2010

Moderating Effects of Requirements Uncertainty on Flexible Software Development Techniques

Werner Mellis
University of Cologne

Claudia Loebbecke
University of Cologne

Richard Baskerville
Georgia State University

Moderating Effects of Requirements Uncertainty on Flexible Software Development Techniques

Werner Mellis
Department of Information Systems and Systems Development
University of Cologne
mellis@wiso.uni-koeln.de

Claudia Loebbecke
Department of Business, Media and Technology Management
University of Cologne
claudia.loebbecke@uni-koeln.de

Richard Baskerville
Department of Computer Information Systems
Georgia State University
baskerville@acm.org

ABSTRACT

Partially due to increasing requirements uncertainty, flexibility has been in the focus of many software development activities for many years. Only few studies have analyzed the indirect effect that different levels of requirements uncertainty have on the effects of established flexible development techniques. This study analyzes how requirements uncertainty moderates the well studied effects of (1) sequential development, (2) investment in architectural design, and (3) intensity of early feedback on the performance of contract development projects. It finds that requirements uncertainty negatively moderates the effects of sequential development. It also points out that requirements uncertainty negatively moderates the effect of investment in architectural design. For agile development approaches, the value of investment in architectural design falls with increasing uncertainty. However, for plan-driven approaches, investment in architectural design is positive at any level of requirements uncertainty. Finally, the paper finds that early feedback throughout the development process is helpful at any level of requirements uncertainty.

Keywords

Software Development Projects, Flexibility, Requirements Uncertainty

INTRODUCTION

Flexible software development, also labeled iterative, incremental, or agile development, has been in the focus of software development activities for more than a decade (Beck, 1999; Blackburn, Scudder, and van Wassenhove, 2000; Cusumano and Selby, 1995; Cusumano and Yoffie, 1998; Larman and Basili, 2003). It commonly refers to a set of specific software development approaches such as short development cycles, different kinds of prototyping, parallel development stages. The impact of these and other approaches on the performance of the development efforts has frequently been investigated and shown to be successful (Baskerville, Ramesh, Levine, Pries-Heje and Slaugther, 2003; Burns and Dennis, 1985; Davis, 1982; El Louardi, Galletta and Sampler, 1998; MacCormack, Verganti and Iansiti, 2001; MacCormack, Kemerer, Cusumano and Crandall, 2003; Mathiassen, Tuunanen, Saarinen and Rossi, 2007; Rai and Al-Hindi, 2000).

Another body of literature demonstrates the value of flexible software development in projects facing uncertainty (Baskerville et al., 2003; Baskerville and Pries-Heje, 2004; Beck, 1999; Beynon-Davies, Tudhope and Mackay, 1999; Bhattacharya, Krishnan and Mahajan, 1998; Boehm, 1988; Hardgrave, Wilson and Eastman, 1999; Krishnan, Eppinger and Whitney, 1997; Mahmood, 1987; Moynihan, 2000a). The sources typically point to the negative effect that especially requirements uncertainty has on software development performance. Many of the respective theoretical models rely on the trade off of benefits from delaying commitment to certain parts of a development against the costs that such delays impose on a project. They employ a static optimization perspective, with an assumption that the costs of various actions are out of the immediate control of the development team.

In recent years, agile development has risen to become a prominent form of flexible software development. Adapted from agile manufacturing (Baskerville, Mathiassen and Pries-Heje, 2005), and initially pronounced in the Agile Manifesto (2001), this family of methods aimed at rapid adaptation to change (Highsmith, 2002). While often characterized by their lightweight structures, user involvement, operational software, etc., definitions of agile methods vary substantially (Conboy, 2009). Indeed, there is now a pronounced need to define and study agility and agile methods, especially beyond the adoption stage (Abrahamsson, Conboy and Xiaofeng, 2009; Dyba and Dingsoeyr, 2008). Since our interest lies in the study of one particular feature across a wide variety of flexible methods and techniques, we consider a broader range than just agile methods. Such a range would include, for example, prototyping as a technique used within other methods as well as prototyping as a systems development method in its own right (Alavi, 1984).

So far, only a few studies (Hardgrave et al., 1999; Hsu, Chan, Liu and Chen, 2008; Kraut and Streeter, 1995; MacCormack et al., 2001; MacCormack and Verganti, 2003) take the indirect route and analyze the effect that different levels of requirements uncertainty have on the effects of various software development techniques. Studying standard software product development, MacCormack et al. (2001) and MacCormack and Verganti (2003) focus their work on investigating three techniques, namely sequential development, investment in architectural design, and earliness of feedback. They show that those techniques have a strong impact on project performance in settings of generally high requirements uncertainty such as the dynamic Internet industry of the late 1990s. They also analyze the dependence (moderation effect) between requirements uncertainty and the impact of the three techniques in their effect on the software development project performance (MacCormack and Verganti, 2003). However, most of these studies examine the moderation effect of requirements uncertainty in the period leading up to, and during the 1990s Internet boom. Our research responds to the call for more “post-adoption” studies that examine flexibility within the constant and routine pressure to deliver adaptive software functional to changing marketplaces (Abrahamsson et al., 2009; Dyba and Dingsoeyr, 2008).

This paper contributes an empirical analysis of the moderating effects of Requirements Uncertainty (U) on the use of several software development techniques for the purpose of achieving software development Project Performance (P). Requirements Uncertainty refers to a situation in which deriving requirements specifications from customer needs is difficult (Harter, Krishnan and Slaughter, 2000; Krishnan, Kriebel, Kekre and Mukhopadhyay, 2000). The specific software development techniques under study here are (1) Sequential Development (S), (2) Investment in Architectural Design (A), (3) Intensity of Early Feedback (F). Our empirical context is set in contract software development projects, and we study only low to medium levels of Requirements Uncertainty.

Contract software development project presumably differ from standard software development projects in at least two ways: Firstly, in contract software development projects, development costs are likely to be less relevant as they are shared by many contract partners. Secondly, in contract development projects, it is generally essential for 'being successful' to explicitly fulfill the contract requests. In standard software development projects, however, which develop software for barely predefined customers but 'to the shelf', it is typically crucial to be first on the market with sufficient quality in order to be successful.

