
Swagster.com

6th **Gdansk Poland** EuroSymposium on Systems Analysis and Design (SAND)

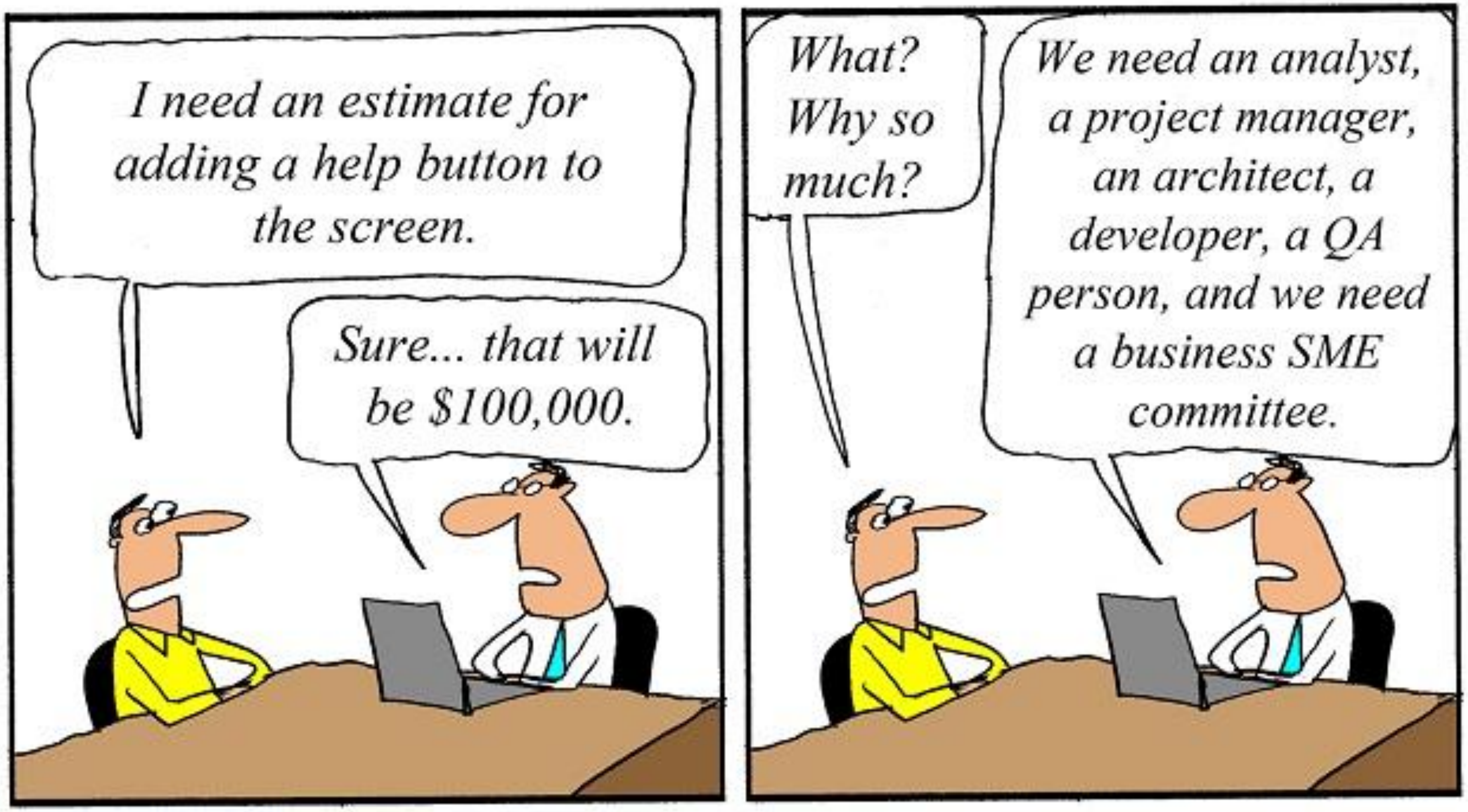
Explaining **Over-Requirement** in Software Development: Three Experiments Investigating **Behavioral Effects**

Doctoral Student: **Ofira Shmueli** OfiraSh@bgu.ac.il
Co-Advisor: Lior Fink FinkL@bgu.ac.il
Presenter: Nava Pliskin PliskinN@bgu.ac.il

Dzień dobry!

Faculty of Engineering Sciences
Department of Industrial Engineering and Management

בוקר טוב!


I need an estimate for adding a help button to the screen.
Sure... that will be $100,000.
What? Why so much?
We need an analyst, a project manager, an architect, a developer, a QA person, and we need a business SME committee.

Definition: **Over-Requirement**

Specifying a product/service beyond customer needs

(Ronen & Pass, 2008, *Focused Operations Management*, WILEY)

Other (equivalent?) terms:

- Over-Specification/Feature
- Requirement/Feature Creep
- Gold-plating

"Please refrain from asking questions until the end because we have a lot of requirements to review today."

Over-Requirement

- Gold-plating consumes extra effort, reduces software integrity

(Boehm & Papaccio, 1988, *IEEE Transactions on Software Engineering*)

- Don't gold-plate!

(NASA, 1992, Recommended approach to software development, Goddard Space Flight Center)

- Unnecessary features (“Bells & Whistles”)

(Ropponen & Lyytinen, 2000, IEEE Transactions on Software Engineering)

- Excessive requirements added are rarely cut off

(Dominus, 2006, http://blog.plover.com/prog/featurism.html)

- One of top ten risks in software development projects

(Boehm, 1991, *IEEE Software*; Schmidt et al. 2001, *JMIS*; Naz & Khokhar, 2009, *ICCMS IEEE*; Suresh, 2011, *International Journal of Research and Reviews in Software Engineering*)

Major risk/concern

Nevertheless, lack of research !

"It takes IT a long time to implement our requirements. Sometimes I just give them requirements we don't even need. By the time they implement them, who knows?"

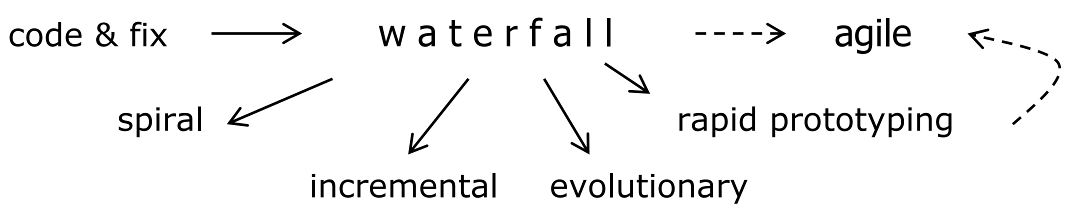
Software Development Facts

- "Software Hall of Shame"
  30 software-development mega-projects became mega-failures (1992-2005)

(Charette, 2005, *IEEE Spectrum*)

- Software-development evolution

(Boehm, 2006, *ACM*; Boehm & Turner, 2003, *IEEE Computer*; www.agilemanifesto.org)



Yet, software projects continue to fail

(highest failure rate in first decade of 21st century)

24% completely abandoned, 44% significantly over budget/scedule

(2009 Standish Group Report)

"Based on our tests, the business stakeholders fall asleep around page 37 of the Functional Requirements Specification. Put the Issues Section on page 40."

