Association for Information Systems
AIS Electronic Library (AISeL)

ECIS 2012 Proceedings

European Conference on Information Systems (ECIS)

5-15-2012

OVER-REQUIREMENT IN SOFTWARE DEVELOPMENT: AN EMPIRICAL INVESTIGATION OF THE 'IKEA' EFFECT

Ofira Shmueli
Ben-Gurion University of the Negev

Lior Fink
Ben-Gurion University of the Negev

Nava Pliskin
Ben-Gurion University of the Negev

OVER-REQUIREMENT IN SOFTWARE DEVELOPMENT: AN EMPIRICAL INVESTIGATION OF THE 'IKEA' EFFECT

Shmueli, Ofira, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel, ofirash@bgu.ac.il

Fink, Lior, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel, finkl@bgu.ac.il

Pliskin, Nava, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel, pliskinn@bgu.ac.il

Abstract

One of the major risks in software development projects is the phenomenon of Over-Requirement, also known as over-specification and gold-plating, where a product or a service is specified beyond the actual requirements of the customer or the market. We argue that Over-Requirement is partially due to the emotional involvement of developers with specified features, an involvement associated with the IKEA or the I-designed-it-myself effect, which implies that people come to overvalue their creations when successfully designed or constructed by them. To investigate this argument, we conducted an experiment in the context of software development in which over 200 undergraduate students participated. The experiment required participants to complete a specification task and measured the change in perceived valuation of a specified nice-to-have feature, by measuring it before and after its specification was completed. The experiment results confirmed the existence of the IKEA effect and its influence on Over-Requirement. The results also imply that the IKEA effect in software development is multifaceted with the level of specification difficulty affecting the magnitude of the IKEA effect, whether the specification difficulty is objective (in terms of constrained specification duration or unconstrained specification freedom) or subjective (as reported by participants).

Keywords: Software Development, Over-Requirement, Over-Specification, Gold-Plating, IKEA Effect, I-designed-it-myself Effect.

1 Introduction

A major risk in software development projects is the phenomenon of Over-Requirement, also known as over-specification and gold-plating, where a product or a service is specified beyond the actual requirements of the customer or the market (Boehm and Papaccio, 1988; Ronen and Pass, 2008). Once a development project is launched, it is very difficult, if not impossible, to cut off its scope, even when some features are not really necessary (Dominus, 2006). Despite the negative impacts of Over-Requirement, it is a common phenomenon that is reported to exceed 30% of developed features (Coman and Ronen, 2009).

The purpose of this work is to show that Over-Requirement is partially due to emotional involvement of developers with already-specified features. More specifically, our objective is to investigate in software-development projects the IKEA effect (Ariely and Jones, 2008), also termed the I-designed-it-myself effect (Franke et al., 2010), which implies that people come to overvalue their creations when they themselves successfully design or construct products. Moreover, if the IKEA effect is indeed manifested, we wish to find out whether and to what extent it can explain the Over-Requirement phenomenon. To meet these objectives, we conducted an experiment targeting the IKEA effect in the context of software development. The following section presents the background for the experiment as well as our hypotheses, leading to details about the experiment method and results respectively in Sections 3 and 4. The paper concludes with a discussion of our results, in Section 5, as well as the contributions and research limitations of this paper, in Section 6.

2 Theoretical Background and Hypotheses

2.1 Over-Requirement

Boehm (1991) included Over-Requirement among the top 10 software development risks, and NASA (1992) listed Over-Requirement among the eight "don't do" warnings in the software-development context. Similarly, Over-Requirement was mentioned as one of the top 10 or 20 risks in most studies devoted to identifying, classifying, and ranking software-development risks (Baccarini et al., 2004; Houston et al., 2001; Khanfar et al., 2008; Schmidt et al., 2001). Table 1 lists some of the major damages that make Over-Requirement risky, varying from delayed launch, through excessive complexity, to demise of an entire company. The first two negative outcomes on the list relate to exceeding planned resources as time or budget. The next three negative outcomes are related to the impact on software complexity, reliability, and maintainability. The rest of the list relates to project resources wasted on functionality of no value instead of core-business functionality as well as to some second-stage outcomes from the users' perspective and possible implications for the entire organization. Nevertheless, Over-Requirement is a common, hardly-reversible phenomenon since excessive extra features introduced during the requirement-engineering phase or later are very rarely cut off from scope (Dominus, 2006; Wetherbe, 1991).