The remainder of the paper is structured as follows: In the next section, we introduce our research model for investigating how requirements uncertainty moderates the effects that the use of sequential development, investment in architectural design, intensity of early feedback have on the performance of contract software development projects; we develop our research hypotheses and outline the data collection approach followed. We then go into some detail of our results and the subsequent statistical analysis, before we present our findings and put them into perspective to earlier research. We conclude with some implications for practice and theory development.

RESEARCH APPROACH

Research Model

Our research model for studying contract software development projects is built on the widely held beliefs that sequential development, investment in architectural design, and intensity of early feedback positively impact project performance (MacCormack et al., 2001, Sanchez and Mahony, 1996). It lets us analyze how different levels of Requirements Uncertainty (U) moderate those effects, i.e., the effects that the use of Sequential Development (S), Investment in Architectural Design (A), and Intensity of Early Feedback (F) have on software development Project Performance (P). See Figure 1.

P = P(U,S,A,F,U*S,U*A,U*F)
with
P = Project Performance
U = Requirements Uncertainty
S = Sequential Development
A = Investment in Architectural Design
F = Intensity of Early Feedback
For the modeling of the interaction terms
U*S, U*A and U*F see Appendix A

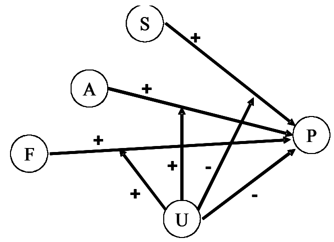

Figure 1. Research Model

To operationalize the variables included in our research model, we took a multiple step research approach in order to design an adequate survey instrument. Following Borsboom and Mellenbergh (2004), Diamantopoulos and Winklhofer (2001), and Straub et al. (2004), we first reviewed the literature on searching for adequate formative and reflective measures (Jarvis, Mackenzie and Podsakoff, 2003) of Project Performance (P), Requirements Uncertainty (U), Sequential Development (S), Investment in Architectural Design (A), and Intensity of Early Feedback (F). We then conducted interviews with practitioners in order to learn about any content discrepancies of the respective items. Based on content validity (Rossiter, 2002) and pilot testing with 'think aloud' protocols, we were able to show that we did neither experience comprehension problems nor disruption of results. To this end, we let project managers assess each of our items on a Likert scale ranging from '1 = do not agree at all' to '5 = fully agree'. (See Appendix 1 for more details on the construct measures.)

Project Performance (P) is a complex and multi-component variable measured along different dimensions in the literature. For instance, Andres and Zmud (2001-2002), Faraj and Sproull (2000), Gemino et al. (2007), Nidumolu (1995), Nidumolu (1996), and Rai and Al-Hindi (2000) distinguish product and process dimensions of project performance. Henderson and Lee (1992) consider effectiveness, efficiency, and time as project performance dimensions. DeLone and McLean (1992), DeLone and McLean (2003), and Ives et al. (1983) define a well established model of product performance.

Requirements Uncertainty (U) is well known in literature and practice (Jones, 1996). Numerous works (Davis, 1982; Fazlollahi and Tanniru, 1991; Hsu et al., 2008; Kraut and Streeter, 1995; Mathiassen and Pedersen, 2008; Mathiassen and Stage, 1990; Moynihan 2000a; Moynihan 2000b; Nidumolu 1995) investigate the effects of requirements uncertainty on software development project performance and/or analyze ways to cope with requirements uncertainty in software development projects. Because requirements uncertainty involves difficulty in requirements specification, any step by step procedures that follow face equal difficulties. More flexible processes, which systematically manage experimentation (Thomke, 1998), can be advantageous.

Sequential Development (S) of software here refers to the intended sequentiality in contrast to the factual one, i.e., to the intent that the various development phases like requirements specification, design, coding, testing overlap (in time). Highly sequential development means that the different development phases are intended to be performed one after the other with hardly any overlap in time. Pursuing a highly sequential development was promoted early by Royce (1970). Highly sequential projects have gained widespread acceptance in practice and have become the basis for many software acquisition standards in government and industry. Further, they are commonly applied in contract software development projects (Boehm, 1988). A low degree of sequential development means that the different development phases are intended to significantly overlap in time.

Investment in Architectural Design (A) often implies building modules or layers in order to separate stable code from code that is expected to be subject to later changes (MacCormack et al., 2001). Architectural design must be optimized under criteria such as process flexibility or code sensitivity and performance, which are usually not compatible. Thus, optimizing the architecture for process flexibility demands greater investment in architectural design.

Early Feedback can for instance be achieved by prototypes or a beta release. MacCormack et al. (2001) specifically measure earliness of the feedback, which they define as the percentage of a product's final functionality (or features) that is included in the prototype, the first built or the beta release. However, in our contract software development projects, the granularity of documented functions or features varies considerably between projects as well as within projects. Thus it remains unclear whether a counted or estimated percentage can be a meaningful measure.

Research Hypotheses

We expect that the positive effect of a higher degree of Sequential Development on Project Performance will decrease with increasing Requirements Uncertainty. We also expect that at higher levels of Requirements Uncertainty less Sequential Development, i.e., less overlapping of different development phases, is advised. We state (see Appendix 2 for formal expressions for these research hypotheses):

- H1a: The effect of Sequential Development (S) on Project Performance (P) is moderated by Requirements Uncertainty (U).
- H1b: With increasing Requirements Uncertainty (U), the effect of Sequential Development on Project Performance (P) decreases.
- H1c: The effect of Sequential Development (S) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U).