Software Development Risks

Over-Requirement

R*elated* risk factors

Requirement quality

Project size

Requirement Quality Risk

- System of poor requirement quality is likely to fail or to malfunction

(Fredrick Brooks, 1975, *The Mythical Man-Month*, Adison-Wesley)

- Poor requirement definition is one of the reasons for failure

(Charette, 2005, *IEEE Spectrum*)

- High probability * High impact

(Han & Huang , 2007, *The Journal of Systems & Software*)

Project Size Risk

- Major risk dimension

(McFarlan, 1981, *HBR*; Zmud, 1980, *MIS Quarterly*)

- Risk is an increasing function of project size

(Barki, Rivard & Talbot, 1993, *JMIS*; Glass, 1998, *JS&S*; Houston, Mackulak & Collofello 2001, *JS&S*)

- Large-scale projects
  - Fail three to five times more than smaller ones

(Charette, 2005, *IEEE Spectrum*)

  - Prone to unexpected colossal events

(Flyvbjerb & Budzier, 2011, *HBR*)

Over-Requirement Damages

- Defocusing and distraction
- Wasted Resources due to increased development efforts
- Increased project complexity
- Delayed launch
- Reduced user satisfaction since software is
  - Complex
  - Defective
  - Unreliable
  - Difficult to manage
  - Costly to maintain
  - Without core features due to cutoffs aimed to meet time/budget constraints
- Loss of entire (supplier/customer) company

References to Over-Requirement Damages

Battles,Mark & Ryan, 1996, *The McKinsey Quarterly*

Westfall, 2005 *ASQ World Conference on Quality and Improvement Proceedings*

Rust,Thompson & Hamilton, 2006, *Harvard business review*

Elliott, 2007, *IEEE International Engineering Management Conference IEMC*

Kautz, 2009, *Journal of Information Technology Theory and Application*

Buschmann, 2009, *IEEE Software*

Coman & Ronen 2009, *Human Systems Management*

Buschmann, 2010, *IEEE Software*

Coman & Ronen, 2010, *International Journal of Project management*

Research Motivation

- High failure rates in software development projects

- Large-scale projects much more **prone to unexpected colossal events**, even bringing organization down (Flyvbjerb & Budzier, 2011, Harvard business review)

- Analysts at PwC (PricewaterhouseCoopers) were quoted at a 26.02.12 conference http://www.cio.com/article/158356/Strategies_for_Dealing_With_IT_Complexity :

"IT complexity acts as a significant tax on IT value"

→ How to reduce Over-Requirement?

Over-Requirement Causes

Related to **developers**…

- Ignore business requirements for sake of technology
- Develop unauthorized features to satisfy their own interest
- Wish for best possible solution
- Desire to fulfill all future needs, add just-in-case functionality
- Do not know which features will eventually be important
- Have misconceptions:
  - Underestimate cost during specification
  - Waste due to attitudes toward time & material contracts

More Over-Requirement Causes

Related to **users** or **managers**…

- All-or-nothing attitude of users
- User diversity
  - Can one system fit all?
  - Can users cope with releases on a continuous basis?
- Managers do not enforce time or budget constraints
- Politics

Common to many causes → ***Human Behavior***

Over-Requirement Causes

References

Anton & Potts 2003, *IEEE Transactions on Software Engineering*

Boehm 1996, *IEEE Software*

Boehm & Papaccio, 1988, *IEEE Transactions on Software Engineering*

Buschmann, 2009, *IEEE Software*

Buschmann, 2010, *IEEE Software*

Coman & Ronen, 2010, *International Journal of Project management*

Cule, Schmidt, Lyytinen & Keil, 2000, *Information Systems Management*

DeMarco &Lister, 2003, *IEEE Software*

Kautz, 2009, *Journal of Information Technology Theory and Application*

Gary 2009, http://www.sentrum.com/news-information/press-releases/item/poor-industry-consultancy-threatens-data-centre-constructions/

Kemerer 1987, *Communications of the ACM*

Koopman 2010, *WESE*

Koopman, 2011, *Embedded Systems Conference Silicon Valley*

McConnell 1997, *IEEE Software*

Miranda & Abran, 2008, *Project Management Journal*

Ropponen & Lyytinen, 2000, *IEEE Transactions on Software Engineering*

Rust ,Thompson & Hamilton, 2006, *Harvard business review*

Schmidt, Lyytinen, Keil & Cule 2001, *Journal of Management Information Systems*

Westfall ,2005*ASQ World Conference on Quality and Improvement Proceedings*


This is a very important project. Can I just stay until I finish capturing ALL the business requirements?
OK! But not a day longer!
Great! Larry's going to live forever!

Prospect theory – Kahneman & Tversky

(1979: An analysis of decision under risk. *Econometrica*)

The Endowment Effect

People place higher **valuation** on objects they own

(Thaler, 1980, *Journal of Economic Behavior & Organization*)

- Holds beyond physical goods

  (Kahneman, Knetsch, & Thaler, 1990, *Journal of Political Economy*)
- Holds for imaginary and real possessions

  (Heyman, Orhun, & Ariely, 2004, *Journal of Interactive Marketing*)

Ownership *duration* has a positive impact on valuation

(Strahilevitz & Loewenstein, 1998, *Journal of Consumer Research*)

The I-Designed-It-Myself Effect – Ariely

(2008: *Predictably Irrational*)

Value gained due to

psychological benefit

of self-specification

(Franke, Schreier, & Kaiser, 2010, *Management Science*)

- People overvalue self-specified objects
- Task *freedom* plays a major role

The IKEA Effect – Ariely

(2008, *Predictably Irrational*, Harper New York)

Value gained due to Self-assembly

("The IKEA Effect: When Labor Leads to Love"
Norton, Mochon, & Ariely, 2009; 2012, *Harvard Business Review; Journal of Consumer Psychology*)

- People overvalue their own creations when labor is fruitful
- Task *difficulty* plays a major role in the IKEA effect

The Planning Fallacy – Kahneman & Tversky

- People **underestimate** the *time* needed to complete task

(Kahneman & Tversky, 1979, *TIMS Studies in Management Science*)

- *Uninvolved* observers **overestimate** the time to completion

(Buehler, Griffin, & Ross, 1995, *European review of social psychology*)

- Applicable beyond the time resource…

(Lovallo & Kahneman, 2003, *Harvard business review*)

Underestimation of time, costs, and risks

Overestimation of benefits

Research Agenda

Research Objectives

- Gain understanding of Over-Requirement roots
- Explore Over-Requirement via Behavioral Economics perspective

Research Question

Do Behavioral Effects explain Over-Requirement?