Damage	References
Delayed project launch	(Buschmann, 2009; Coman and Ronen, 2009; Coman and Ronen, 2010)
Project overruns	(Buschmann, 2009)
Excessive complexity	(Battles et al., 1996; Buschmann, 2010; Coman and Ronen, 2009; Coman and Ronen, 2010)
Increased probability of defects and reliability problems	(Coman and Ronen, 2010; Westfall, 2005)
Difficult to manage and costly to maintain systems	(Battles et al., 1996; Buschmann, 2010; Elliott, 2007)
Devoting human and machine resources developing functionality of no real value	(Elliott, 2007; Westfall, 2005)
Defocusing and distraction from real value requirements	(Coman and Ronen, 2009; Coman and Ronen, 2010; Elliott, 2007)
Cutting-off core features due to project time constraints	(Coman and Ronen, 2009; Coman and Ronen, 2010)
Reduced user satisfaction	(Kautz, 2009; Rust et al., 2006)
Hurt supplier's reputation	(Kautz, 2009)
Loss of the entire company	(Coman and Ronen, 2009; Coman and Ronen, 2010)

Table 1. Over-requirement damages as reported in the literature

The causes of Over-Requirement seem to be rooted in human nature and behaviour, with professional interest or pride of developers and demands of users considered to be the main reasons (Ropponen and Lyytinen, 2000). Users and developers alike often ignore business requirements for the sake of advanced technology (Buschmann, 2009; Schmidt et al., 2001). Developers sometimes introduce unauthorized features that satisfy their own interests (Coman and Ronen, 2010; McConnell, 1997) and, in some cases while adopting the "optimizer approach" (Ronen and Pass, 2008), wish to achieve the best possible solution (Rust et al., 2006; Westfall, 2005) or aim to fulfil all future needs, adding just-in-case functionality (Buschmann, 2010; Coman and Ronen, 2010). Users often exhibit an all-or-nothing attitude (Cule et al., 2000), adding costly "bells and whistles" to system requirements (Markus and Keil, 1994), and even trying to coax individual developers into implementing their favorite features (McConnell, 1997). In the case of an outsourced software-development project, a contract type of time-and-materials might contribute to Over-Requirement (Kautz, 2009), due to the inclination to invest more time to boost earnings, a behaviour that is characteristic of companies as well as individuals (Gary, 2009). The cause of Over-Requirement is sometimes related to politics according to DeMarco & Lister (2003), who describe a case where some of the stakeholders, who are adversaries, overload a project with excessive functionality.

2.2 The IKEA effect

This paper focuses on the IKEA effect, which pertains to the influence of one's putting himself or herself into the production process of an object on the perceived object's value. This behavioural-economics effect, also termed the I-designed-it-myself effect (Franke et al., 2010), is named after the successful Swedish do-it-yourself furniture manufacturer (Ariely and Jones, 2008). The IKEA effect implies that people who construct a product themselves come to overvalue their creations (Norton et al., 2009; 2011). People who design and construct an object themselves, whether manually (Norton et al., 2011) or in an intangible manner using a computerized toolkit, exhibit a value increment ascribed to that object purely due to the fact that they feel like its originators (Franke et al., 2010). It is noteworthy that when people fail to complete building a product, the IKEA effect dissipates since only fruitful labor seems to lead to higher valuation (Norton et al., 2009; 2011). The IKEA effect was demonstrated in a series of experiments that focused on the perceived value gained after designing a product, along with some other relevant factors such as the outcome of the process and the participant's contribution to the process (Franke et al., 2010). It is worth mentioning that testing the impact of one's contribution feeling, which was manipulated by different levels of design freedom, has revealed a positive influence on the perceived value.

2.3 Hypotheses

We wish to follow previous experiments (Franke et al., 2010; Norton et al., 2011), where building an origami or designing a T-shirt have resulted in added value in the constructor's or designer's eye, to find out whether the IKEA effect occurs also while specifying and designing a software feature in software-development projects. Although in previous research the added value was measured by terms of money and willingness-to-pay, value can be measured in other terms, as done in this work. We argue that due to the IKEA effect software developers get emotionally attached to their creation, whether the features created are essential or merely nice-to-have, and once their ideas and cognitive effort are invested in feature specification, their feelings toward that feature and its perceived value change. This change reduces the willingness of developers to exclude the feature from project scope, potentially leading to Over-Requirement. Since the IKEA effect is about the specification effort, the first two of our four hypotheses address two effort factors: specification duration (the length of time devoted to the specification task) and specification freedom (the freedom to contribute to the specification task).