Further, we assume that higher Investment in Architectural Design allows reducing the amount of rework effort caused by one unit of requirements change, i.e., that higher Investment in Architectural Design improves Project Performance (MacCormack et al., 2001) - an effect which may be moderated by Requirements Uncertainty (as shown for standardized software development projects by MacCormack and Verganti (2003)). The question remains whether the effect is also moderated by Requirements Uncertainty in the case of contract software development. Hence we propose for testing:

- H2a: The effect of Investment in Architectural Design (A) on Project Performance (P) is moderated by Requirements Uncertainty (U).
- H2b: Increasing Investment in Architectural Design (A) by the same amount yields a greater improvement of Project Performance (P) with increasing Requirements Uncertainty (U).
- H2c: The effect of Investment in Architectural Design (A) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U)

Finally, we assume that Intensity of Early Feedback improves Project Performance (MacCormack et al., 2001) -- again, an effect which may be moderated by Requirements Uncertainty (as shown for standardized software development projects by MacCormack and Verganti (2003)). Similar to H2, the question remains whether the effect is also moderated by Requirements Uncertainty in the case of contract software development. Hence we propose for testing:

- H3a: The effect of Intensity of Early Feedback (F) on Project Performance (P) is moderated by Requirements Uncertainty (U).
- H3b: Increasing Intensity of Early Feedback (F) by the same small amount yields a greater improvement of Project Performance (P) with increasing Requirements Uncertainty (U).
- H3c: The effect of Intensity of Early Feedback (F) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U).

Data Collection

The data collection was supported by a leading national professional organization for project management ("GPM Deutsche Gesellschaft für Projektmanagement e.V."). The organization has about 3,600 members from all branches, about 500 of which are software project managers. Our call addressed software project managers, who recently had finished a software development project they were responsible for. We required that the project was concerned with the development of contract commercial software. We also required that the project was finished within 3 to 18 months before filling in the questionnaire so that details of the project were vividly remembered and the project results had been available long enough to allow a reliable evaluation of Project Performance. Aiming at rather homogenous contract software development projects, we excluded projects developing gaming software, embedded, or security-centered software. We also excluded software projects undertaken for public institutions.

Data were collected in two phases. In the first phase, the call for participation was sent out with the organization's newsletter in October 2005. Projects to be included into the study should have been finished between July 1, 2004 and June 30, 2005. Upon receiving the newsletter, 146 project managers declared their interest to participate with one project each. We contacted all 146 to check whether the project to be reported fulfilled the requirements as outlined above. In that process, we excluded 26 projects which did not meet the requirements. The remaining 120 project managers received a self-report form, 32 of which were completed and returned (response rate of 27%).

We sent out a second call for participation in February 2006, in which 197 project managers declared to be interested. Based on the same procedure as before, another 162 self-report surveys were sent out; 56 were completed and returned (35%). As we contacted all project managers individually, we could avoid duplicates. For the then total of 88 filled out questionnaires, we double checked again the project description. As there was some doubt that the project belonged to the intended type or was identified properly, we excluded another 20 data sets from further analysis. At the end of the data collection, we had 68 data sets for the final analysis, for which we will report the results below. Sample sizes of comparable studies are Hardgrave et al. (1999): 133; Harter et al. (2000): 30; Krishnan et al. (2000): 43; MacCormack et al. (2001): 29; MacCormack and Verganti (2003): 29.

RESULTS

Table 1 summarizes the data of our sample.

Variable	Description	N	Mean	SD	Min.	Max.
P -- Project Performance						
P1	User satisfaction	**68**	**3.72**	**0.75**	**2**	**5**
P2	**Adherence to budget**	**68**	**2.91**	**1.32**	**1**	**5**
P3	**Adherence to schedule**	**68**	**3.12**	**1.40**	**1**	**5**
P4	**Efficiency**	**68**	**3.35**	**0.99**	**1**	**5**
U -- Requirements Uncertainty						
U1	**Derivable without problems**	**68**	**2.87**	**1.05**	**1**	**5**
U2	**Known procedures**	**68**	**2.59**	**1.00**	**1**	**5**
U3	**Easy to understand**	**68**	**3.01**	**1.01**	**1**	**5**
S -- Sequential Development						
S1	**Intention to separate analysis and design**	**68**	**2.84**	**1.22**	**1**	**5**
S2	**Intention to separate design and coding**	**68**	**2.82**	**1.05**	**1**	**5**
S3	**Thorough gate keeping**	**68**	**3.41**	**1.22**	**1**	**5**
A -- **Investment in Architectural Design**						
A1	**Flexibility for change**	**68**	**3.74**	**1.02**	**1**	**5**
A2	**Ease of change**	**68**	**3.63**	**1.11**	**2**	**5**
A3	**Measures for flexibility**	**68**	**3.29**	**1.13**	**1**	**5**
F -- **Intensity of Early Feedback**						
F1	**Practical use of software**	**68**	**3.49**	**1.10**	**1**	**5**
F2	**Practical experience of interacting components**	**68**	**3.40**	**1.07**	**1**	**5**
F3	**Evaluations of prototypes or versions**	**68**	**2.76**	**1.34**	**1**	**5**

Table 1. Descriptive Statistics

Formative and reflective variables coexist in our research model (see Appendix 1 for details.) While Requirements Uncertainty (U), Investment in Architectural Design (A), and Intensity of Early Feedback (F) are reflective variables, Project Performance (P) and Sequential Development (S) denote formative ones. Thus, we adapted separate reliability and validity measures for each as presented below.

Reflective Variables: Reliability and Validity Measures

The reflective variables in our research model are Requirements Uncertainty (U), Investment in Architectural Design (A), and Intensity of Early Feedback (F). We calculate a Cronbach's Alpha of 0.76 for Requirements Uncertainty (U), 0.78 for Investment in Architectural Design (A) and 0.66 for Intensity of Early Feedback (F). Based on Nunnally's (1978) guidelines, we suggest that a score of 0.70 (0.80) or above is an acceptable value of internal consistency for exploratory (confirmatory) research.

Concerning composite reliability, we observe results generally above the threshold of 0.7 (Fornell and Larcker, 1981) as we find 0.92 for Requirements Uncertainty (U), 0.87 for Investment in Architectural Design (A), and 0.60

for Intensity of Early Feedback (F). An explorative factor analysis (Table 2) demonstrates that the three variables are single factored. All loadings are above 0.7. Therefore the variance shared between variables and their respective items is higher than the variance of the measurement error (Carmines and Zeller, 1979).

Concerning the discriminant validity of our reflective variables, we assess the item to variable correlations. The figures in Table 3 support discriminant validity of Requirements Uncertainty (U), Investment in Architectural Design (A), and Intensity of Early Feedback (F). Another criterion of discriminant validity demands that the square root of the average variance extracted for a reflective variable (Table 4) is greater than the correlation of that variable to other variables (Fornell and Larcker, 1981). The criterion is satisfied for our reflective variables.