Research Hypotheses (Sample)

1. The IKEA effect positively affects Over-Requirement
2. The Endowment effect positively affects Over-Requirement
3. The Planning Fallacy positively affects Over-Requirement

Research Methodology

Three Experiments

– Factorial design

Representing behavioral effects

– Participants

Advanced undergraduate IS-major IE&M students


We have a big issue: we can't figure out which requirements are the critical ones.
I know this might sound kooky, but what if we ask the customer?

Methodology – 1st Experiment

Factorial 2×2×2 design, representing 3 variables

- Specification **duration** (10/30 minutes - manipulated)
- Specification **freedom** (low/high - manipulated)
- **Challenge** feeling (low/high - not manipulated, measured)

– Three steps

– One hour long

3 Steps of 1st Experiment

1. **Questionnaire A**
   - Case story – software system for remote-banking clients
   - Participants asked to evaluate the importance of 16 features

2. **Specification process**
   (10/30 minutes duration × low/high freedom manipulations)
   - One **same** feature for all participants
   - Deliberately chosen to be a nice-to-have feature

3. **Questionnaire B**

   Participants are asked to
   - Re-evaluate importance
   - Report various feelings

     (including challenge feeling regarding specification task)
   - Answer demographic and background questions

Objectives – 1st Experiment

- Investigate the IKEA Effect (and Endowment Effect)
- For a certain specified **Over-Required** feature, measure the change in perceived valuation

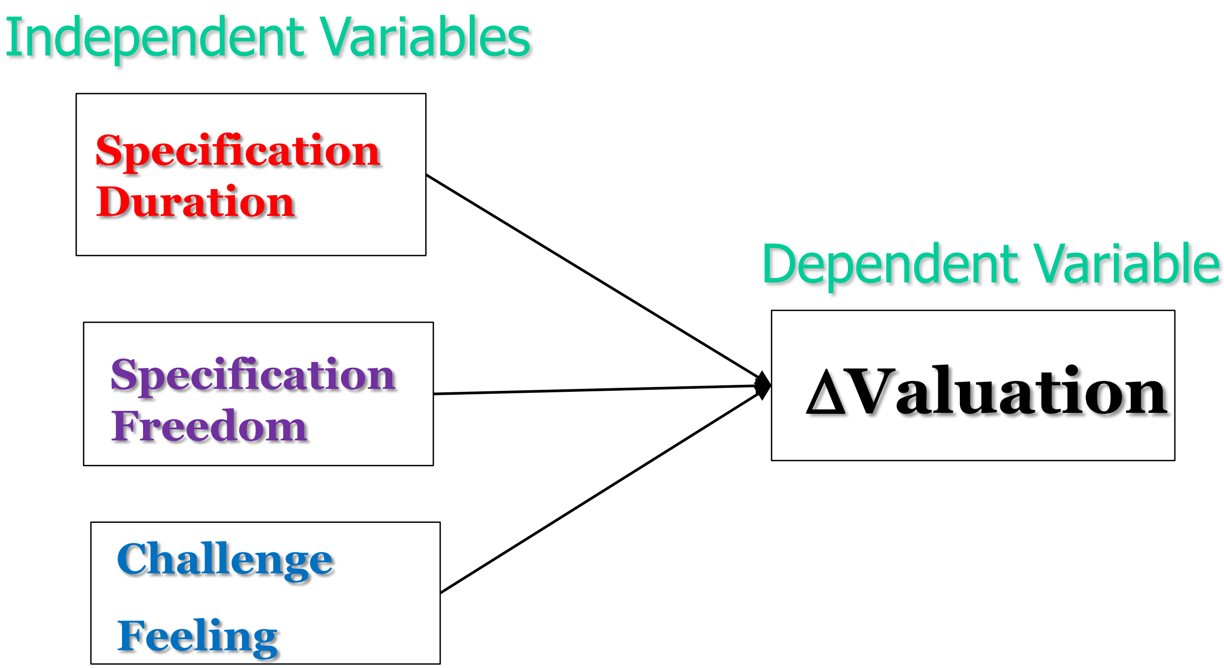
Valuation

ΔValuation = Difference between two measures:

- After specification valuation measure – at Stage 3
- Before specification valuation measure – at Stage 1

Each **valuation** measure is on a continuous importance scale from 0 = **Not Important** to 100 = **Very Important**

