Decision-making Processes and Cognitive Biases in Designing Software Architectures

Akash Manjunath, Garching, 19.02.2018

Software Engineering betrieblicher Informationssysteme (sebis)
Fakultät für Informatik
Technische Universität München

wwwmatthes.in.tum.de
Agenda

- Introduction
- Goals
- Research Questions
- Expert Reviews
- Research Implications
What is Software Architecture?
- Blueprint about the structure of the software
- Responsibility of software architects
  - Designing the architecture involves decision-making
    - Examples: Which design patterns to choose? Which combination of technologies to use? etc.
Challenge for Software Architects

- Too many technologies to mix and match
- Decision-making
  - complex, implicit and knowledge intensive
  - process is not well understood
- Heuristics such as past experience, familiarity, trends etc. are used for decision-making
- Decisions are biased due to the use of heuristics
- Results in *sub-optimal* or *satisficing* solutions
Goals

- From the context of designing software architectures
  - To formalize the decision-making process to make it explicit
  - To understand which cognitive biases influence software architects when designing architectures
1. Which decision-making models are relevant in the context of making software architecture design decisions?

2. What is the relationship between the decision-making models and the OODA loop?

3. Which cognitive biases influence software architects when designing architectures?
Several models of decision-making
- Focus on models relevant in the context of designing software architectures

**Decision-Making Models**

- Normative Approach
  - Rational Economic Model

- Behavioral Approach
  - Naturalistic Decision-Making: Recognition Prime Model
  - Bounded Rational Model
Research Questions

1. Which decision-making models are relevant in the context of making software architecture design decisions?

2. How can we establish a relationship between the models with the OODA loop decision cycle?

3. Which cognitive biases influence software architects when designing software architectures?
OODA Loop
Observe, Orient, Decide and Act

- Decision cycle proposed by John Boyd, a military strategist
- Popular tool used by decision-makers in different fields of work
- Much research conducted on OODA Loop and DMMs, but not inside the boundaries of designing software architectures

![OODA Loop Diagram](image)

* Software Architect
** Adapted from The Journal of Software Engineering
Matrix of DMMs and OODA Loop

<table>
<thead>
<tr>
<th>Observe</th>
<th>Orient</th>
<th>Decide</th>
<th>Act</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision-making Model 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decision-making Model 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decision-making Model 3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Normative Model
Rational Economic Model **

Make decision on software architecture

Observe
Define concern / problem
Gather info about requirement
Gather and analyze info about the concern

Orient
Create alternative list

Decide
Rank alternatives using ranking algorithm
Choose optimal alternative

Act
Implement alternative
Test validity of alternative

Feedback Loop

Implicit guidance and control

* Software Architect
** Adapted from Buchanan and Huczynski (2004); Drucker (2001); Miller Hickson and Wilson (2002).
Behavioral Model
Naturalistic Decision Making – Recognition Prime Model

Observe
- Define concern / problem
- Mentally gather information

Orient
- Is situation familiar?
  - Yes, seek more info and assess
  - No, recognition
- Are expectations violated?
  - Yes, create alternative list
  - No

Decide
- Mental simulation of a sub-optimal alternative in action
- Modify
  - Will it work?
    - Yes, but not perfect
    - No

Act
- Implement Decision
- Evaluate against standards

Feedback Loop
- Implicit guidance and control
- Unfolding circumstances
- Outside Information

*Software Architect
** Adapted from: A Recognition Primed Decision (RPD) Model of Rapid Decision Making by Gary Klein (1993)
Behavioral Model
Bounded Rational **

Make decision on software architecture

Implicit guidance and control

Observe

Define concern / problem

Gather info about the concern

Orient

Create alternative list

Decide

Rank Alternatives based on heuristics

Choose satisfying alternative

Act

Implement Decision

Evaluate against standards

Feedback Loop

* Software Architect
**Adapted from Decision-making in practice: The use of cognitive heuristics by senior managers by Mark Crowder (2013)
Research Questions

1. Which decision-making models are relevant in the context of making software architecture design decisions?
2. How can we establish a relationship between the models with the OODA loop decision cycle?
3. Which cognitive biases influence software architects when designing software architectures?
Cognitive Biases

- What are cognitive biases?
  - Systematic deviation from rationality in judgement
  - Due to limitations in human cognitive capacity
  - Impacts decisions and judgements
- Over 200 types of cognitive biases
- Not all of them are relevant for designing software architecture

33 cognitive biases recognized as relevant from the context of making architectural design decisions
Cognitive Biases Classification (1/2)
Two-level classification

Cognitive Biases

Observe Phase
- Information Gathering Biases
- Information Presentation Biases

Orient Phase
- Information Filtering Biases
  - Biases due to resemblance / parallelism
  - Biases due to previous knowledge / experience
  - Biases due to trends