Loadings on U		Loadings on A		Loadings on F	
U1	**0.911**	**A1**	**0.852**	**F1**	**0.809**
U2	**0.754**	**A2**	**0.769**	**F2**	**0.742**
U3	**0.793**	**A3**	**0.873**	**F3**	**0.775**

Table 2. Exploratory Factor Analysis Loadings for Reflective Variables [Requirements Uncertainty (U), Investment in Architectural Design (A), and Intensity of Early Feedback (F)]

	U	A	F
U1	0.911**	-0.048	0.315**
U2	0.754**	-0.249*	0.166
U3	0.793**	0.091	0.220
A1	-0.051	0.852**	0.268*
A2	-0.093	0.769**	0.147
A3	-0.057	0.873**	0.300*
F1	0.201	0.420**	0.809**
F2	0.147	0.225	0.742**
F3	0.324**	0.023	0.775**

(Legend: **p < 0.1%, *p < 1%)

Table 3. Item to Variable Correlations

	A	**F**	**S**	**U**	**P**
A	0.835				
F	0.448	0.772			
S	0.258	-0.042	..		
U	-0.050	0.116	-0.218	0.887	
P	0.374	0.293	0.241	-0.647	

Table 4. Correlations between Variables

Formative Variables: Reliability and Validity Measures

Although not expected, the formative variables -- Project Performance (P) and Sequential Development (S) -- may be characterized by their items revealing covariance. Therefore loadings and cross loadings to the variables as well as correlations and shared variances between the items and the formative variables cannot be interpreted as statements about variable validity (Diamantopoulos and Winklhofer, 2001; Jarvis et al., 2003). Therefore, we test for multi-collinearity (Table 5). The respective multi-collinearity for Project Performance (P) and Sequential Development (S) is not strong enough to justify deleting any one item from either one of the two variables. The maximum variance inflation factor (VIF) is 2.22, which is far below the common cut-off threshold of 10 (Chatterjee, Hadi and Price, 2000).

Regression	$R^2_{adj.}$	VIF		Regression	$R^2_{adj.}$	VIF
S				P		
S1=a S2+b S3	0.46	1.85		P1=a P2+b P3+c P4	0.12	1.14
S2=a S1+b S3	0.55	2.22		P2=a P1+b P3+c P4	0.42	1.72
S3=a S1+b S2	0.39	1.64		P3=a P1+b P2+c P4	0.39	1.64
				P4=a P1+b P2+c P3	0.21	1.27

Table 5. Testing for Multi-Collinearity

To check the validity of formative variables, we employed MIMIC models (details available upon request) and found the two formative variables to be valid.

ANALYSIS

Since formative and reflective measurement models coexist in our research model, we apply the partial least squares algorithm (PLS) for data analysis. We used smartPLS with standardized indicators (mean 1, standard deviation 0).

For using PLS, the sample size needs to be at least ten times the number of items in the most complex variable (Gefen, Straub and Boudreau, 2000; Marcoulides and Saunders, 2006), which in our case is 30. Hence, our sample size of 68 is sufficient.

Besides the effect of Requirements Uncertainty (U), Model I captures the direct and moderated effects of Sequential Development (S) and Investment in Architectural Design (A) on Project Performance (P). We did not compute models containing both Investment in Architectural Design (A) and Intensity of Early Feedback (F) as these two techniques are strongly correlated. Model II considers the direct and indirect effect of Intensity of Early Feedback (F) on Project Performance (P). Since Model II demonstrates an indirect, but no direct effect, we compute Model III to check whether assuming an indirect effect is warranted. In the three models, we enclose the direct effect of Sequential Development (S), Investment in Architectural Design (A), and Intensity of Early Feedback (F) on Project Performance (P) to avoid quasi-moderation effects.

Model I $P = a_0 + a_1 * U + a_2 * S + a_3 * A + a_5 * U * S + a_6 * U * A$

Model II $P = a_0 + a_1 * U + a_2 * S + a_4 * F + a_5 * U * S + a_7 * U * F$

Model III $P = a_0 + a_1 * U + a_2 * S + a_4 * F + a_5 * U * S$

Model I supports H1a, H1b, and H2a. The path coefficient for U*A is negative, rejecting hypothesis H2b. In Model II, the moderator of Intensity of Early Feedback (F) is not significant, rejecting H3a (rendering H3b and H3c irrelevant). In order to check whether Intensity of Early Feedback (F) is in fact significant, we delete U*F from the model. This led to Model III. Further the R^2 of Model III (R_{III}^2) is not significantly smaller than the R^2 of Model II (R_{II}^2) according to the F statistics. Thus, Requirements Uncertainty (U) does not moderate the effect of Intensity of Early Feedback (F) on Project Performance (P), while Intensity of Early Feedback (F) is significant in Model III. For the path coefficients for the three models computed over the full sample see Appendix 3.

In order to investigate hypothesis H1c and H2c, we ordered the sample according to the degree of Requirements Uncertainty (U) and divided it equally into four sub-samples, (1) highestU, (2) higherU, (3) lowerU, and (4) lowestU. For the four sub-samples, we compute a simple linear model

Model IV $P = a_0 + a_1 * U + a_2 * S + a_3 * A$

We find that the path coefficient of Sequential Development (S) is significant in all sub-samples (see Appendix 4) The coefficient of Sequential Development (S) for the sub-sample highestU is negative, supporting hypothesis H1c. The coefficient of Investment in Architectural Design (A) for all sub-samples is either positive or not significant. Therefore hypothesis H2c is rejected.

