Asia Pacific Journal of Information Systems
Vol. 26 No. 2 (JUNE 2016), 322-337

ISSN 2288-5404 (Print) / ISSN 2288-6818 (Online)
http://dx.doi.org/10.14329/apjis.2016.26.2.322

The Dark Side of Emotional Involvement in Software Development: A Behavioral Economics Perspective

Ofira Shmueli[a], Nava Pliskin[b,*], Lior Fink[c]

[a] *Ph.D., Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Israel.*
[b] *Professor, Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Israel*
[c] *Associate Professor, Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Israel*

A B S T R A C T

Research on information systems and software engineering has often neglected behavioral effects, which may play a role in decision making on software development. The current study addresses this issue by empirically investigating the behavioral roots of over-requirement in the context of a software development project via an experiment. The negative phenomenon of over-requirement refers to specifying a software system beyond the actual needs of the customer or the market, which overload the system with unneeded features. The research question addressed here is whether over-requirement is due in part to the emotional involvement of developers with the software features they developed because of behavioral effects. Previous studies have demonstrated that under the endowment, I-designed-it-myself, and IKEA effects, people become emotionally involved and overvalue physical items that they respectively possess, self-design, or self-create. The findings of our experiment show that participants over-valued features they were assigned to be responsible for, to specify, or to construct, thereby confirming that the three behavioral effects play a role in software development decisions and affect over-requirement. Thus, the study contributes to software development research and practice from the behavioral economics perspective, highlighting the roots of over-requirement.

Keywords: Software Development, Over-Requirement, Endowment Effect, I-Designed-it-Myself Effect, IKEA Effect

Ⅰ. Introduction

Past research on information systems in general and on software engineering in particular largely ignores the behavioral economics perspective. Technical and methodological approaches to system development, which evolved over decades (Boehm, 2006), improved and leveraged technical aspects of information systems but, at the same time, failed to significantly lower the high failure rates in software

*Corresponding Author. E-mail: pliskinn@bgu.ac.il Tel: 97286472203

development projects, as was reported in the 2009 Standish Group Report (http://www.standishgroup.com/newsroom/chaos_2009.php, cited by Boehm and Lane (2010)). In line with the call by Goes (2013) to bring in the behavioral economics perspective, to which a number of researchers already responded recently (Fleischmann et al., 2014; Shmueli et al., 2015a; Shmueli et al., 2015b), this study shows that over-requirement in software development is partially due to behavioral effects.

The problem of over-requirement, known also as over-specification and gold-plating, refers to specifying a software system beyond the actual needs of the customer or the market and, thus, overloading the system with unnecessary features (Boehm and Papaccio, 1988; Ronen and Pass, 2008). The negative phenomenon of over-requirement, pertaining to at least 30% of the functionality in software projects (Coman and Ronen, 2009), harms projects and companies (Buschmann, 2009; Coman and Ronen, 2010). Hence, Boehm (1991) considered over-requirement as one of the top 10 software development risks and NASA (1992) reported it as one of eight “don't do” software-development warnings. Ever since, over-requirement has continued to be mentioned as a top risk (Baccarini et al., 2004; Bernstein, 2012; Houston et al., 2001; Kaur et al., 2013; Khanfar et al., 2008; Malhotra et al., 2012; Pass and Ronen, 2014; Schmidt et al., 2001; Wheatcraft, 2011).

The list of negative consequences of over-requirement is quite long and includes delayed launch, excessive complexity, cutting off core features to make room for over-required ones, going out of business for an entire company (Coman and Ronen, 2009; Coman and Ronen, 2010), spending resources on functionality of no value (Westfall, 2005), defocusing away from valued requirements (Elliott, 2007), and resource overruns (Buschmann, 2009). Over-requirement is associated also with reliability problems (Coman and Ronen, 2010; Westfall, 2005) and with difficult-to-manage and costly-to-maintain systems (Battles et al., 1996; Buschmann, 2010). In fact, more functions may paradoxically lead to lower value from a user perspective (Rust et al., 2006), so much so that when a system does not meet the customer's wishes, s/he may opt for another supplier next time (Kautz, 2009). Over-requirement is also hardly reversible since any introduced and developed feature, whether over-required or not, is rarely removed from project scope (Dominus, 2006).

Most over-requirement reasons are related to the behavior of software development stakeholders, including the interest of developers, the ambitious demands of users (Ropponen and Lyytinen, 2000), the tendency of users and developers to often ignore business requirements for the sake of advanced technology (Buschmann, 2009; Schmidt et al., 2001), or the politics involved (DeMarco and Lister, 2003). Software developers may introduce over-requirement to satisfy their professional interest (Coman and Ronen, 2010; McConnell, 1997), to achieve the best possible solution (Rust et al., 2006; Westfall, 2005), or to fulfil all future needs and add just-in-case functionality (Buschmann, 2010; Coman and Ronen, 2010), especially when not knowing which features will be more important (Anton and Potts, 2003; Boehm, 1996). The desire to build one off-the-shelf software system that fits all (Coman and Ronen, 2010; Rust et al., 2006) or to release improved versions on a continuous basis (Coman and Ronen, 2010) are two other reasons for over-requirement. With an all-or-nothing attitude (Cule et al., 2000), users may favour inclusion of costly bells and whistles (Markus and Keil, 1994) and coax developers into implementing their favourite yet unneeded features (McConnell, 1997). Given the negative impacts of

over-requirement, these descriptive behaviors actually contradict normative economic principles.