Decide Phase
- Biases due to complexity
- Biases due to nature of invention/trends

Act Phase
- Biases due to previous knowledge / experience
- Biases related to strategy making

Level 1 Classification
Level 2 Classification
### Cognitive Biases Classification (2/2)

<table>
<thead>
<tr>
<th>Observe Phase</th>
<th>Orient Phase</th>
<th>Decide Phase</th>
<th>Act Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Gathering Biases</td>
<td>Information Filtering Bias</td>
<td>Biases related to Complexity</td>
<td>Misinformation Effect</td>
</tr>
<tr>
<td>Completeness Bias</td>
<td>Base Rate Fallacy</td>
<td>Attenuation</td>
<td>Post-purchase Rationalization</td>
</tr>
<tr>
<td>Confirmation Bias</td>
<td><strong>Biases related to Semblance</strong></td>
<td>Hard-easy Effect</td>
<td></td>
</tr>
<tr>
<td>Information Bias</td>
<td>Similarity Bias</td>
<td>Planning Fallacy</td>
<td></td>
</tr>
<tr>
<td>Levels-of-processing Effect</td>
<td><strong>Biases related to Previous Knowledge / Experience</strong></td>
<td>Time-saving Bias</td>
<td></td>
</tr>
<tr>
<td>Reference Bias</td>
<td>Availability Bias</td>
<td>Parkinson’s Law of Triviality</td>
<td></td>
</tr>
<tr>
<td>Search Bias</td>
<td>Functional Fixedness</td>
<td>Well-travelled Road Effect</td>
<td></td>
</tr>
<tr>
<td><strong>Information Presentation Biases</strong></td>
<td>Google Effect</td>
<td><strong>Biases related to Trends</strong></td>
<td></td>
</tr>
<tr>
<td>Framing Bias</td>
<td>Law of the Instrument</td>
<td>Bandwagon Effect</td>
<td></td>
</tr>
<tr>
<td>Similarity Bias</td>
<td>Mere Exposure Effect</td>
<td>IKEA Effect</td>
<td></td>
</tr>
<tr>
<td><strong>Bias related to Trends</strong></td>
<td><strong>Biases related to Previous Knowledge / Experience</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bandwagon Effect</td>
<td>Habit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Law of the Instrument</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mere Exposure Effect</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negativity Bias</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Biases related to strategy-making</strong></td>
<td>Test Bias</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hyperbolic Discounting</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inconsistency</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cognitive Biases Catalogue
An Example: Planning Fallacy

<table>
<thead>
<tr>
<th>Planning Fallacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definitions Block</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Definition 1: The tendency to underestimate task-completion times.</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Definition 2: The planning fallacy is a phenomenon in which predictions about how much time will be needed to complete a future task display an optimism bias and underestimate the time needed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OODA Class: Decide Phase</th>
<th>OODA Subclass: Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Reasoning for classification:* Time is a crucial factor in software projects. Often, the implementation times fall short of the initial estimates. The reason being underestimation of task-completion times due to lack of understanding of the complexities involved.

Examples and impact on architecture design decisions

*Example: Choosing spring-security as the security framework:* Spring is one of the most popular choice for developing Java-based enterprise applications. To meet the security requirements, spring-security would be an automatic choice as it is part of the framework itself. It is easy to assume that configuring the application security would be as easy as developing an application in spring. However, it is not an easy solution to implement without a proper understanding. If the decision-makers assumes that the security aspect is as easy as feature development, then it leads to an optimism bias resulting in time estimate errors.

*Impact:* A common result is missing delivery deadlines. The added pressure resulting from the missed deadlines leads to implementation of sub-par solutions.

Debiasing techniques

The decision-maker must understand how to estimate time. There are many workflows for time estimation which can be used. One simple way is to add a buffer time to the initial time estimate in order to complete tasks. It is common to set the buffer time to 10% of the total estimate.

Related biases

Complexity bias, Parkinson’s Law of triviality, Time-saving bias.
Feedback and reviews were gathered by presenting results online

Target group – software architects, lead developers and product owners

“The biases and classification feel genuine. The question next is how to rectify them.“
- Vice President and Software Architect a Morgan Stanley

“We are biased because we take a look at what worked in the past and trust our future decisions based on it”
- CTO at Schoen Digital Labs

“Lots of biases and too much information. Reading all of it was intensive“
- Software Architect at Siemens

“It would be helpful if I could somehow get a notification as to which stage of decision-making I am in along with the biases I should be aware of“
- Software Architect at Siemens

“Content is good and worth reading“
- Lead developer at J.P Morgan
Research Implications

- Enforcing a structured way of decision-making from software architecture perspective to make less biased decisions
- Avoid *observe* and *orient* paralysis
- Basis for developing cognitive bias recognition engine for decision support systems
- Trainings – companies such as Siemens and IBM provide basic trainings on cognitive theories
Thank you