FINDINGS AND DISCUSSION

Table 6 summarizes the respective hypotheses tests:

H	Hypothesis	Result
1a	The effect of Sequential Development (S) on Project Performance (P) is moderated by Requirements Uncertainty (U)	Supported [Model I, Appendix 3]
1b	With increasing Requirements Uncertainty (U), the effect of Sequential Development (S) on Project Performance (P) decreases	Supported [Model I, Appendix 3]
1c	The effect of Sequential Development (S) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U)	Supported [Appendix 4]
2a	The effect of Investment in Architectural Design (A) on Project Performance (P) is moderated by Requirements Uncertainty (U)	Supported [Model I, Appendix 3]
2b	With increasing Requirements Uncertainty (U), the effect of Investment in Architectural Design (A) on Project Performance (P) increases	Rejected [Model I, Appendix 3]
2c	The effect of Investment in Architectural Design (A) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U)	Rejected [Appendix 4]
3a	The effect of Intensity of Early Feedback (F) on Project Performance (P) is moderated by Requirements Uncertainty (U)	Rejected [Comp. Model II and Model III, Appendix 3]
3b	With increasing Requirements Uncertainty (U), the effect of Intensity of Early Feedback (F) on Project Performance (P) increases	Not Relevant [since H3a is rejected]
3c	The effect of Intensity of Early Feedback (F) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U)	Not Relevant [since H3a is rejected]

Table 6. Results of Hypotheses Testing

Our analysis yields four main findings:

(1) *For contract development software projects, we find a significant correlation between the Investment in Architectural Design (A) and the Intensity of Early Feedback (F) as shown in Appendix 3.*

The significant correlation between the investment in architectural design and the intensity of early feedback might indicate that the two techniques require each other, i.e. striving for intensive early feedback is successful only if the investment in architectural design is high. However, it might also just point to the simple fact that many project managers just see both techniques to be applied jointly. Overall, our finding is in line with the findings of Baskerville et al. (2003), Baskerville and Pries-Heje (2004), and MacCormack et al. (2003). Although they partly investigate different techniques, they all underline that the selected techniques are applied in certain combinations.

(2) *The effect of Sequential Development (S) on Project Performance (P) decreases with increasing Requirements Uncertainty (U). Above some degree of Requirements Uncertainty (U), increasing Sequential Development (S) actually has a negative effect on Project Performance (P).*

According to our study, the generally known effect of highly sequential development on project performance decreases with increasing requirements uncertainty. Hence, in situations with high requirements uncertainty software developers should opt for less sequential development, i.e., for overlapping phases (see also Davis, 1982; Fazlollahi and Tanniru, 1991; Mathiassen and Stage, 1990; Saarinen and Vepsalainen, 1993).

(3) *The effect of Investment in Architectural Design (A) on Project Performance (P) decreases with increasing Requirements Uncertainty (U). In other words with increasing U, one needs higher Investment in Architectural Design (A) to achieve the same Project Performance (P). However - different from the situation concerning*

Sequential Development (S) - increasing Investment in Architectural Design (A) in our data set never leads to decreasing Project Performance (P).

Similarly to the above, we find that the effect that higher investment in architectural design has on the performance of contract software development decrease with increasing requirements uncertainty (Finding 3). In other words with increasing requirements uncertainty, one needs higher investment in architectural design to achieve the same improvement of project performance. This finding is in line with Baskerville and Stage (1996) who claim that preventive architectural design efforts (i.e., investment in architectural design) are often in vain and in total as high as any corrective rework thus prevented. It, however, also contradicts our expectation derived from MacCormack and Verganti (2003) on standard software development projects. We can speculate about explanations for this contradiction. For example, there are different project samples. Standard software developers hope to distribute the risk among many clients, rendering high risks from requirements uncertainty acceptable. The respective projects are often driven by early, flexible delivery and short response times (Cusumano and Selby, 1995); development costs are usually considered less relevant because of expected scale effects. Thus, higher investment in architectural design is more easily justified in standard software development projects (Iansiti and MacCormack, 1997). In contract software development projects, however, the risk caused by requirements uncertainty has to be covered by one client, this makes it less attractive to invest in architectural design in order to allow earlier delivery.

(4) The effect of Intensity of Early Feedback (F) on Project Performance (P) is not moderated by Requirements Uncertainty (U).

Finally, our research shows that the effect of intensity of early feedback on project performance is *not* moderated by requirements uncertainty. In other words, intensive early feedback increases project performance independent of the degree of requirements uncertainty. Comparing our finding to the results of MacCormack and Verganti (2003) is tricky. MacCormack and Verganti demonstrate that in samples with higher market uncertainty Early Market Feedback is significantly related with product performance, while in samples with lower market uncertainty it is not. They demonstrate differential validity, i.e., the significance of the regression coefficient of the interaction term increases with uncertainty (see Carte and Russel, 2003). They do not report results concerning the moderation effect of early market feedback in the sense of differential prediction, i.e., they do not state that the regression coefficient of Early Market Feedback is positive. However MacCormack and Verganti report such moderation effect for investment in architectural design. Therefore, we might assume that they abstained from reporting that the regression coefficient of early market feedback as it was not significant. If so, this would be in agreement with our results.

However, our results lead us to reject our Hypothesis H3a, which means our results contradict the corresponding hypothesis in MacCormack and Verganti. We can speculate about explanations for this contradiction. For example, increasing intensive early feedback requires increasing development efforts with increasing requirements uncertainty. Thus two sub-effects result: the positive general effect of increasing intensive early feedback and the negative effect of increasing efforts required to provide intensive early feedback. The two sub-effects may well countervail each other with increasing requirements uncertainty when measuring project performance.

Overall, our findings confirm some insights from the literature and disconfirm others (see above). In order to generalize from our study, one may have to reflect upon two limitations to our empirical work on contract software development projects: One limitation is the number of collected data sets; it is statistically sufficient, but still raises some issues. The other limitation lies in the correlation between the Investment in Architectural Design (A) and the Intensity of Early Feedback (F). We admit the statistical difficulties arising from the correlation. Perhaps, the separating the two techniques is not feasible in real life or when collecting real life project data.

IMPLICATIONS FOR PRACTICE AND THEORY DEVELOPMENT

We confirm existing work that shows correlation between investment in architecture and early feedback. We also confirm sequential development becomes less effective for project performance as requirements uncertainty increases, and may even damage performance as uncertainty rises. We were unable to confirm any effects of uncertainty on the relation between early feedback and project performance.

Requirements uncertainty negatively moderates the effects of investment in architectural design. For agile development approaches, the value of architectural investment falls with increasing uncertainty. However, for plan-driven approaches, investment in architectural design is positive at any level of requirements uncertainty. There are three main implications for practice from our research: Firstly, software development projects should consider non-sequential development in case of high requirements uncertainty. Secondly, they should invest in architectural design in case of higher requirements uncertainty. Last, but not least, they should intensify early feedback, whenever possible.