It thus makes sense to consider in the context of software development the behavioral effects that behavioral economists have experimentally investigated. While most behavioral economics research on these effects referred to tangible objects, as using mugs and candy bars (Ariely and Jones, 2008; Kahneman et al., 1991), some research referred to intangible goods, especially with regard to the endowment effect (e.g., Kahneman et al. (1990) and Hoorens et al. (1999)). The endowment effect was demonstrated by Hoorens et al. (1999) with respect to working time and by Kahneman et al. (1990) with respect to property rights. Also, it was demonstrated by Heyman et al. (2004) in an experiment that emulated Internet auctions, that even just imagining possession resulted in over-bidding.

The purpose of this work is therefore to show that over-requirement is associated with three effects demonstrated by behavioral economists: (1) the endowment effect - the tendency of people to overvalue their possessions (Thaler 1980); (2) the I-designed-it-myself effect - the tendency of people to overvalue their self-designed products (Franke et al., 2010); and (3) the IKEA effect - the tendency of people to overvalue their self-constructed products (Ariely and Jones, 2008). The research question addressed in this study is whether the emotional involvement that evolves upon assuming responsibility for a software feature, specifying it, or constructing it, biases the importance evaluation of that feature by the software developer, due respectively to the endowment, I-designed-it-myself, and IKEA effects. An experiment emulating a fictitious software development project was conducted to investigate this research question, using a 2×2×2 factorial design created by manipulating the allocation of responsibility for an over-required feature and the specification and construction of its pseudo-code.

To set the background for presenting the details of the experimental method in Section 3, Section 2 concisely reviews the literature on the three effects and presents the hypotheses tested. Then, Section 4 presents the results while Section 5 discusses the limitations of this research and points to future research directions. Section 5 also refers to the contribution of this empirical study to the understanding of software developers' perception biases regarding feature importance that may lead to sustained over-requirement. This paper makes practitioners aware that developers are prone to cognitive biases due to the endowment, I-designed-it-myself, and IKEA effects and alerts researchers to the linkage between behavioral economics and software development, opening the road for theory development at the intersection of both.

Ⅱ. Theoretical Background and Hypotheses

2.1. The Endowment Effect

Derived from Prospect Theory (Kahneman and Tversky, 1979), the endowment effect is about overvaluation of owned objects (Thaler, 1980), implying that ownership positively influences the perceived value of the owned object (Kahneman et al., 1990, 1991). Ariely and Jones (2008) and Kahneman et al. (1991) attribute to the endowment effect a positive valuation difference between an owner's willingness-to-accept (WTA) for an object and a non-owner's willingness-to-pay (WTP) for the same object. While showing that the duration of ownership positively impacts product valuation and perceived at-

tractiveness (Strahilevitz and Loewenstein, 1998), merely touching an object results in perceived ownership and increased valuation (Peck and Shu, 2009).

Hence, we hypothesize that software developers may exhibit ownership feelings while assuming responsibility for a necessary or a nice-to-have feature and attribute a higher value to that feature than otherwise due to the endowment effect. The first hypothesis suggests that upon assuming responsibility for a software feature, including an over-required one, there is a positive impact on its perceived value. Thus,

H1: Assuming responsibility for a feature has a positive impact on its perceived value.

2.2. The I-designed-it-myself Effect

The I-designed-it-myself effect is about the psychological benefit gained by self-designing an object (Franke et al., 2010; Ulrich, 2009), implying that designing a product leads the designer to overvalue it. Franke et al. (2010) demonstrated positive differences between a designer's WTP for a self-designed object and for an identical object designed by others and attributed these differences to the I-designed-it-myself effect. After controlling for other possible explanations such as preference fit, sunk cost effect, and mood effect, Franke et al. asserted that the feeling of being the originator of the object is the primary explanation for the I-designed-it-myself effect. They also found this effect to be mediated by a person's feeling of accomplishment and to be moderated by the quality of the outcome and by the perceived contribution to the design process. Hence, we hypothesize that software developers may become emotionally attached to their creation while specifying a necessary or nice-to-have feature and attribute a higher value to that feature than otherwise due to the I-designed-it-myself effect. The second hypothesis suggests that upon specifying a feature, including an over-required one, there is a positive impact on its perceived value. Thus,

H2: Specifying a feature has a positive impact on one's perceived value of the feature.

2.3. The IKEA Effect

The IKEA effect is about the attachment feelings of people towards objects they self-create and implies that self-constructing a product positively influences its perceived value and leads the creator to overvalue that product (Ariely and Jones, 2008; Norton et al., 2012). Norton et al. (2012) attribute to the IKEA effect the positive differences they experimentally demonstrated in the WTP for a self-assembled object between the constructor and non-constructors, and in the WTP of the constructor between a self-assembled object and an identical object assembled by others. Their experiments covered a variety of object types and showed that this effect holds not only for hedonic objects (as origami or Lego models) but also for more practical ones (as IKEA boxes). Hence, we hypothesize that software developers may become emotionally attached to their creation while constructing a necessary or nice-to-have feature and attribute a higher value to that feature than otherwise due to the IKEA effect. The third hypothesis suggests that upon constructing a feature, including an over-required one, there is a positive impact on its perceived value. Thus,

H3: Constructing a feature has a positive impact on one's perceived value of the feature.