One of the explanations for the IKEA effect is that individuals who create an object become attached to it by feeling psychological ownership (Pierce et al., 2003), which is associated with the endowment effect. The "endowment effect" (Thaler, 1980), which implies that people place a higher value on objects they own than on objects they do not own, has been shown to be positively affected by the duration of ownership (Strahilevitz and Loewenstein, 1998). To test this duration effect in the context of software development, the first hypothesis in this study concerns the relationship between specification duration and feature valuation, arguing that the emotional attachment of software developers to features they specified, including Over-Required features, is positively related to the time invested in the specification task. Thus, perceived valuation of a specified feature increases with specification duration.

H1: Specification duration has a positive impact on feature valuation

Previous research has shown that when the design of a product is enabled by a computerized toolkit, availability of more design freedom generates higher subjective value than when design freedom is limited (Franke et al., 2010). To test this effect in the context of software development, the second hypothesis concerns the relationship between specification freedom available to developers and feature valuation, arguing that the emotional attachment of software developers to features they specified, including Over-Required features, is positively related to the freedom enjoyed during the specification task due to the increased investment of cognitive effort and the increased feeling of contribution. Thus, perceived valuation of a specified feature increases with specification freedom.

H2: Specification freedom has a positive impact on feature valuation

Norton et al. (2009) argue that for the IKEA effect to be manifested, the task at hand should be difficult enough to bring about emotional attachment and lead to higher valuation, but not too difficult to complete. In the software-development context, increased freedom might make a task too difficult to complete, either because of a time constraint or due to some other subjective individual difficulty. The third hypothesis pertains to the time constraint while the fourth hypothesis pertains to the subjective individual difficulty.

H3 concerns the two-way interaction between the impacts of specification duration and specification freedom on feature valuation. Keeping task difficulty in mind, we argue that specifying a task under high freedom within short specification duration might make the task too difficult to complete. On the other hand, specifying a task under low freedom within long specification duration might make the task not difficult enough. In both situations, feature valuation will be negatively affected.

H3: There is a two-way interaction between specification duration and specification freedom in their impact on feature valuation, so that valuation increases with duration for high specification freedom and decreases with duration for low specification freedom

As described above regarding the argument of Norton et al. (2011) that once the task becomes too difficult the value is reduced, the next assumption is that beyond the objective factors of difficulty (specification freedom and duration), which are addressed by H3 above, a task might be considered difficult as a consequence of subjective factors, in particular the extent to which the task presents a challenge to the software developer. Thus, H4 concerns the degree to which individual software developers are challenged by the specification task, arguing that these feelings of challenge moderate the effects of specification freedom and duration on feature valuation. Specifically, the developer cannot benefit from the combination of high freedom and long duration (the two-way interaction described in H3) when s/he feels a high degree of challenge. In such situations, the developer may prefer tasks that are characterized by low freedom and long duration. Put differently, we expect the combined effect of specification freedom and duration to be manifested only when the task at hand is not perceived to be overly challenging.

H4: There is a three-way interaction among specification duration, specification freedom, and feeling of challenge in their impact on feature valuation, so that the former two-way interaction between specification duration and freedom (H3) is maintained only for individuals who do not experience a high degree of challenge

3 Method

To empirically test the research model depicted in Figure 1 and the four hypotheses developed above, we conducted an experiment that was based on a factorial design of 2×2×2 with three dichotomous independent variables: (1) specification duration, (2) specification freedom, and (3) challenge feeling. The first two independent variables (objective factors) were manipulated, while the third (subjective factor) was measured.

The dependent variable, i.e., the change (Δ) in perceived valuation of a certain software feature being specified by participants, was measured as well by asking experiment participants to assess the importance of the specified feature twice, before and after specification, and then calculating the difference between valuation figures (Δvaluation).