Research Model #1

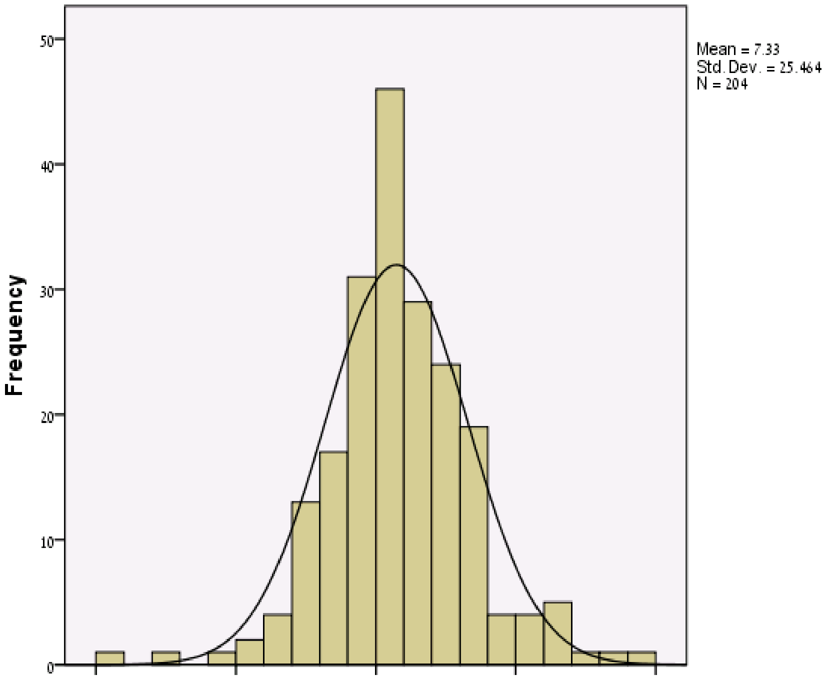


Feature Valuation

Significant difference between **After** and **Before** valuations → **IKEA**

	Minimum	Maximum	Mean	Median	Std. deviation
Δ Valuation	-94	95	7.3	5.5	25.4

ΔValuation



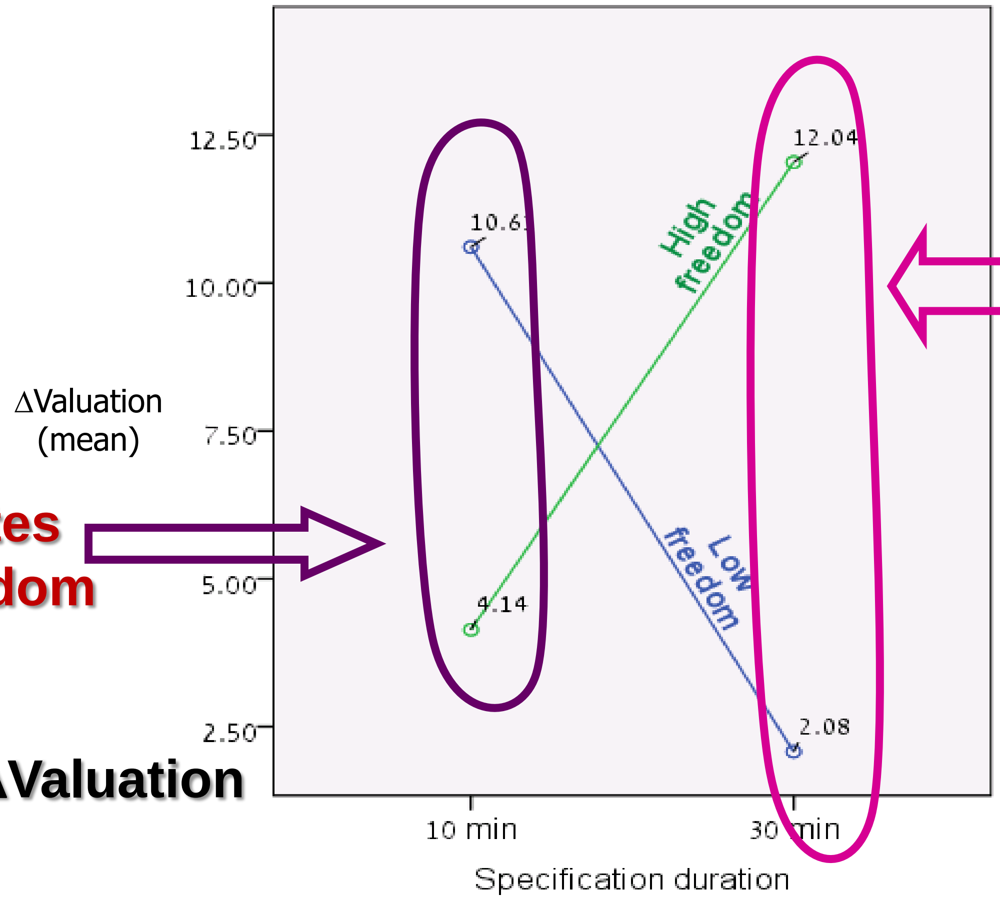
Cell means for ΔValuation

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)