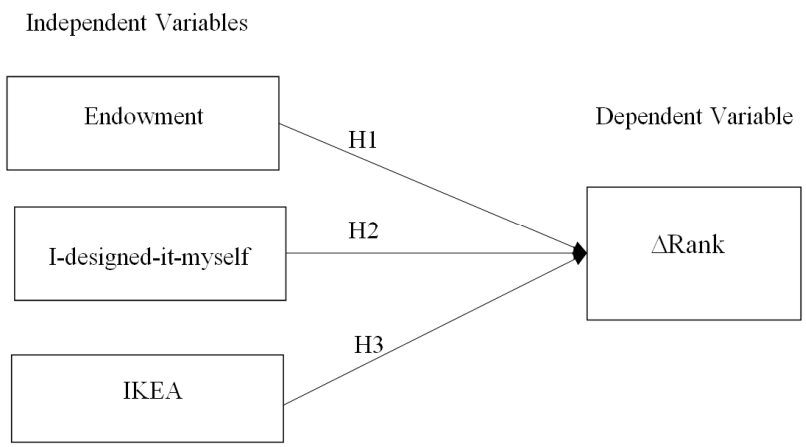
Ⅲ. Method

The experiment conducted to empirically test the research model (<Figure 1>), including the above three hypotheses, used a 2×2×2 factorial design to manipulate three dichotomous independent variables (endowment, I-designed-it-myself, and IKEA) and to measure the dependent variable, Δrank, which represents the change in perceived importance ranking order for a specific feature before and after manipulation.

The participants in the experiment were advanced senior students of Industrial Engineering, majoring in Information Systems, who took one of two courses, Introduction to Decision Making or Information Systems Management, because students in these courses had the knowledge and capabilities to perform the experimental tasks at the time of participation. Incentives for participation included an extra bonus point (1 out of 100) to the course grade and, to motivate applying the best cognitive effort in the experiment, a chance to win one of four $70 worth 500GB 2.5" Passport portable external hard drives in a raffle (such that the better the quality of work, the higher the chances of winning). Following a pilot study with a sample of 16 students from the same population, performed to test the experimental design and manipulations, minor modifications were made to them according to feedback received from the pilot study participants.

Relying on student participants is common in experiments devoted to behavioral economics research (Franke et al., 2010; Kahneman et al., 1991; Norton et al., 2012). Recruiting students as participants is acceptable as well in experiments exploring behavior, behavioral biases, and decision making in the context of software development (Andres and Zmud, 2001; Keil et al., 2007; Keil et al., 2000; Khatri et al., 2006; Umapathy et al., 2008; Williams et al., 2000) or in experiments exploring information system investment (Legoux et al., 2014).

The experiment took about one hour and involved five steps, during which participants were randomly assigned to one of eight groups, representing the between-subject combinations of the three effects being manipulated dichotomously. At the beginning of each step, participants were informed of the time dedicated to that step and were instructed to carefully



<Figure 1> Research Model

read a written mission statement that instructed them, as described next, to either complete a paper-based questionnaire or perform on paper some other task. At the end of each step and prior to the next step, the completed papers were collected. The five steps detailed next were devoted to the background story and the baseline ranking (Step 1), the three manipulations (Step 2 through Step 4, one manipulation per independent variable), and the epilogue (Step 5).

Step 1 - Seven minutes were dedicated to the first step. To begin with, a written fictitious case about developing a software system for remote-banking clients was presented to participants, including a list of nine functional features (presented in the Appendix), deliberately composed to be diverse in terms of importance toward the system's goal. To measure their perceived value ranking baselines, participants were then asked to complete a questionnaire and rank the nine features in descending importance toward meeting the system's goal (from the most important in the first top place to the least important in the last bottom place).

Step 2 - Five minutes were dedicated to the second step, which manipulated the *endowment* effect by assigning participants with *responsibility* for one of the nine features of the remote-banking system. Participants were told in a written statement that they assume responsibility to further develop one feature of the system in future development phases of the project. Half of the participants, who were grouped under the endowment condition (endowment manipulation = Yes), were assigned the same feature, "Presenting my itemized expense report for a specific period" (hereinafter the "Experimental Feature"), deliberately chosen to be over-required given the system's goal. The other half of participants (endowment manipulation = No) were not assigned the Experimental Feature but were assigned an alternative feature, "Presenting my account data - balance, savings, etc." (hereinafter the "Placebo Feature") to emulate a similar manipulation and to control for manipulation-related confounding. All participants were asked to think about their assigned feature and to write on paper, in a given space of two lines, their thoughts and comments about it.

Step 3 - Eighteen minutes were dedicated to the third step, which manipulated the *I-designed-it-myself* effect by asking participants to *specify* in detail, given general instructions, the logic and screen layout of one of the nine features of the remote-banking system. Half of the participants, who were grouped under the I-designed-it-myself condition (I-designed-it-myself manipulation = Yes), were asked to specify the Experimental Feature, while the others (I-designed-it-myself manipulation = No) were asked to specify the Placebo Feature.

Step 4 - Eighteen minutes were dedicated to the fourth step, which manipulated the *IKEA* effect by asking participants to *construct* one of the nine features of the remote-banking system. Participants were given scrambled pseudo-code of the algorithm associated with the feature assigned to them and were asked to construct the pseudo-code according to guidelines, placing its lines in the right order. Half of the participants, who were grouped under the IKEA condition (IKEA manipulation = Yes), were required to construct the Experimental Feature, while the others (IKEA manipulation = No) were required to construct the Placebo Feature.