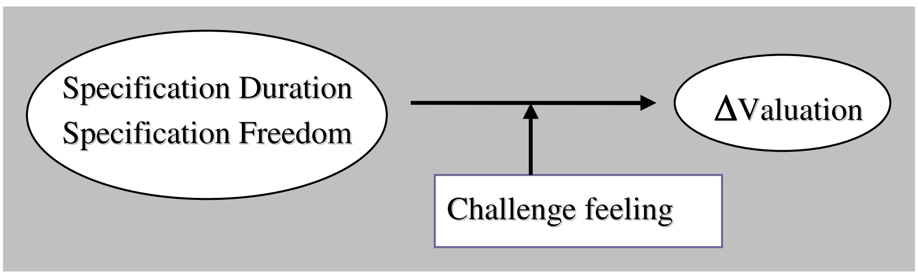

Figure 2. Research model

The experiment involved three steps and took about one hour. Step 1 involved filling up Questionnaire I by the participants. The beginning of Questionnaire I presented a fictitious case about developing a software system for remote-banking clients. After presenting the case and in order to set the perceived participant valuation baseline, participants were asked to evaluate the importance of 16 different functional features listed, given the system's goal, on a 1-100 importance scale. At the end of Step 1 the filled-up Questionnaire I was collected.

Step 2 of the experiment required participants to specify one of the 16 features evaluated in Step 1. The specified feature, deliberately chosen to be unnecessary but nice-to-have, was the same for all participants but they were unaware of the sameness (practically, we were interested only in the valuations of this feature and used the other 15 features as distractions). For manipulating the first independent variable, two different time durations were defined (short duration of 10 minutes and long duration of 30 minutes) for separate experimental groups. For manipulating the second independent variable, the specification task was provided at two different freedom levels (low freedom by providing detailed instructions and high freedom by providing brief instructions) for separate experimental groups.

Step 3 of the experiment involved re-assessing feature valuation as perceived by participants. Participants were asked to fill up Questionnaire II by re-evaluating the importance (on a 1-100 scale, as before) of the specific feature they worked on. In addition, Questionnaire II included questions about participant feelings regarding various aspects. For measuring the third independent variable, one of these questions asked participants to report the challenge level of the specification task, on 1-7 Likert scale (1-not challenging at all, 7 – very challenging). The end of Questionnaire II included demographic and background questions such as gender, age, and previous software-development experience. At the end of Step 3 the filled-up Questionnaire II was collected.

Advanced Engineering students in the Information Systems track at an Israeli university were asked to participate in the experiment. Participants were randomly assigned to groups that performed the experiment in separate sessions.

4 Results

Data collection yielded 212 gender-balanced participants: 94 fourth-year students (79% response rate) and 118 third-year students (90% response rate). The findings of the experiment described in the previous section were obtained via analysis of variance (ANOVA) in SPSS. Table 2 shows the cell means for Δvaluation, and Table 3 presents the ANOVA results for two models: the hypothesized model (with four hypothesized effects) and the full factorial model (controlling for the main effect of challenge feeling and for its two-way interactions with specification duration and freedom). As shown in Table 3, the two different models yield consistent results.

Challenge feeling	Low freedom		High freedom	
	10 min	30 min	10 min	30 min
Low	14.214 (6.738)	-5.875 (6.303)	2.125 (8.914)	19.667 (5.942)
High	7.000 (4.324)	10.027 (4.145)	6.146 (3.937)	4.417 (4.212)

Table 2. Cell means for Δvaluation; estimated marginal means are shown with std. errors in parentheses

	Model 1 Hypothesized model	Model 2 Full factorial model
Duration	0.006	0.006
Freedom	0.182	0.182
Duration × Freedom	4.019 *	4.019 *
Duration × Freedom × Challenge	2.454 *	6.681 **
Challenge		0.024
Challenge × Duration		0.055
Challenge × Freedom		1.475

Table 3. *ANOVA results for the hypothesized and full factorial models; F values are shown; * p < 0.05, ** p < 0.01*

Testing H1, concerning the impact of specification duration as a main effect, did not reveal any significant change in Δvaluation between the 10-minutes and 30-minutes specification duration groups (F=0.006, p=0.939). The same is true for H2, concerning the impact of specification freedom as a main effect (F=0.182, p=0.670). However, the hypothesized two-way and three-way interactions were found to be statistically significant, supporting H3 and H4. Figure 2, depicting findings related to H3, demonstrates that for high specification freedom, specification duration positively affects Δvaluation, while the opposite occurs for low specification freedom (F=4.019, p<0.05).