Cell means for ΔValuation

Estimated marginal means are shown with std. errors in parentheses

Specification freedom & duration



10 minutes low freedom

↓

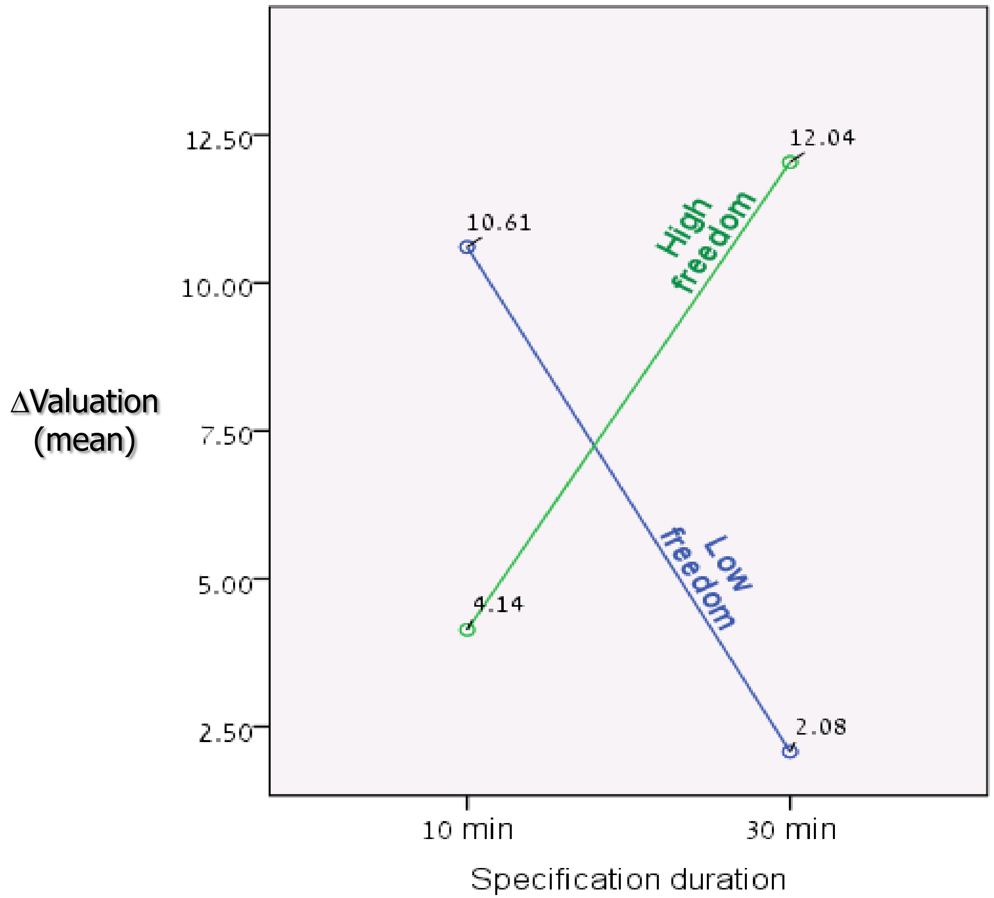
higher ΔValuation

30 minutes high freedom

↓

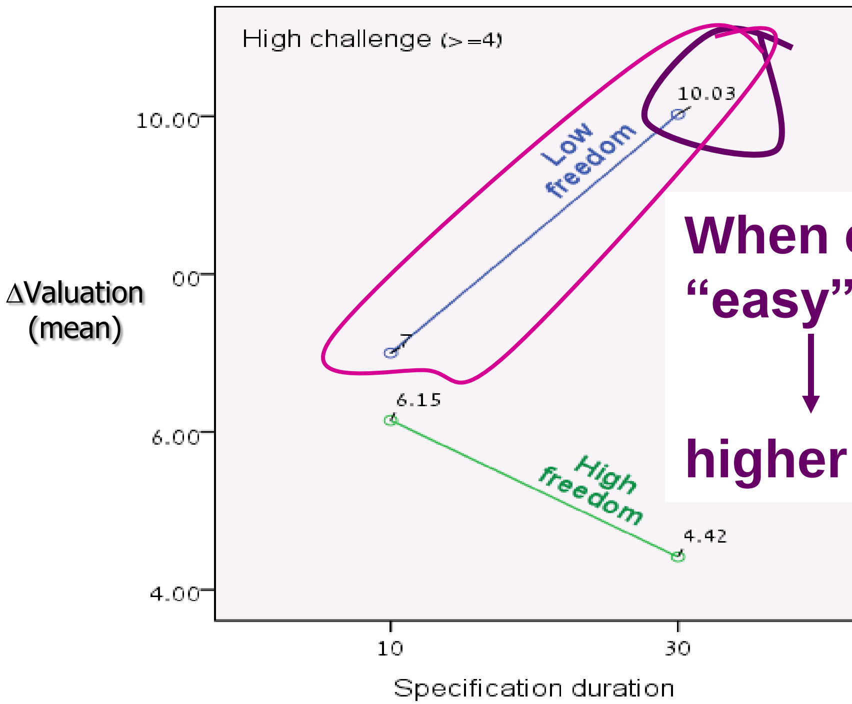
higher ΔValuation

Specification freedom & duration



Freedom, duration & challenge

Challenged participants

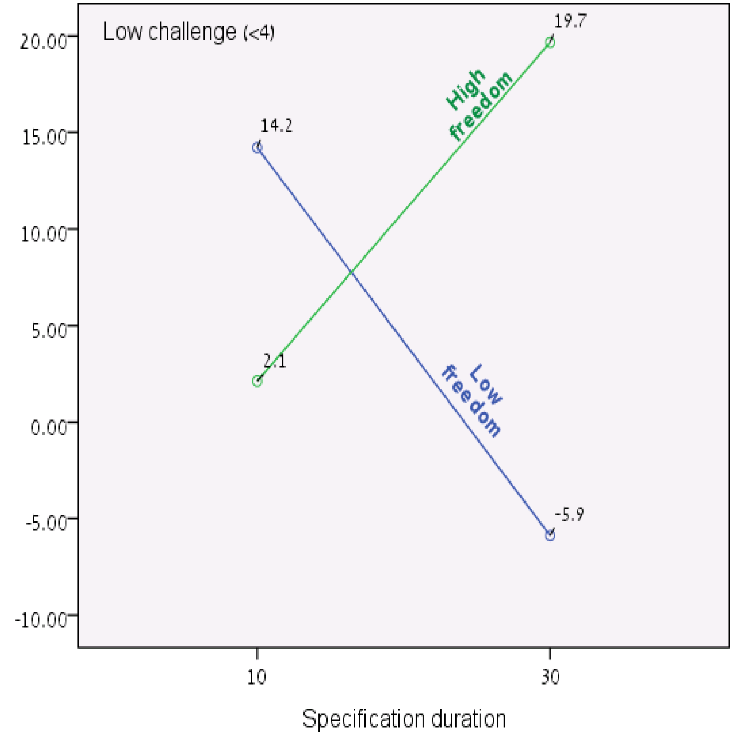


When challenged "easy" conditions

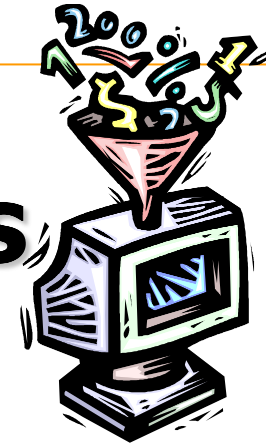
↓

higher ΔValuation

Unchallenged participants



1st Experiment – Conclusions



- Feature specification leads to emotional attachment
    - Developers are biased due to the IKEA effect

- The IKEA Effect is more complex than described in the literature because it is related to
    - objective difficulty (duration, freedom)
    - subjective difficulty (challenge)


Our goal this year is to find and fix as many functional specification errors as possible...
How many errors do you need?

Methodology – 2nd Experiment

– Factorial design 2×2 design, representing 2 variables

- **Previous Knowledge** (with / without - manipulated)
- **Role** (software developer / software consultant - manipulated)

– Four steps

– Half an hour long

4 Steps of 2nd Experiment

1. Background

(developer/consultant role manipulation)

- Case story – software project for building three towers by robot
- A list of 16 optional features (different importance)

2. Time estimation

(developer/consultant role × with/without previous knowledge manipulations)