Step 5 - Five minutes were dedicated to the fifth and last step of the experiment, which involved filling up a two-part questionnaire. In the first part, after being informed about management's intention to reduce project scope due to resource constraints, participants were asked to help management by re-ranking the nine features in descending importance order,

as performed in the baseline task in Step 1. In the second part, participants were asked to respond to demographic and background questions such as gender and age.

To avoid a sequence effect, the description of Steps 3 and 4 applies to half of the participants. While half of the participants were exposed (as described above) to the I-designed-it-myself manipulation in Step 3 and to the IKEA manipulation in Step 4, the other half were exposed to the alternative sequence: the IKEA manipulation in Step 3 and the I-designed-it-myself manipulation in Step 4. With no statistically significant differences found between the data collected under these two sequences, the sequencing dimension was ignored in data analysis that yielded the results presented in the next section.

The three independent variables (<Figure 1>) were manipulated in Steps 2, 3, and 4. In each step, half of the participants experienced the endowment, I-designed-it-myself, or IKEA effect towards the Experimental Feature by respectively being responsible for, specifying, or constructing it. In parallel, the other half of the participants performed exactly the same tasks for the Placebo Feature. Thus, the differences across the two groups in each step were related to the assigned feature only, whether Experimental or Placebo.

The Δrank dependent variable measured the change in the perceived ranking of the Experimental Feature before and after manipulation. This dependent variable therefore measured the change in perceptions toward a specific feature (Experimental Feature), when the manipulations were either on this feature or on a different feature (Placebo Feature). Importantly, each manipulation was independent of the other two manipulations, yielding eight possible combinations of the three independent variables.

Ⅳ. Results

Data were collected from 86 participants (40% female), most of whom were in their late 20s (37% aged 20 to 25; 63% aged 26 to 30). Due to missing data for two participants, the usable responses analyzed were received from 84 participants, being distributed in each of the eight experimental conditions as depicted in <Table 1>, reflecting the 2×2×2 factorial design used to test the research model in <Figure 1>.

Prior to manipulations, in Step 1 of the experiment, participants positioned the Experimental Feature out of all the nine features in the seventh place of importance (ranking mean of 5.79), representing their objective valuation of the feature. Overall, participants who were exposed to at least one effect (i.e., assumed responsibility for, designed, or constructed the Experimental Feature) ranked the Experimental Feature in the fourth place (ranking mean of 5.23) in Step 5 of the experiment, with a statistically sig-

<Table 1> Participant Distribution Among the Eight Experimental Conditions

	I-designed-it-myself = Yes		I-designed-it-myself = No		
	IKEA = Yes	IKEA = No	IKEA = Yes	IKEA = No	
Endowment = Yes	12	10	12	10	Endowment = Yes : 44
Endowment = No	10	10	10	10	Endowment = No : 40
	I-designed-it-myself = Yes : 42		I-designed-it-myself = No : 42		
	IKEA = Yes : 44; IKEA = No : 40				

nificant difference between the before and after rankings ($t(73) = 2.855$, $p = 0.006$). Over 90% (76 out of 84) of the participants reported that they regularly use remote access to their bank account, implying pre-existing familiarity with the usage and functionality of the nine features referred to in the experiment and ruling out the possibility that familiarity accounted for our results. To improve the comprehensibility of the results, given that a negative measure of Δrank represents an improvement in value assessment, the arithmetic sign of Δrank and its *t*-statistics were reversed (multiplied by -1) so that increased value assessments are positive.

4.1. Ranking the Experimental Feature Before and After Manipulation

<Table 2> presents descriptive statistics for the ranking means of the Experimental Feature before and after manipulation for participants in each of the eight experimental conditions. With a lower figure representing a higher ranking position, the only group in which the ranking mean of the Experimental Feature after manipulation reflected a decreased ranking position compared to its ranking mean before manipulation is the group without any effect manipulation (Endowment = No; I-designed-it-myself = No; IKEA = No).

<Table 3> relates to each of the three effects as a main effect and it shows the ranking means of the Experimental Feature, before and after manipulating each effect, for participants who were exposed to that effect (effect = Yes; worked on the Experimental Feature) and for those who were not (effect = No; worked on the Placebo Feature) across the different conditions of the other two effects. *t*-test results for the differences between the before and after ranking means are provided as well, demonstrating that where effect = Yes, all three effects generated statistically significant ranking improvements for the Experimental Feature, supporting H1 through H3. Where effect = No, in contrast, none of the differences between the before and after measures in all groups were statistically significant.

4.2. Comparing Δrank across Features

This sub-section presents the results related to

<Table 2> Means of Before and After Ranking for the Experimental Feature

	I-designed-it-myself = Yes				I-designed-it-myself = No			
	IKEA = Yes		IKEA = No		IKEA = Yes		IKEA = No	
	Before	After	Before	After	Before	After	Before	After
Endowmen t = Yes	6.33	5.92	5.70	4.70	6.75	5.42	6.00	5.80
Endowment = No	6.60	5.80	4.70	3.30	5.50	5.50	4.40	4.70