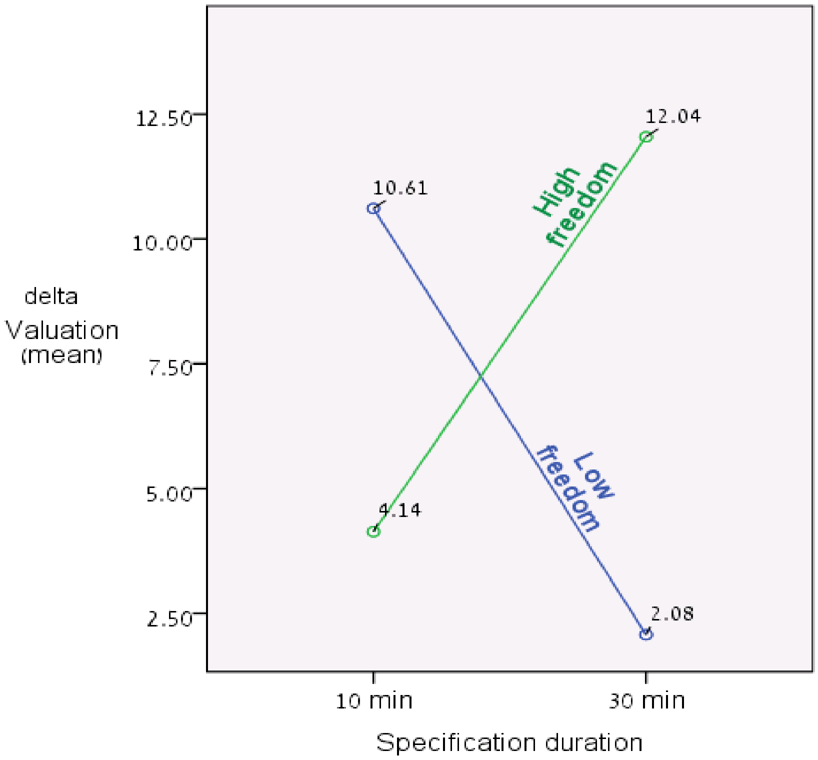

Figure 2. *The interaction between specification freedom and duration (H3)*

Taking into consideration the challenge feelings reported by participants, the hypothesized three-way interaction was found to be statistically significant (F=6.681, p<0.01). This interaction is shown in Figure 3a for participants reporting a challenging (4-7) experience and in Figure 3b for participants reporting an unchallenging (1-3) experience. Figure 3b shows a positive effect of specification freedom on Δvaluation over time, similar to Figure 2, while Figure 3a shows otherwise.