Contributing to theory, we provide another hint that 'one-size-fits-all approaches' to software development are barely appropriate (Hardgrave et al., 1999; Shenhar, 2001), and that we need to provide the theoretical grounding for a better understanding of contingencies in software development. Further, this research demonstrates that contract software development needs techniques different from those needed in standard software development. It thus contributes to the understanding of the influence of competitive conditions on contract software development techniques. Finally, our results confirm the literature (e.g., MacCormack et al., 2001) that investment in architectural design is advantageous in case of requirements uncertainty. However, they leave the investigation whether preventive investment in architectural design is to be preferred over corrective investment demands for future research.

REFERENCES

Abrahamsson, P., Conboy, K. and Xiaofeng, W. (2009) 'Lots done, more to do': The current state of agile systems development research, *European Journal of Information Systems*, 18, 4, 281-284.

Alavi, M. (1984) As assessment of the prototyping approach to information systems development, *Communications of The ACM*, 27, 6, 556-563.

Andres, H. and Zmud, R. (2001-2002) A Contingency Approach to Software Project Coordination, *Journal of Management Information Systems,* 18, 3, 41-70.

Baskerville, R., Mathiassen, L. and Pries-Heje, J. (2005) Agility in fours: IT diffusion, IT infrastructures, IT development, and business, in Richard Baskerville, Lars Mathiassen, Jan Pries-Heje and Janice DeGross (Eds.), *Business Agility and Information Technology Diffusion*, New York, NY, USA, Springer, 3-10.

Baskerville, R. and Pries-Heje, J. (2004) Short cycle time systems development, *Information Systems Journal,* 14, 3, 237-264.

Baskerville, R., Ramesh, B., Levine, L., Pries-Heje, J. and Slaugther, S. (2003) Is Internet speed software development different?, *IEEE Software,* 20, 6, 70-77.

Baskerville, R. and Stage, J. (1996) Controlling prototype development through risk analysis, *MIS Quarterly,* 20, 4, 481-501.

Beck, K. 1999) Extreme programming explained: Embrace change, Addison-Wesley, Reading/MA.

Beynon-Davies, P., Tudhope, D. and Mackay, H. (1999) Information systems prototyping in practice, *Journal of Information Technology,* 14, 1, 107-120.

Bhattacharya, S., Krishnan, V. and Mahajan, V. (1998) Managing new product definition in highly dynamic environments, *Management Science*, 44, 11 part 2, 50-64.

Blackburn, J., Scudder, G. and Van Wassenhove, L. (2000) Concurrent software development, *Communications of the ACM*, 43, 11, 200-214.

Boehm, B. (1988) A spiral model of software development and enhancement, *IEEE Computer*, 21, 5, 61-72.

Borsboom, D. and Mellenbergh, G. (2004) The concept of validity, *Psychological Review*, 111, 4, 1061-1071.

Burns, R. and Dennis, A. (1985) Selecting the appropriate application development methodology, *Data Base*, Fall 1985, 19-23.

Carmines, E. and Zeller, R. (1979) Reliability and validity assessment, Sage Publications, Beverly Hills.

Carte, T. and Russell, C. (2003) In pursuit of moderation: Nine common errors and their solutions, *MIS Quarterly,* 27, 3, 479-501.

Chatterjee, S., Hadi, A. and Price, B. (2000) Regression analysis by example, John Wiley and Sons, New York.

Conboy, K. (2009) Agility from first principles: Reconstructing the concept of agility in information systems development, *Information Systems Research*, 20, 3, 329-354.

Cusumano, M. and Selby, R. (1995) Microsoft secrets: How the world's most powerful software company creates technology, shapes markets, and manages people, The Free Press, New York.

Cusumano, M. and Yoffie, D. (1998) Competing on Internet time. Lessons from netscape and its battle with Microsoft, The Free Press, New York.