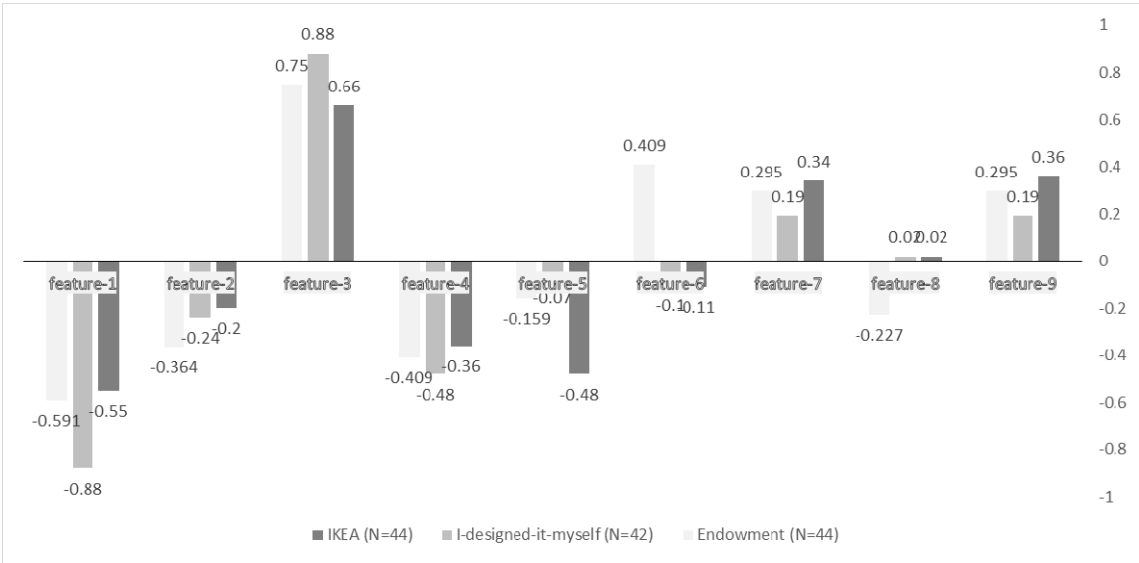
<Table 3> Statistics for the Before and After Ranking Means of the Experimental Feature

Effect	effect = Yes; Experimental Feature			effect = No; Placebo Feature		
	Before	After	*t*-test	Before	After	*t*-test
Endowment	6.23	5.48	$t(43) = 2.660$, $\mathbf{p = 0.011}$	5.30	4.83	$t(39) = 1.216$, $p = 0.231$
I-designed-it-myself	5.86	4.98	$t(41) = 2.348$, $\mathbf{p = 0.024}$	5.71	5.36	$t(41) = 1.245$, $p = 0.220$
IKEA	6.32	5.66	$t(43) = 2.048$, $\mathbf{p = 0.047}$	5.20	4.63	$t(39) = 1.631$, $p = 0.111$

the comparison between the Δrank of the Experimental Feature and the Δranks of all other features for each of the three effects. It is worth mentioning that while the previous sub-section has compared data obtained from different groups of subjects with regard to the same feature (Experimental Feature), the current sub-section compares data regarding different features obtained from the same subjects.

Comparing the change between the before and after ranks of the Experimental Feature to the change in ranks of the other eight features, <Figure 2> presents the Δrank means for Feature 1 through Feature 9, including Feature 3 (Experimental Feature), for each effect (represented by a different grey color). After manipulating the endowment, I-designed-it-myself, or IKEA effects, <Figure 2> shows that the mean Δrank for the Experimental Feature represents an increment to ranking order by 0.75, 0.88, and 0.66, respectively. In contrast, the mean Δrank for each of the other features represents either a lower increment or a decrement.

<Table 4> elaborates on the differences observed in <Figure 2> by presenting a comparison between mean Δrank for the Experimental Feature and for the average of the means for the other eight features obtained by the same participants for each of the three effects. Again, each effect is considered separately as a main effect, across the different conditions of the other two effects. All comparisons yield statistically significant differences, strengthening the support for H1 through H3 by eliminating possible alter-



<Figure 2> ΔRank Means Under the Three Effects for Each of the Nine Features (Feature 3 is the Experimental Feature)

<Table 4> Statistics for Δrank Means

Effect	Experimental Feature	Other eight features	*t*-test
Endowment = Yes	0.75	-0.09	$t(43) = 2.661$, $\boldsymbol{p = 0.011}$
I-designed-it-myself = Yes	0.88	-0.17	$t(41) = 2.484$, $\boldsymbol{p = 0.017}$
IKEA = Yes	0.66	-0.12	$t(43) = 2.126$, $\boldsymbol{p = 0.039}$

native explanations for the before-after improvements, such as a confounding effect of repeated measurement or a mood effect in which value assessments increase as a result of a general enjoyment caused by performing the experimental tasks.

Ⅴ. Discussion and Conclusion

This study addresses the roots of over-requirement in software development from the behavioral economics perspective by demonstrating the negative implications of the endowment, I-designed-it-myself, and IKEA effects in the software development context. The results of this study show that participants came to overvalue an over-required software feature that they assumed responsibility for, specified, or constructed and positively changed its importance ranking position among nine features by advancing it on average from seventh to fourth place. This mean increase in ranking, recorded when participants were made aware of the need to reduce project scope, points to overall decreased willingness to exclude that feature from scope due to the behavioral effects.

Additionally, a positive impact of each of the three behavioral effects on the perceived valuation of the Experimental Feature was demonstrated, confirming the three hypotheses put forth in Section II. To help control for possible confounding variables, participants under no-manipulation conditions were assigned to the Placebo Feature, assuring that subjects in all the eight experimental groups passed through a similar procedure and experienced similar emotions and cognitive efforts.

