the field test with a single customer must be transformed to a market test that considers the needs of the market. Companies often do not understand the distinction between field and market testing and may overreact to field test data from a single customer (Day, Dougherty, and Adams 1998).

At eSoft, there is no evidence of problems at the unit testing level. But problems do occur at the integration testing level. A quality assurance analyst states that:

> "… we have a turnover from development to quality assurance … And it's been to the point before where we've had something that wouldn't even compile."

eSoft conducts its testing at the end of the development cycle, and problems are typically discovered late in the process. There is no evidence of market testing during development. In contrast, the flexible development literature suggests integration and market testing throughout the development cycle (Boehm 1988; Baskerville and Stage 1996; Cusumano and Yoffie 1999). The periodic tests not only look for programming flaws but they also insure that the flexible processes do not lose track of the dynamically changing market.

Project Planning

Timeline goals are recommended for all software development projects. However, at eSoft, timelines are only mentioned in relation to specific customer deliverables. It appears that no clear timelines for the development of product-specific artifacts exist.

Standards

Standards are another important control on adaptable development. These can include everything from programming languages to process and documentation techniques. At eSoft a clear policy on standards does not exist. One programmer mentions using an alternative language (COBOL) because he is more familiar with it. Of course, this will result in support problems for other employees who may not be familiar with COBOL. Without standards maintenance, testing and handoffs become much more difficult.

FIFTH ITERATION – BUILDING THE FINAL SOFTWARE PRODUCT DEVELOPMENT MODEL

Each of the areas mentioned in the previous sections are important considerations for development, and they can provide essential controls for the flexible development process. Unfortunately without some common vision, the various control forces may become conflicting demands fighting for dominance over the development team.

This is the problem observed at eSoft where each individual self-constructed their own product vision, and these visions were often in conflict with one another. The leap from general corporate strategy to specific product concept was too large. Although the team was expected to derive its own product concept, it needed a defined starting point. We have indicated this as the core value proposition. The core value proposition is a vision of the broad priorities for the target market space. Although it does not define specifics of the product it does prioritize the areas that need to be investigated first through market probes. The final model that emerges from our analysis is shown in Figure 3.

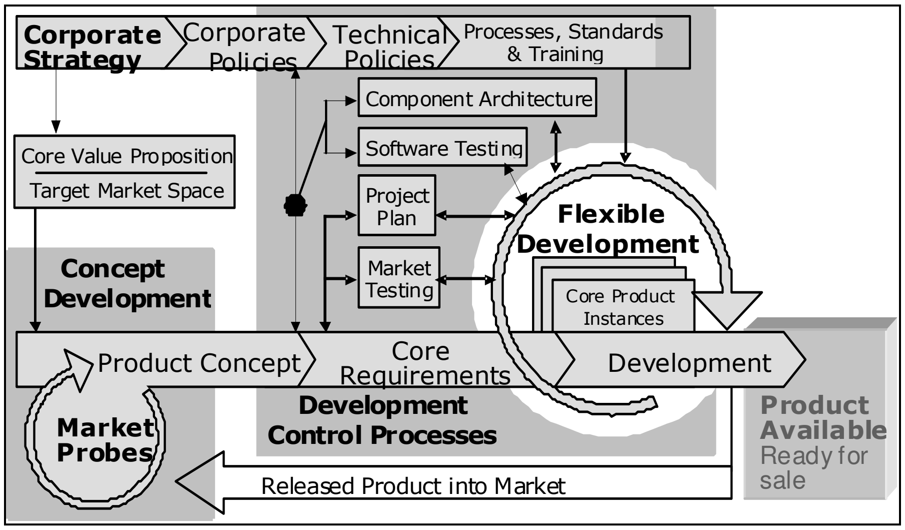

Figure 3. The Emergent Model of Software Product Development

This model describes the key journey through the product development cycle. The cycle includes three distinct types of activities: Concept Selection, Development Control Processes; and Flexible Development.

Concept Development is a managed set of market probes intended to map the possible market space. The output of the process is a product concept that spans the boundary into the next stage of development.

Development Control Processes are artifacts that guide the flexible development process so that it remains focused. The guidance is intended to insure that development delivers against the target product concept; therefore, these artifacts are simply further boundary spanning items that translate the concept into other terms that are familiar to the developers. However, the controls are also informed by any technical policies that span the entire company. The relationships between the specific control processes and development are interactive (two-headed arrows), indicating how changes during flexible development require revisiting processes like changes to the plan or additional testing.

Flexible Development contains the actual development processes. Many different approaches could be used within the development teams; however, the limiting factors are the initial requirements and the other development control processes.

The primary guiding force in this model comes from the evolving product concept. However, this concept and the development controls should also be informed by any common corporate requirements.

DISCUSSION AND FUTURE RESEARCH DIRECTIONS

Significant attention has been given to the issue of software development processes and flexible development techniques. In the new product development field there have been important research findings on effective approaches for new product development. However, there has been less attention given to full life cycle product development processes. Both traditional product development and software development processes offer insights, but neither literature stream gives a complete picture of the software product development process. One exception is the "Four Cycles of Control" framework, which also seeks to organize and identify needed controls in software product development, while also accommodating flexibility and responsiveness to a changing environment. The framework is, however, at a higher level, and focuses on the type, timing and content of product releases (Rautiainen et al. 2002).