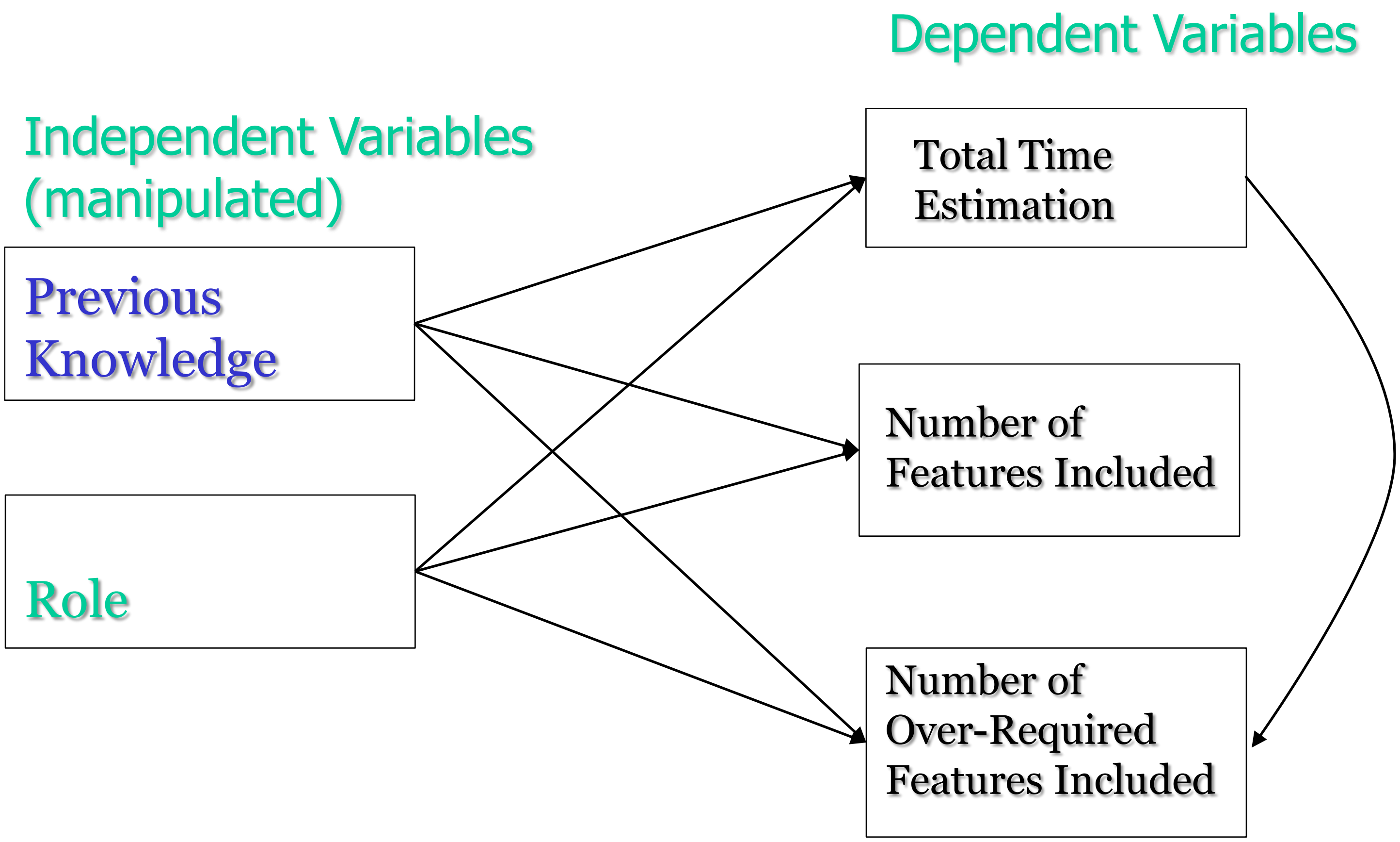
- Development time for each of 16 features
  - time to develop by self (if in a developer role)?
  - time to be developed by colleague (if in a consultant role)?

3. Project Scoping

- Considering their earlier time estimations in #2 and given project duration constraint (18)
  - what features to include in project scope?

4. Final Questioning – feelings, attitude, demographic questions

Research Model #2



Dependent Variables

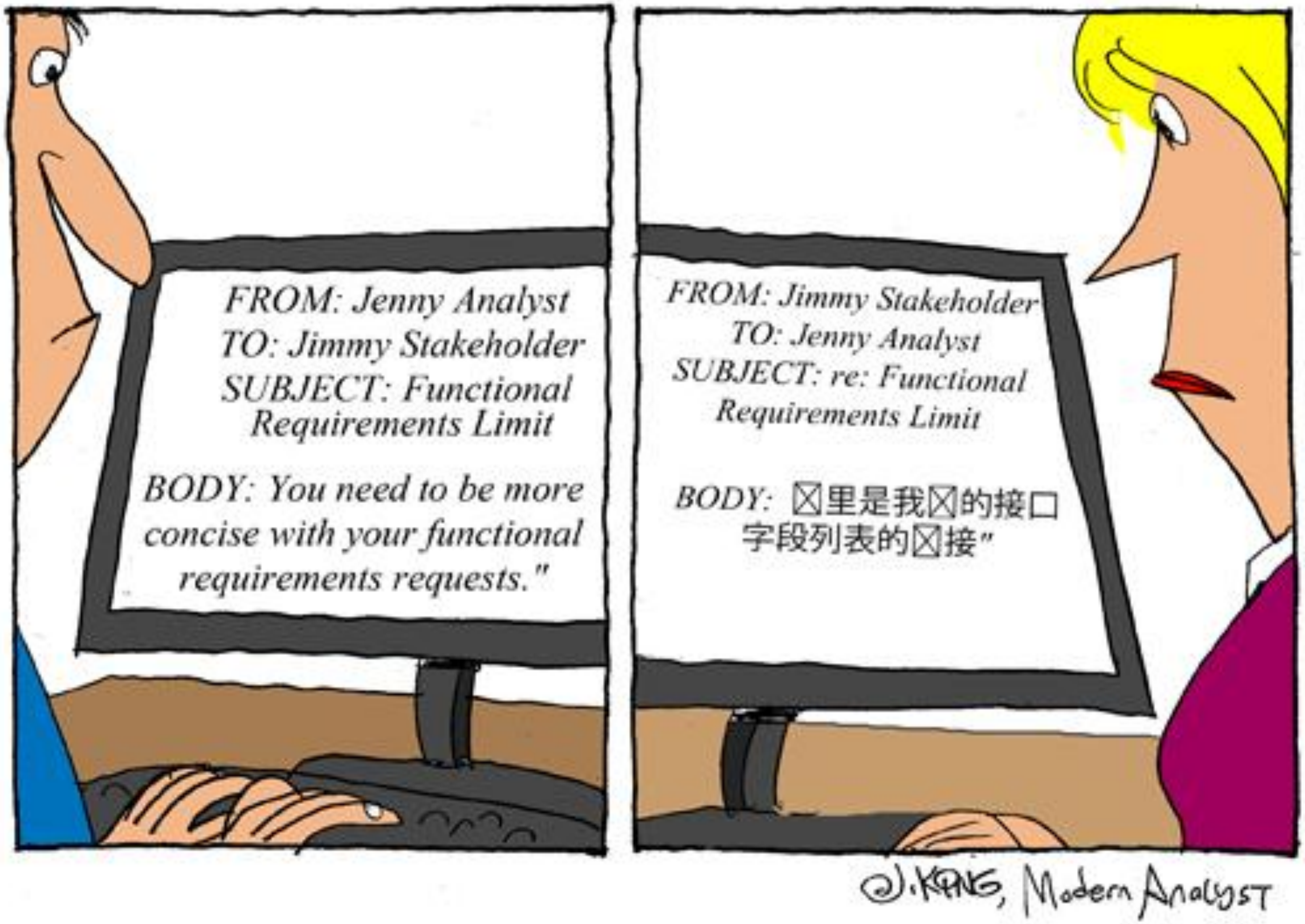
- Total Time Estimation
- Number of Features Included
- Number of Over-Required Features Included
  (out of five features determined earlier as unnecessary by two course instructors)

2nd Experiment – Conclusions

Knowledge and role affect the planning fallacy

- Previous knowledge about development times in the past reduces:
  - Time underestimation
  - Scope overloading
  - Over-Requirement
- Role plays a role:

  Compared to consultants, developers tend to include in project scope
  - More features
  - More Over-Required features
- Lower time estimations are associated with more Over-Requirement


FROM: Jenny Analyst
TO: Jimmy Stakeholder
SUBJECT: Functional Requirements Limit
BODY: You need to be more concise with your functional requirements requests."
FROM: Jimmy Stakeholder
TO: Jenny Analyst
SUBJECT: re: Functional Requirements Limit
BODY: ☒里是我☒的接口字段列表的☒接"
©J.KANG, Modern Analyst

Methodology – 3rd Experiment

- Factorial 2×2×2 design, representing 3 variables
  - **Endowment** (with / without - manipulated)
  - **I-Designed-it-Myself** (with / without - manipulated)
  - **IKEA** (with / without - manipulated)
- Five steps:
  - 1) Start, 2) Task A, 3) Task B, 4) Task C, 5) Finish
  - Two Features: X, Y
  - Parts of Feature X / Y were assigned in Tasks B / C
- One hour long

5 Steps of 3rd Experiment

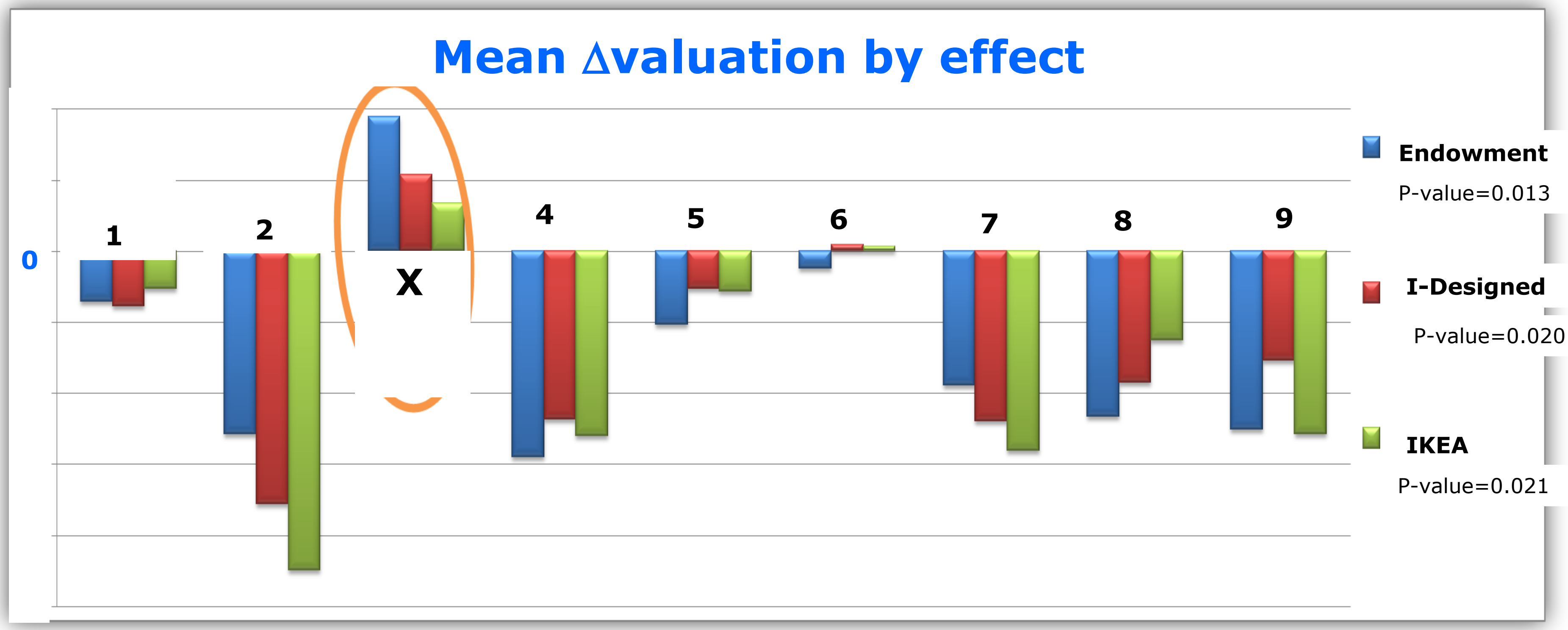
Background story • Development of a software system for remote-banking clients

Three Tasks • Half of the participants performed each task for Feature X
Half of the participants performed each task for Feature Y

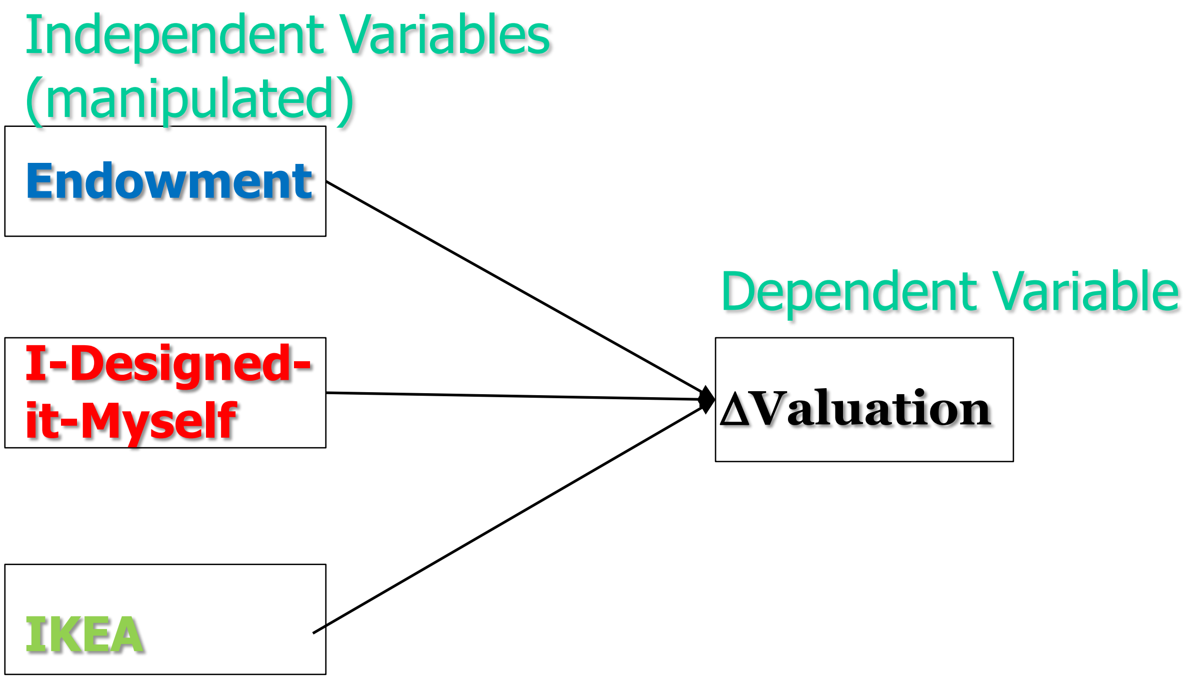
1) **START** – Participants evaluated the importance of 9 features of the system, including X and Y
2) **Task A** – **Endowment** manipulation: Participants told that the feature (half X and half Y) is "theirs" and asked to describe it in three lines of text
3) **Task B** – **I-Designed-it-Myself** manipulation: Participants asked to specify a feature (part, half X and half Y) in 2 pages
4) **Task C** – **IKEA** manipulation: Participants asked to re-arrange pseudo code for a feature (part, half X and half Y) according to instructions
5) **Finish** – Participants asked to re-evaluate importance of 9 features