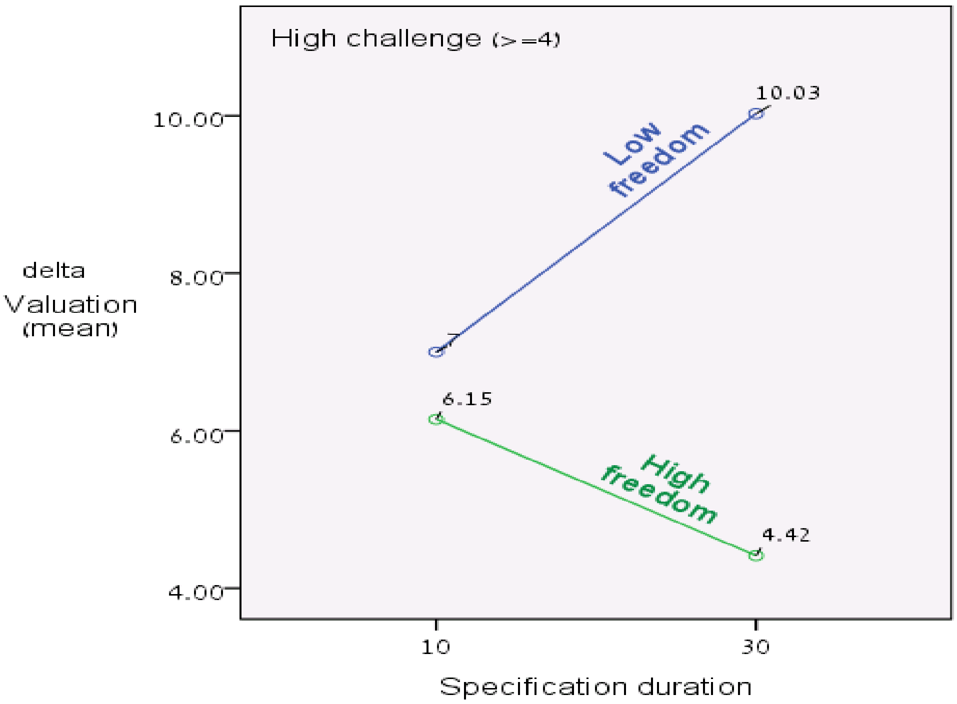

Figure 3a. *The interaction between specification freedom and duration for challenged participants (H4)*

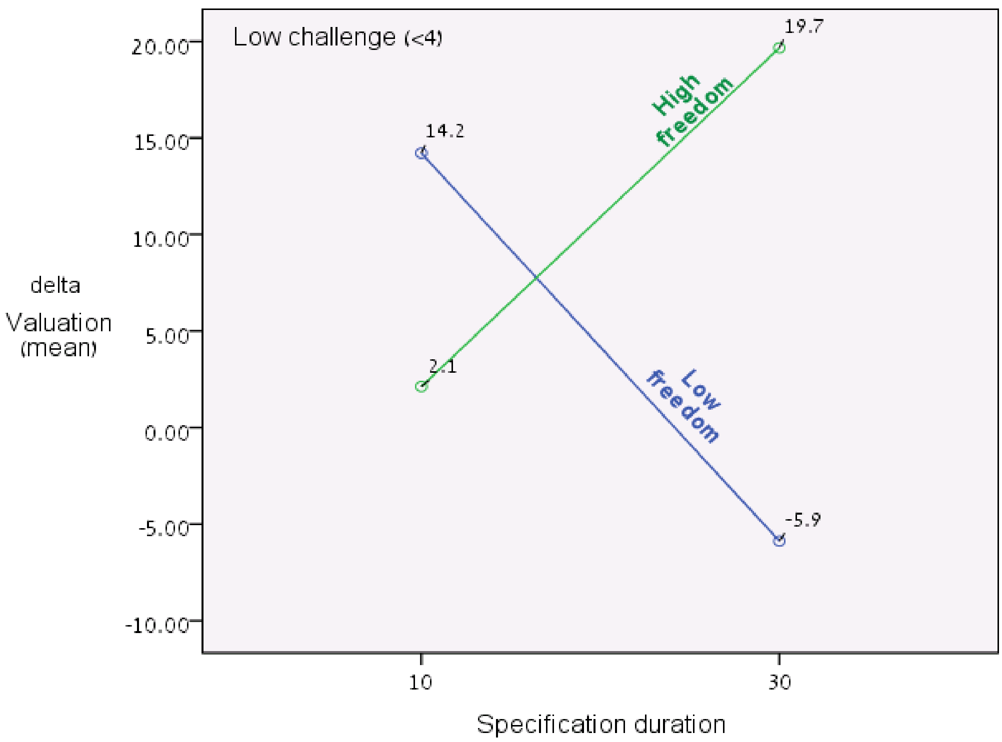

Figure 3b. *The interaction between specification freedom and duration for unchallenged participants (H4)*

5 Discussion

Our findings have revealed that valuations provided after completion of the specification task (*M*=78.11) were significantly above (p<0.001) valuations provided before commencing the specification task (*M*=70.77). This difference (Δvaluation) in and by itself may indicate the existence of the IKEA effect, because all participants specified the feature to some extent.

The results of the experiment demonstrate the IKEA effect within the context of software development and explain how the IKEA effect can lead to the development of Over-Required features. When a software developer specifies a feature, investing cognitive effort in the process, s/he gets attached with that specified feature and perceives it as more valuable than it actually is. These attachment feelings make it harder to exclude the feature from the project scope even when it is not essential.