Our case study on eSoft has confirmed the importance of these issues. eSoft has implemented a dynamic, loosely structured development environment, but they are struggling to transition to a product-based solution. It is clear that more than an organic, freely evolving approach is required. The interviews indicate that the missing element is the focus that would be supplied by better control structures. Implementation of a two-phased probing/developing process as shown in Figure 3 would help direct their efforts. Additionally, the establishment of control initiatives would direct and place boundaries on their development while still retaining needed flexibility. These control initiatives must be informed by the underlying product concept. Without grounding in a common concept vision they will become opportunities for confusion rather than direction.

The interviews at eSoft caught the company at a time of transition as they dealt with a change from custom development to product development. At the time of the study, their goal was not to abandon custom support, but to launch a hybrid approach with a common core product and a custom integration layer for every customer. However, their existing control structures remained focused on the custom environment. Outcome-based controls on customer deliverables focused attention on meeting customer needs at the expense of developing a stable product solution for all customers. The value of this environment to researchers was that it provided easy access to issues that need to be addressed in developing new product development processes.

The next steps of this research will be an effort to validate and generalize the findings. Future initiatives will include multiple cases to compare the product development at other companies, and, possibly, a broad-based survey to look for general trends across companies. Finally, this research area has considerable room for subsequent extensions. Follow up research can concentrate on the two-phased nature of development, the design and use of market probes, and the control processes around flexible development processes.

ACKNOWLEDGEMENTS

The authors acknowledge the openness and cooperation of the software managers and developers at eSoft. The paper benefited from insightful comments of the reviewers.

REFERENCES

1. Baskerville, R. and Stage, J. (1996). "Controlling Prototype Development through Risk Analysis." *MIS Quarterly* **20**(4): 481-504.
2. Boehm, B. (1988). "A Spiral Model of Software Development and Enhancement." *IEEE Computer* 61-72.
3. Bourgeois, L. and Eisenhardt, K. (1988). "Strategic Decision Processes in High Velocity Environments: Four Cases in the Microcomputer Industry." *Management Science* **34**(7): 816-835.
4. Brown, S. and Eisenhardt, K. (1997). "The Art of Continuous Change: Linking Complexity Theory and Time-paced Evolution in Relentlessly Shifting Organizations." *Administrative Science Quarterly* **42**: 1-34.
5. Carlile, P. (2002). "A Pragmatic View of Knowledge and Boundaries: Boundary Objects in New Product Development." *Organization Science* **13**(4): 442-455.
6. Carmel, E. and Becker, S. (1995). "A Process Model for Packaged Software Development." *IEEE Transactions on Engineering Management* **42**(1): 50-61.

7. Cusumano, M. (1997). "How Microsoft Makes Large Teams Work Like Small Teams." *MIT Sloan Management Review* **39**(1): 9-20.

8. Cusumano, M. and Yoffie, D. (1999). "Software Development on Internet Time." *IEEE Computer* 60-69.

9. Dahan, E. and Mendelson, H. (2001). "An Extreme-Value Model of Concept Testing." *Management Science* **47**(1): 102-116.

10. Day, G., Dougherty, D. and Adams, M. (1998). "Enhancing New Product Development Performance: An Organizational Learning Perspective." *Journal of Product Innovation Management* **15**: 403-422.

11. Eisenhardt, K. (1989). "Building Theories from Case Study Research." *Academy of Management Review* **14**(4): 532-550.

12. Eisenhardt, K. and Tabrizi, B. (1995). "Accelerating Adaptive Processes: Product Innovation in the Global Computer Industry." *Administrative Science Quarterly* **40**: 84-110.

13. Hanssen, G.K. and Faegir, T.E. (2006). "Agile Customer Engagement: A Longitudinal Qualitative Case Study." *ISESE'06*, September 21-22 2006, Rio de Janeiro Brazil, 164-173.

14. Hevner, A., Collins, R., and Garfield, M. (2002). "Product and Project Challenges in Electronic Commerce Software Development." *Database for Advances in Information Systems* **33**(4): 10-23.

15. Jordan, G. and Segelod, E. (2006). "Software Innovativeness: Outcomes on Project Performance, Knowledge Enhancement, and External Linkages." *R&D Management* **36**(2): 127-142.

16. Krishnan, V. and Ulrich, K. (2001). "Product Development Decisions: A Review of the Literature." *Management Science* **47**(1): 1-21.

17. MacCormack, A., Verganti, R. and M. Iansiti (2001). "Developing Products on "Internet Time": The Anatomy of a Flexible Development Process." *Management Science* **47**(1): 133-150.

18. McConnell, S. (1996). *Rapid Development*. Redmond, Washington, Microsoft Press.

19. Nambisan, S. and Wilemon, D. (2009). "Software Development and New Product Development: Potentials for Cross-Domain Knowledge Sharing." *IEEE Transactions on Engineering Management* **47**(2), 211-220.

20. Rautiainen, K., Lassenius, C. and Sulonen, R. (2002). "4CC: A Framework for Managing Software Product Development." *Engineering Management Journal* **14**(2), 27-32.

21. Robey, D. (1986). *Designing Organizations*. Homewood, Illinois, Irwin.