Davis, G. (1982) Strategies for information requirements determination, *IBM Systems Journal,* 21, 1, 3-30.
DeLone, W. and McLean, E. (2003) The DeLone and McLean model of information systems success: A ten-year update, *Journal of Management Information Systems*, 19, 4, 9-30.
DeLone, W. and McLean, E. (1992) Information system success: The quest for the dependent variable, *Information Systems Research*, 3, 1, 60-95.
Diamantopoulos, A. and Winklhofer, H. (2001) Index construction with formative indicators: An alternative to scale development, *Journal of Marketing Research*, 38, 2, 269-277.
Dyba, T. and Dingsoeyr, T. (2008) Empirical studies of agile software development: A systematic review, *Information and Software Technology*, 50, 9-10, 833-859.
El Louardi, M., Galletta, D. and Sampler, J. (1998) An empirical validation of a contingency model for information requirements determination, *The DATA BASE for Advances in Information Systems*, 29, 39, 31-51.
Faraj, S. and Sproull, L. (2000) Coordination expertise in software development teams, *Management Science*, 46, 12, 1554-1568.
Fazlollahi, B. and Tanniru, M. (1991) Selecting a requirement determination methodology-contingency approach revisited, *Information & Management*, 21, 5, 291-303.
Fornell, C. and Larcker, D. (1981) Evaluating structural equation models with unobservable variables and measurement error, *Journal of Marketing Research*, 18, 1, 39-50.
Gefen, D., Straub, D. and Boudreau, M. (2000) Structural equation modeling and regression: Guidelines for research practice, *Communications of the Association for Information Systems*, 4, 7, 1-77.
Gemino, A., Reich, B. and Sauer, C. (2007) A temporal model of information technology project performance, *Journal of Management Information Systems*, 24, 3, 9-44.
Hardgrave, B., Wilson, R. and Eastman, K. (1999) Towards a contingency model for selecting an information system prototyping strategy, *Journal of Management Information Systems*, 16, 2, 113-136.
Harter, D., Krishnan, M. and Slaughter, S. (2000) Effects of process maturity on quality, cycle time, and effort in software product development, *Management Science*, 46, 4, 451-466.
Henderson, J. and Lee, S. (1992) Managing I/S design teams: A control theories perspective, *Management Science*, 38, 6, 757-777.
Highsmith J (2002) Agile Software Development Ecosystems, Addison-Wesley, Boston.
Hsu, J., Chan, C., Liu, J. and Chen, H. (2008) The impacts of user review on software responsiveness: Moderating requirements uncertainty, *Information & Management*, 45, 4, 203-210.
Iansiti, M. and MacCormack, A. (1997) Developing products on Internet time, *Harvard Business Review*, 75, 5, 108-117.
Ives, B., Olson, M. and Baroudi, J. (1983) The measurement of user information satisfaction, *Communications of the ACM*, 26, 10, 785-793.
Jarvis, C., Mackenzie, S. and Podsakoff, P. (2003) A critical review of construct indicators and measurement model misspecification in marketing and consumer research, *Journal of Consumer Research*, 30, 2, 199-218.
Jones, C. (1996) Patterns of software systems failure and success, Thompson Computer Press, Boston.
Kraut, R. and Streeter, L. (1995) Coordination in software development, *Communications of the ACM*, 38, 3, 69-81.
Krishnan, V., Eppinger, S. and Whitney, D. (1997) A model based framework to overlap product development activities, *Management Science*, 43, 4, 43-451.
Krishnan, M., Kriebel, C., Kekre, S. and Mukhopadhyay, T. (2000) An empirical analysis of productivity and quality in software products, *Management Science*, 46, 6, 745-759.
Larman, C. and Basili, V. (2003) Iterative and incremental development: A brief history, *IEEE Computer*, 36, 6, 47-56.
MacCormack, A., Kemerer, C., Cusumano, M. and Crandall, B. (2003) Trade-offs between productivity and quality in selecting software development practices, *IEEE Software*, 20, 5, 78-85.
MacCormack, A. and Verganti, R. (2003) Managing the sources of uncertainty: Matching process and context in software development, *Journal of Product Innovation Management*, 20, 3, 217-232.
MacCormack, A., Verganti, R. and Iansiti, M. (2001) Developing products on "Internet time": The anatomy of a flexible development process, *Management Science*, 47, 1, 133-150.
Mahmood, M. (1987) Systems development methods - A comparative investigation, *MIS Quarterly*, 11, 3, 293-311.
Marcoulides, G. and Saunders, C. (2006) PLS: A silver bullet?, *MIS Quarterly*, 30, 2, iii-ix.
Mathiassen, L. and Pedersen, K. (2008) Managing uncertainty in organic development projects, *Communications of the AIS*, 23, 27, 483-500.

Mathiassen L. and Stage J. (1990) Complexity and uncertainty in software design, in Dewayne Perry (Ed.) *Proceedings of the 1990 IEEE Conference on Computer Systems and Software Engineering*, October 8-10, Los Alamitos, CA, 482-489.

Mathiassen, L., Tuunanen, T., Saarinen, T. and Rossi, M. (2007) A contingency model for requirements development, *Journal of the Association for Information Systems*, 8, 11, 569-597.

Moynihan, T. (2000a) Requirements-Uncertainty: Is it best formulated as a latent, aggregate or profile construct?, *European Journal of Information Systems*, 9, 2, 82-90.

Moynihan, T. (2000b) Coping with 'requirements-uncertainty': The theories-of-action of experienced IS project managers, *The Journal of Systems and Software*, 53, 2, 99-109.

Nidumolu, S. (1996) A comparison of the structural contingency and risk-based perspectives on coordination in software-development projects, *Journal of Management Information Systems*, 13, 2, 77-113.

Nidumolu, S. (1995) The effect of coordination and uncertainty on software project performance: Residual performance risk as an intervening variable, *Information Systems Research*, 6, 3, 191-219.

Nunnally, J. (1978) Psychometric theory, McGraw-Hill, New York.

Rai, A. and Al-Hindi, H. (2000) The effects of development process modeling and task uncertainty on development quality performance, *Information & Management*, 37, 6, 335-346.

Rossiter, J. (2002) The C-OAR-SE procedure for scale development in marketing, *International Journal of Research in Marketing*, 19, 4, 305-335.

Royce, W. (1970) Managing the development of large software systems: Concepts and techniques, in *Proceedings WESCON Western Electric Show and Convention*, Los Angeles, CA, USA, 1-9, reprinted in Proceedings of ICSE 9, Pittsburgh, PA, USA, 343-252.

Saarinen, T. and Vepsalainen, A. (1993) Managing the risks of information systems implementation, *European Journal of Information Systems*, 2, 4, 283-295.

Sanchez, R. and Mahoney, J. (1996) Modularity, flexibility, and knowledge management in product and organization design, *Strategic Management Journal*, 17, Winter Special Issue, 63-76.

Shenhar, A. (2001) One size does not fit all projects: Exploring classical contingency domains, *Management Science*, 47, 3, 394-414.

Straub, D., Boudreau, M. and Gefen, D. (2004) Validation guidelines for IS positivist research, *Communication of the AIS*, 13, 24, 380-427.

Thomke, S. (1998) Managing experimentation in the design of new products, *Management Science*, 44, 6, 743-762.

APPENDIX 1. CONSTRUCT MEASURES

We measure *Project Performance (P)* as a formative variable[6] with four independent dimensions (see Table A1). The value of Project Performance is computed as a weighted sum of the four dimensions. We chose to let the PLS-algorithm determine the weights. Thus the weights cannot be interpreted independently from the specific sample investigated. This is no loss of generality since the dimensions are measured on interval scales and therefore are unique only up to linear scale transformations.

P	Project Performance (P)
P1	Customer is very satisfied with delivered software.
P2	Budget was exactly adhered to.
P3	Scheduled date of delivery was exactly adhered to.
P4	In relation to the invested effort, the development team achieved very much.

Table A1. Items for measuring Project Performance

We measure *Requirements Uncertainty (U)* as a reflective variable[7] using three items (see Table A2). All three items refer to the ease of converting user needs into a set of requirements specifications.

U	Requirements Uncertainty
U1	Software requirements could be derived from customer needs without any difficulty.
U2	A sequence of steps could be followed for converting customer needs into software requirements.
U3	Customer needs were easy to understand.