These findings contribute to practice by alerting practitioners that, similar to ownership feelings associated with overvaluation due to merely touching an object (Peck and Shu, 2009), just assuming responsibility for a software feature might be associated with overvaluation. The same holds for specifying or constructing a feature. The advice emerging from this work for practitioners is to be aware that behavioral effects may impede the objectivity of judgement regarding feature value by software developers. In other words, asking developers to be responsible for, specify, or construct a software feature ties them emotionally to that feature and leads them to overvalue that feature at the expense of other features. The phenomenon of over-requirement may occur upon expecting them to be objective in their recommendations about project scope, particularly about scope reduction.

While this study opens avenues for further exploration of software development processes from the behavioral economics perspective, it is important to acknowledge its limitations. One limitation is related to the use of students as participants and is somewhat mitigated by the fact that most participants were in their late 20s, toward the end of their academic studies, and already held part-time jobs in industry. Thus, the findings of the current study can be generalized to some extent due to temporal proximity to the target population (Compeau et al., 2012). Also, the procedure of the current experiment, the cover stories, and the tasks to be performed were designed to confront participants with tasks, emotions, and dilemmas of real software development circumstances. Under this setting, findings about the influence of cognitive biases on decisions can be generalized from students to experts as both groups are likely to have similar cognitive limitations and are likely to exhibit bounded rationality in decision making (Seddon and Scheepers, 2012).

Another limitation of this study is related to using the plan-based methodology as opposed to agile methodologies. Due to the generality of the behavioral

effects, however, it is reasonable that the findings are applicable to agile development because attachment toward a specified feature might influence one's judgment upon facing the need to reduce the content of an agile iteration. This methodological limitation can be overcome by customizing an experimental setting to agile development scenarios and both limitations can be overcome in further research by examining real agile projects.

Another direction for further research relates to factors found to impact the magnitude of behavioral effects, which can be addressed in future experiments and even in real software development projects. For example, Bühren and Pleßner (2014) demonstrated how the magnitude of the endowment effect is influenced by the way people gain possession over an object: simply winning it in a lottery, working in order to obtain it, or it being a reward for their successful work.

Another example relates to the observation of Norton et al. (2009) that perceived difficulty of a construction task is an important factor and that the task at hand should be difficult enough but not too difficult for the IKEA effect to emerge. When the construction task is too difficult, attachment can be thwarted (Lowry et al., 2011) as increase in perceived value emerges due to feelings of pride and competence (Mochon et al., 2012).

Franke et al. (2010) found regarding the I-designed-it-myself effect, for instance, that feeling of perceived contribution, elevated by the availability of more design freedom, positively impacts the perceived value of a self-designed product. Likewise, due to the same effect, availability of more alternatives and more features during product design were shown to lead to higher product value (Dellaert and Stremersch, 2005), and the level of perceived contribution was found to be one of the motivational factors for the voluntarily participation of software developers in open-source Linux development processes (Hertel et al., 2003). In addition, the effort invested in the process seems to play a role regarding the magnitude of both the IKEA effect and I-designed-it-myself effect, where the ease of a self-designed process may harm the feeling of being the originator and lead to a decrease in perceived value (Franke et al., 2010), whereas much effort may increase perceived value (Franke and Schreier, 2010; Moreau et al., 2011). All these factors can be taken into consideration in future research, testing their impact on the magnitude of behavioral effects in software development.

In conclusion, our findings suggest that emotional attachment to specific software features, generated in the course of assuming responsibility for, designing, or constructing those features in the process of software development, may lead to biased assessments of feature importance and may result in the inclusion within project scope of features that otherwise would not be included. These findings confirm that behavioral economics can serve as a theoretical basis for attributing performance problems in software development to systematic behavioral biases. This study is therefore among the first to draw theoretically and methodologically on behavioral economics to shed light on the behavioral roots of anomalies in software development, such as over-requirement.

<References>

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

[2] Anton, A. I., and Potts, C. (2003). Functional Paleontology: The Evolution of User-Visible System Services. *IEEE Transactions on Software Engineering*, *29*(2), 151-166.

[3] Ariely, D., and Jones, S. (2008). *Predictably Irrational: The Hidden Forces that Shape our Decisions*. Harper Collins, New York.

[4] Baccarini, D., Salm, G., and Love, P. (2004). Management of Risks in Information Technology Projects. *Industrial Management and Data Systems*, *104*(4), 286-295.

[5] 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*, *1996*(3), 116-127.

[6] Bernstein, L. (2012). Things I Learned from Taming Software Development. *ACM Sigsoft Software Engineering Notes*, *37*(6), 5-6.

[7] Boehm, B. (1996). Anchoring the Software Process. *IEEE Software*, *13*(4), 73-82.

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

[9] Boehm, B. (2006). A View of 20th and 21st Century Software Engineering. *International Conference on Software Engineering*, New York, May 20-28.

[10] Boehm, B., and Lane, J. A. (2010). Evidence-Based Software Processes. *International Conference on Software Process*, Paderborn, Germany, July 8-9.

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

[12] Bühren, C., and Pleßner, M. (2014). The Trophy Effect. *Journal of Behavioral Decision Making*, *27*(4), 363-377.

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

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

[15] Coman, A., and Ronen, B. (2010). Icarus' Predicament: Managing the Pathologies of Overspecification and Overdesign. *International Journal of Project Management*, *28*(3), 237-244.

[16] Coman, A., and Ronen, B. (2009). Overdosed Management: How Excess of Excellence Begets Failure. *Human Systems Management*, *28*(3), 93-99.

[17] Compeau, D., Marcolin, B., Kelley, H., and Higgins, C. (2012). Generalizability of Information Systems Research using Student Subjects-A Reflection on our Practices and Recommendations for Future Research. *Information Systems Research*, *23*(4), 1093-1109.