We expected to observe more attachment feelings with increased time or creativity allowed for specifying the feature. The absence of evidence supporting H1 and H2 (main effects for specification duration and specification freedom) implies that the IKEA effect influences developer attachment toward specified feature in a rather complex way. This implication is strengthened by evidence found in support of H3 and H4 (interaction of duration × freedom and interaction of duration × freedom × challenge). These interactions suggest that the specification freedom and subjective difficulty feeling (measured by reported challenge-level), as well as specification duration, influence the change in perceived valuation of a specified feature. Both interactions relate to Norton et al's (2011) argument that in order to gain higher valuation the task should be difficult enough but not too difficult. It is important to note that both interactions remain statistically significant even when controlling for the potential confounding effects of participants' feeling of contribution, their perception of the task as being time consuming, and the quality of their work (estimated by one of the authors).

The interaction of specification duration and specification freedom suggests that the attachment feeling is influenced by a combination of both factors. For freedom level to impact attachment, it is essential to have the right duration for the specification mission. This means that when specification freedom is high one should allot more specification time and when specification freedom is low less time can be allotted. Otherwise, the specification task might be either too difficult or too boring, leading to less apparent valuation increment vis-a-vis the specified feature and hence less attachment. This interaction implies that there might be a difference between simple structured tasks where one has to follow detailed instructions, and more complicated unstructured tasks where the instructions to follow are vague. Since most tasks in previous research on the IKEA effect involved following instructions, with no time constraint (Franke et al., 2010; Norton et al., 2011), this difference in the patterns of the IKEA effect should be further explored.

The three-way interaction brings into consideration the subjective difficulty feeling regarding the specification task. While the two-way interaction involved objective difficulty factors of specification freedom and duration, we assume that software developers experience different levels of difficulty, affecting their individual ability to handle the specification task and, therefore, their attachment level and magnitude of the freedom × duration effect. For the three-way interaction, the results show a difference between low-challenged and high-challenged participants, with low-challenged participants confirming the freedom × duration effect (Figure 3b). This finding can be explained by arguing that high-challenged participants found the high freedom condition too difficult, while low-challenged participants found the low freedom condition too boring or easy. Our findings furthermore suggest that these effects are exacerbated over time.

6 Contributions and Limitations

The above findings might serve practitioners as well as researchers. Managers of software development projects should be aware that, due to the IKEA effect, once the developers they manage specify a feature, they come to Over-Value that feature. The main contribution of the present study to research lies in drawing upon concepts from behavioural economics to understand the phenomenon of Over-Requirement in software development projects. Beyond this contribution, the study also advances the literature on the IKEA effect by exploring the interactions between the objective factors of freedom and duration and the subjective factor of challenge.

As the current study focused on the IKEA effect, which is one of several effects reported in the behavioural-economics literature, other behavioural effects might be relevant to the Over-Requirement phenomenon. Future research should also explore the endowment effect (Thaler, 1980), derived from Prospect Theory (Kahneman & Tversky 1979), which implies that people place a higher value on objects they own than on objects they do not own.

Our study has two primary limitations that can be addressed in future research. Although participants were advanced undergraduate students with enough skills and academic background for the specification task, most of them did not have software-development experience on the job in the industry. The experiment took about an hour and tried to emulate reality as much as possible. Yet, it is rarely possible in an experiment to account for such aspects as real task durations, human interactions or organizational culture and maturity. To address both limitations, further research can target participants that work full time as software developers and ask them relevant questions about features they actually specified.

Notwithstanding its limitations, this study uses behavioural economics to demonstrate the implications of the IKEA effect for software development projects in general and for Over-Requirement in particular. Our findings confirm the notion that the lens of behavioural economics has the potential to advance our understanding of software development processes.

References

Ariely, D. and Jones, S. (2008). Predictably irrational: The hidden forces that shape our decisions. HarperCollins, New York.

Baccarini, D., Salm, G. and Love, P. (2004). Management of risks in information technology projects. Industrial Management and Data Systems, 104, 286-295.

Battles, B.E., Mark, D. and Ryan, C. (1996). An open letter to CEOs: how otherwise good managers spend too much on information technology. The McKinsey Quarterly, 3, 116-116.

Boehm, B. and Papaccio, P. (1988). Understanding and Controlling Software Costs. IEEE Transactions on Software Engineering, 14 (10), 1462-1477.

Boehm, B. (1991). Software Risk Management: Principles and Practices. IEEE Software, 8 (1), 32-41.

Buschmann, F. (2009). Learning from Failure, Part 1: Scoping and Requirements Woes. IEEE Software, 26 (6), 68-69.

Buschmann, F. (2010). Learning from Failure, Part 2: Featuritis, Performitis, and Other Diseases. IEEE Software, 27 (1), 10-11.