Table A2. Items for measuring Requirements Uncertainty

We measure *Sequential Development (S)* as a formative variable with three independent items (see Table A3). The intention to start development only after the requirements specifications are completed is particularly significant in contractual projects and independent of the intention to start coding only after the design is finalized. Therefore three items are used to capture the intention to avoid overlap of development activities in time. Two items capture the intention to prevent starting an activity prematurely. The third item refers to assuring that the preceding activity is properly terminated and need not to be revisited later.

S	Sequential Development
S1	Design stage was decided to start so late, such that only marginal changes of requirements were expected to occur afterwards.
S2	Coding of a component was started so late, such that only marginal changes of the underlying documents (component specification, design documents) were expected to occur afterwards.
S3	At the transition between two stages, every participant could agree that the preceding stage could be terminated.

Table A3. Items for measuring Sequential Development

6 A variable is formative if the direction of causality is from items to the variable. The items of a formative variable must not be interchangeable, co-variation between indicators is not necessary, and the nomological net of indicators can differ.

7 A variable is reflective if the direction of causality is from variable to the items. The items of a reflective variable need to be interchangeable, co-variation between indicators is necessary, and the indicators share a common nomological net.

We measure Investment in Architectural Design (A) by a subjective estimation as a reflective variable of the effort directed towards optimizing Project Performance. We use the three items depicted in Table A4.

A	Investment in Architectural Design
A1	Architectural design aimed at ensuring that during development the product could be changed easily.
A2	Architectural design aimed at meeting new and changed requirements with minimal effort during realization.
A3	Specific measures were taken to increase flexibility and changeability (e.g., architecture evaluation for flexibility and changeability, explicit priorities for flexibility compared to product performance, use of specific technologies like flexible interfaces, adaptors etc.).

Table A4. Items for measuring Investment in Architectural Design

To overcome that deficiency of our data, we measure *Intensity of Early Feedback (F)* as a reflective variable with three items (see Table A5). More or less agreement with these items - expressed on a 5-point Likert scale - is interpreted as intensive or less intensive early feedback.

F	Intensity of Early Feedback
F1	Intensive early feedback of the customer (or a potential customer) on the software's practical use was obtained.
F2	Early practical experience was gathered about the interaction of various software components.
F3	Before requirements were completely determined, the customer had opportunity to evaluate holistic aspects of the software (like user interface or system behavior).

Table A5. Items for measuring Intensity of Early Feedback

APPENDIX 2. HYPOTHESES

H1a: The effect of Sequential Development (S) on Project Performance (P) is moderated by Requirements Uncertainty (U), i.e., $\frac{\partial P}{\partial S}$ is dependent on U or $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial S}\right) \neq 0$.

H1b: With increasing Requirements Uncertainty (U), the effect of Sequential Development (S) on Project Performance (P) decreases, i.e., $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial S}\right) < 0$.

H1c: The effect of Sequential Development (S) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U), i.e., there is a u_0, such that for $u>u_0$ and for $u<u_0$ the sign of $\frac{\partial P}{\partial S}(u)$ is different.

H2a: The effect of Investment in Architectural Design (A) on Project Performance (P) is moderated by Requirements Uncertainty (U), i.e., $\frac{\partial P}{\partial A}$ is dependent on Requirements Uncertainty (U) or $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial A}\right) \neq 0$.

H2b: Increasing Investment in Architectural Design (A) by the same amount yields a greater improvement of Project Performance (P) with increasing Requirements Uncertainty (U), i.e., $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial A}\right) > 0$.

H2c: The effect of Investment in Architectural Design (A) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U), i.e., there is a u_0, such that for $u>u_0$ and for $u<u_0$ the sign of $\frac{\partial P}{\partial A}(u)$ is different.

H3a: The effect of Intensity of Early Feedback (F) on Project Performance (P) is moderated by Requirements Uncertainty (U), i.e., $\frac{\partial P}{\partial F}$ is dependent on Requirements Uncertainty (U) or $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial F}\right) \neq 0$.

H3b: Increasing Intensity of Early Feedback (F) by the same small amount yields a greater improvement of Project Performance (P) with increasing Requirements Uncertainty (U), i.e., $\frac{\partial}{\partial U}\left(\frac{\partial P}{\partial F}\right) > 0$.

H3c: The effect of Intensity of Early Feedback (F) on Project Performance (P) changes direction within the considered range of Requirements Uncertainty (U), i.e., there is a u_0, such that for $u>u_0$ and for $u<u_0$ the sign of $\frac{\partial P}{\partial F}(u)$ is different.

APPENDIX 3. MODELS PREDICTING PROJECT PERFORMANCE (P) - PATH COEEFICIENTS

Model	I		II		III	
U	-0.208	**	-0.374	*	-0.377	***
		[2.077]		[1.982]		[2.997]
S	0.337	*****	0.405	*****	0.405	*****
		[4.353]		[4.848]		[4.418]
A	0.179	**				
		[2.077]				
F			0.368		0.367	**
				[1.554]		[2.037]
U*S	-0.323	****	-0.322	****	-0.322	****
		[3.512]		[4.082]		[3.575]
U*A	-0.253	***				
		[3.090]				
U*F			-0.003	****		
				[0.012]		
R^2_{adj}	0.535		0.549		0.557	
DF	62		62		63	
t-values in squared brackets						
*****p < 0.01%, ****p < 0.1%, ***p < 1%, **p < 5%, *p < 10%						

APPENDIX 4. MODEL IV FOR SUB-SAMPLES

Sub-Sample	lowestU		lowerU		higherU		highestU	
U	-0.452	****	0.008		0.068		-0.560	***
		[6.081]		[0.025]		[0.236]		[3.137]
S	0.579	***	0.724	**	0.855	***	-0.327	**
		[3.690]		[2.244]		[3.438]		[2.267]
A	0.216	**	0.185		0.333		-0.138	
		[2.242]		[0.971]		[1.558]		[0.942]
R^2_{adj}	0.825		0.605		0.691		0.586	
DF	13		13		13		13	
t-values in squared brackets								
****p < 0.1%, ***p < 1%, **p < 5%								