[18] Cule, P., Schmidt, R., Lyytinen, K., and Keil, M. (2000). Strategies for Heading Off IS Project Failure. *Information Systems Management*, *17*(2), 65-73.

[19] Dellaert, B., and Stremersch, S. (2005). Marketing Mass-Customized Products: Striking a Balance between Utility and Complexity. *Journal of Marketing Research*, *42*(2), 219-227.

[20] DeMarco, T., and Lister, T. (2003). Risk Management during Requirements. *IEEE Software*, *20*(5), 99-101.

[21] Dominus, M. (2006). Creeping Featurism and the Ratchet Effect. *The Universe of Discourse*, 15 May (available at http://blog.plover.com/prog/featurism.html; accessed on 17 November 2015).

[22] Elliott, B. (2007). Anything is Possible: Managing Feature Creep in an Innovation Rich Environment. *International Engineering Management Conference*, Piscataway, NJ, July 29 - Aug 1.

[23] Fleischmann, M., Amirpur, M., Benlian, A., and Hess, T. (2014). Cognitive Biases in Information Systems Research: A Scientometric Analysis. *European Conference on Information Systems*,

Tel-Aviv, Israel, June 9-11.

[24] Franke, N., and Schreier, M. (2010). Why Customers Value Self-Designed Products: The Importance of Process Effort and Enjoyment. *Journal of Product Innovation Management*, *27*(7), 1020-1031.

[25] Franke, N., Schreier, M., and Kaiser, U. (2010). The "I Designed it Myself" Effect in Mass Customization. *Management Science*, *56*(1), 125-140.

[26] Hertel, G., Niedner, S., and Herrmann, S. (2003). Motivation of Software Developers in Open Source Projects: An Internet-Based Survey of Contributors to the Linux Kernel. *Research Policy*, *32*(7), 1159-1177.

[27] Heyman, J. E., Orhun, Y., and Ariely, D. (2004). Auction Fever: The Effect of Opponents and Quasi-Endowment on Product Valuations. *Journal of Interactive Marketing*, *18*(4), 7-21.

[28] Hoorens, V., Remmers, N., and van de Riet, K. (1999). Time is an Amazingly Variable Amount of Money: Endowment and Ownership Effects in the Subjective Value of Working Time. *Journal of Economic Psychology*, *20*(4), 383-405.

[29] 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.

[30] Kahneman, D., Knetsch, J. L., and Thaler, R. (1991). Anomalies the Endowment Effect, Loss Aversion, and Status Quo Bias. *The Journal of Economic Perspectives*, *5*(1), 193-206.

[31] Kahneman, D., Knetsch, J. L., and Thaler, R. (1990). Experimental Tests of the Endowment Effect and the Coase Theorem. *The Journal of Political Economy*, *98*(6), 1325-1348.

[32] Kahneman, D., and Tversky, A. (1979). Prospect Theory: An Analysis of Decision Under Risk. *Econometrica*, *47*(2), 263-291.

[33] Kaur, K., Jyoti, B., and Rani, R. (2013). Analysis of Gold Plating: A Software Development Risk. *International Journal of Computer Science and Communication Engineering*, *2*(1), 51-54.

[34] 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.

[35] Keil, M., Im, G. P., and Mähring, M. (2007). Reporting Bad News on Software Projects: The Effects of Culturally Constituted Views of Face-Saving. *Information Systems Journal*, *17*(1), 59-87.

[36] Keil, M., Tan, B., Kwok-Kee, W., and Saarinen, T. (2000). A Cross-Cultural Study on Escalation of Commitment Behavior in Software Projects. *MIS Quarterly*, *24*(2), 299-325.

[37] 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.

[38] Khatri, V., Vessey, I., Ramesh, V., Clay, P., and Sung-Jin, P. (2006). Understanding Conceptual Schemas: Exploring the Role of Application and IS Domain Knowledge. *Information Systems Research*, *17*(1), 81-102.

[39] Legoux, R., Leger, P., Robert, J., and Boyer, M. (2014). Confirmation Biases in the Financial Analysis of IT Investments. *Journal of the Association for Information Systems*, *15*(1), 33-52.

[40] Lowry, G. S., Franssen, C. L., and Lowry, J. E. (2011). Handcrafting Attachment: A User-Centered Approach. *Journal of Service Science*, *4*(1), 39-44.

[41] Malhotra, N., Bhardwaj, M., and Kaur, R. (2012). Estimating the Effects of Gold Plating using Fuzzy Cognitive Maps. *International Journal of Computer Science and Information Technologies*, *3*(4), 4806-4808.

[42] 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*(4), 11-25.

[43] McConnell, S. (1997). Achieving Leaner Software. *IEEE Software*, *14*(6), 127-128.

[44] Mochon, D., Norton, M. I., and Ariely, D. (2012).

Bolstering and Restoring Feelings of Competence Via the IKEA Effect. *International Journal of Research in Marketing*, *29*(4), 363-369.

[45] Moreau, C. P., Bonney, L., and Herd, K. B. (2011). It's the Thought (and the Effort) that Counts: How Customizing for Others Differs from Customizing for Oneself. *Journal of Marketing*, *75*(5), 120-133.

[46] NASA. (1992). *Recommended Approach to Software Development.* Goddard Space Flight Center, Greenbelt, MD.