We focused on ΔValuation for X as the dependent variable, whether the participant performed tasks on (nice-to-have) **X or on Y**

Findings – 3nd Experiment



Research Model #3



3nd Experiment – Conclusions

✓The Endowment Effect

✓The IKEA Effect

✓The I-designed-it-myself Effect

✓The Endowment Effect * The IKEA Effect

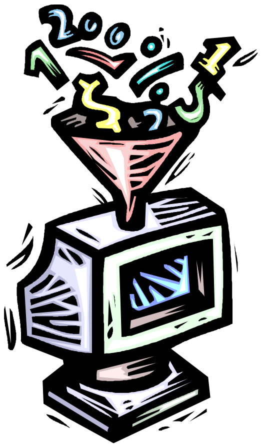
✓The Endowment Effect * The I-designed-it-myself Effect

x The IKEA Effect * The I-designed-it-myself Effect

x Three-way interaction

Doctorate Innovation

- Empirical exploration of Over-Requirement and its behavioral roots
- Investigation of behavioral effects in software development
  - An intangible process yielding an intangible product
- Consideration TOGETHER of
  - Behavioral effects
  - Related variables
  - Interactions



Expected Contributions

Research

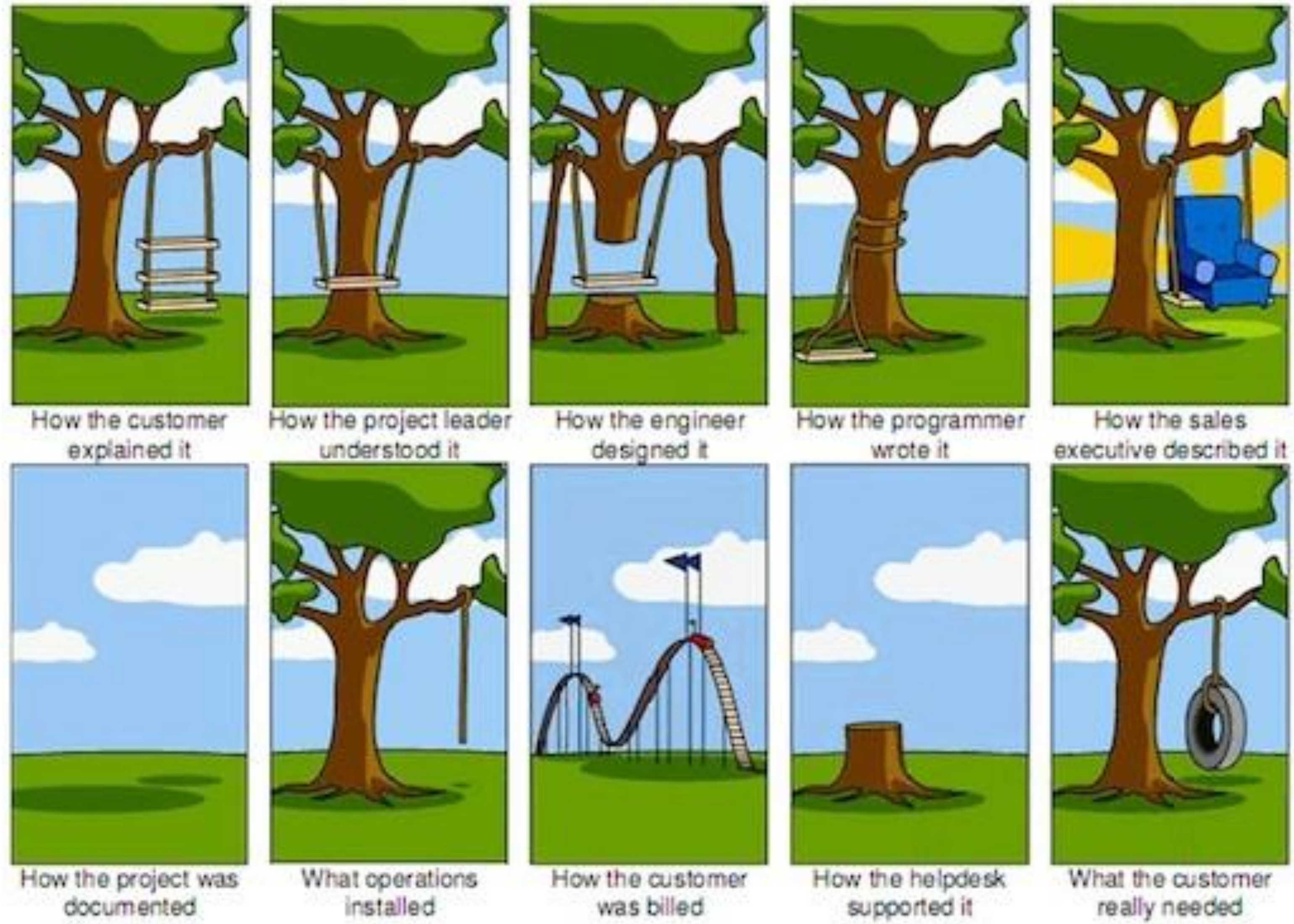
Knowledge about behavioral Over-Requirement roots

Practice

- Managers beware: Endowment/I-designed-it-myself/IKEA → emotional attachment after feature engagement → developers become attached and, hence, subjective

When labor leads to love …

- Manager awareness of behavioral Over-Requirement roots→
  - Acknowledging developer attachment and subjectivity
  - Adopting agile practices (small iterations overcome 3 effects?) since findings lend support to agile development
  - Recruiting others, like consultants or uninvolved developers (overcome planning fallacy? 3 effects?)


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed

Dziękuje!

Thanks!

Kwestia?

Questions?

Dzień dobry!