Coman, A. and Ronen, B. (2009). Overdosed management: How excess of excellence begets failure. Human Systems Management, 28 (3), 93-99.

Coman, A. and Ronen, B. (2010). Icarus' predicament: Managing the pathologies of overspecification and overdesign. International Journal of Project Management, 28, 237-244.

Cule, P., Schmidt, R., Lyytinen, K. and Keil, M. (2000). Strategies for heading off IS project failure. Information Systems Management, 17 (2), 65-73.

DeMarco, T. and Lister, T. (2003). Risk management during requirements. IEEE Software, 20 (5), 99-101.

Dominus, M. 2006, 15 May 2006-last update. Creeping featurism and the ratchet effect. Available: http://blog.plover.com/prog/featurism.html [2010, 5/25].
Elliott, B. (2007). Anything is possible: managing feature creep in an innovation rich environment. In Proceedings of the IEEE International Engineering Management Conference (IEMC 2007), IEEE, p. 304, Piscataway, NJ, USA.
Franke, N., Schreier, M. and Kaiser, U. (2010). The "I Designed It Myself" Effect in Mass Customization. Management Science, 56 (1), 125-140.
Gary, M. 2009, 16 december 2009-last update. Poor industry consultancy threatens data centre constructions [Homepage of Sentrum], [Online]. Available: http://www.sentrum.com/news-information/press-releases/item/poor-industry-consultancy-threatens-data-centre-constructions/ [2011, 6/21].
Houston, D.X., Mackulak, G.T. and Collofello, J.S. (2001). Stochastic simulation of risk factor potential effects for software development risk management. The Journal of Systems & Software, 59 (3), 247-257.
Kautz, K. (2009). The Impact of Pricing and Opportunistic Behavior on Information Systems Development. Journal of Information Technology Theory and Application, 10 (3), 24-41.
Khanfar, K., Elzamly, A., Al-Ahmad, W., El-Qawasmeh, E., Alsamara, K. and Abuleil, S. (2008). Managing Software Project Risks with the Chi-Square Technique. International Management Review, 4 (2), 18-29.
Markus, M.L. and Keil, M. (1994). If we build it, they will come: Designing information systems that people want to use. Sloan management review, 35, 11-25.
McConnell, S. (1997). Achieving leaner software. IEEE Software, 14 (6), 127-128.
NASA (1992). Recommended approach to software development . 3rd edn. Goddard Space Flight Center, Greenbelt, Maryland, USA.
Norton, M.I., Mochon, D. and Ariely, D. (2009). The IKEA effect: When labor leads to love. Harvard Bus.Rev, 87 (2), 30.
Norton, M.I., Mochon, D. and Ariely, D. (2011). The" IKEA Effect": When Labor Leads to Love. working paper. Harvard Business School, 1-34.
Pierce, J.L., Kostova, T. and Dirks, K.T. (2003). The state of psychological ownership: Integrating and extending a century of research. Review of General Psychology, 7 (1), 84-107.
Ronen, B. and Pass, S. (2008). Focused operations management: Achieving more with existing resources. John Wiley & Sons Inc, Hoboken, New Jersy, USA.
Ropponen, J. and Lyytinen, K. (2000). Components of software development risk: How to address them? A project manager survey. IEEE Transactions on Software Engineering, 26 (2), 98-112.
Rust, R.T., Thompson, D.V. and Hamilton, R.W. (2006). Defeating feature fatigue. Harvard business review, 84 (2), 98-107.
Schmidt, R., Lyytinen, K., Keil, M. and Cule, P. (2001). Identifying Software Project Risks: An International Delphi Study. Journal of Management Information Systems, 17 (4), 5-36.
Strahilevitz, M.A. and Loewenstein, G. (1998). The effect of ownership history on the valuation of objects. Journal of Consumer Research, 25 (3), 276-289.
Thaler, R. (1980). Toward a positive theory of consumer choice. Journal of Economic Behavior & Organization, 1 (1), 39-60.
Westfall, L. (2005). The What, Why, Who, When and How of software requirements. ASQ World Conference on Quality and Improvement Proceedings, 59, 97-105.
Wetherbe, J.C. (1991). Executive information requirements: Getting it right. MIS Quarterly, 15 (1), 51-65.