[47] Norton, M. I., Mochon, D., and Ariely, D. (2009). The IKEA Effect: When Labor Leads to Love. *Harvard Business Review*, *87*(2), 30.

[48] Norton, M. I., Mochon, D., and Ariely, D. (2012). The IKEA Effect: When Labor Leads to Love. *Journal of Consumer Psychology*, *22*(3), 453-460.

[49] Pass, S., and Ronen, B. (2014). Reducing the Software Value Gap. *Communications of the ACM*, *57*(5), 80-87.

[50] Peck, J., and Shu, S. (2009). The Effect of Mere Touch on Perceived Ownership. *Journal of Consumer Research*, *36*(3), 434-447.

[51] Ronen, B., and Pass, S. (2008). *Focused Operations Management: Achieving More with Existing Resources.* John Wiley and Sons, Hoboken, NJ.

[52] 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.

[53] Rust, R. T., Thompson, D. V., and Hamilton, R. W. (2006). Defeating Feature Fatigue. *Harvard Business Review*, *84*(2), 98-107.

[54] 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.

[55] Seddon, P. B., and Scheepers, R. (2012). Towards the Improved Treatment of Generalization of Knowledge Claims in IS Research: Drawing General Conclusions from Samples. *European Journal of Information Systems*, *21*(1), 6-21.

[56] Shmueli, O., Pliskin, N., and Fink, L. (2015a). Can the outside view Approach Improve Planning Decisions in Software Development Projects? *Information Systems Journal*, *26*(4), 395-418. DOI: 10.1111/isj.12091.

[57] Shmueli, O., Pliskin, N., and Fink, L. (2015b). Explaining Over-Requirement in Software Development Projects: An Experimental Investigation of Behavioral Effects. *International Journal of Project Management*, *33*(2), 380-394.

[58] 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.

[59] Thaler, R. (1980). Toward a Positive Theory of Consumer Choice. *Journal of Economic Behavior & Organization*, *1*(1), 39-60.

[60] Ulrich, K. T. (2009). *Design: Creation of Artifacts in Society.* University of Pennsylvania Working Paper (available at http://ssrn.com/abstract=1951106, accessed September 5 2015).

[61] Umapathy , K., Purao , S., and Barton , R. R. (2008). Designing Enterprise Integration Solutions: Effectively. *European Journal of Information Systems*, *17*(5), 518-527.

[62] Westfall, L. (2005). The what, Why, Who, when and how of Software Requirements. *ASQ World Conference on Quality and Improvement*, Seattle, WA, May 16-18.

[63] Wheatcraft, L. S. (2011). Triple Your Chances of Project Success, Risk and Requirements. *INCOSE International Symposium*, Denver, CO, June 23.

[64] Williams, L., Kessler, R. R., Cunningham, W., and Jeffries, R. (2000). Strengthening the Case for Pair Programming. *IEEE Software*, *17*(4), 19-25.

<Appendix> Feature List

1. Presenting my account data (balance, savings, etc.)
2. Ordering new checkbooks
3. Presenting my itemized expense report for a specific period
4. Transferring money from my account to somebody else's account
5. Listing my credit card charges
6. Listing my account's transactions for the current month
7. Listing my account's transactions for the last three months
8. Requesting my transaction reports for previous years
9. Listing my deposited and withdrawn checks

◆ About the Authors ◆

Ofira Shmueli

Ofira Shmueli holds a Ph.D. in Information Systems from the Department of Industrial Engineering and Management at Ben-Gurion University of the Negev, Israel. Her B. Sc. in Mathematics and Computer Science is from the Hebrew University and her M. Sc. degree in Information Systems is from the Tel Aviv University. She has harnessed her vast practical experience as a software developer, system analyst, and project manager in her doctoral research which addressed the behavioral roots of over-requirement. Her work, showing that software systems contain functionality that is well beyond need due to behavioral biases, has been published in the *Information Systems Journal* and in the *International Journal of Project Management*.

Nava Pliskin

Nava Pliskin acquired her Ph.D. and S.M. degrees from Harvard University and her B. Sc. degree from Tel-Aviv University. She is a tenured full professor in the Department of Industrial Engineering and Management at Ben-Gurion University of the Negev, Israel, where she holds the Solomon and Abraham Krok Chair in Entrepreneurial Management. The Harvard Business School invited her during 1996-1997 to serve as a Visiting Associate Professor and hold the Thomas Henry Carroll Ford Foundation Chair. Professor Pliskin is conducting research on topics related to management and strategy of information systems at the organizational and individual levels. Her research papers have been published in such leading journals as: *Journal of the Association for Information Systems, ACM Transactions on Information Systems, Information Society, Communications of the ACM, IEEE Transactions on Engineering Management, Information & Management, Database for Advances in Information Systems, and Decision Support Systems*.

Lior Fink

Lior Fink is an associate professor of information systems in the Department of Industrial Engineering and Management at Ben-Gurion University of the Negev, Israel. He holds a bachelor's degree in psychology and economics, a master's degree in social-industrial psychology, and a Ph.D. degree in information systems from Tel Aviv University. His research interests focus on economic and strategic aspects of IT development, outsourcing, and use. Lior's articles have been published in numerous information systems journals including *MIS Quarterly, European Journal of Information Systems*, *Information Systems Journal, Journal of the Association for Information Systems, Journal of Information Technology*, and *Journal of Strategic Information Systems*. His work has also appeared in economics, project management, and medical informatics journals.

Submitted: December 30, 2015; 1st Revision: May 30, 2016; Accepted: June 7, 2016