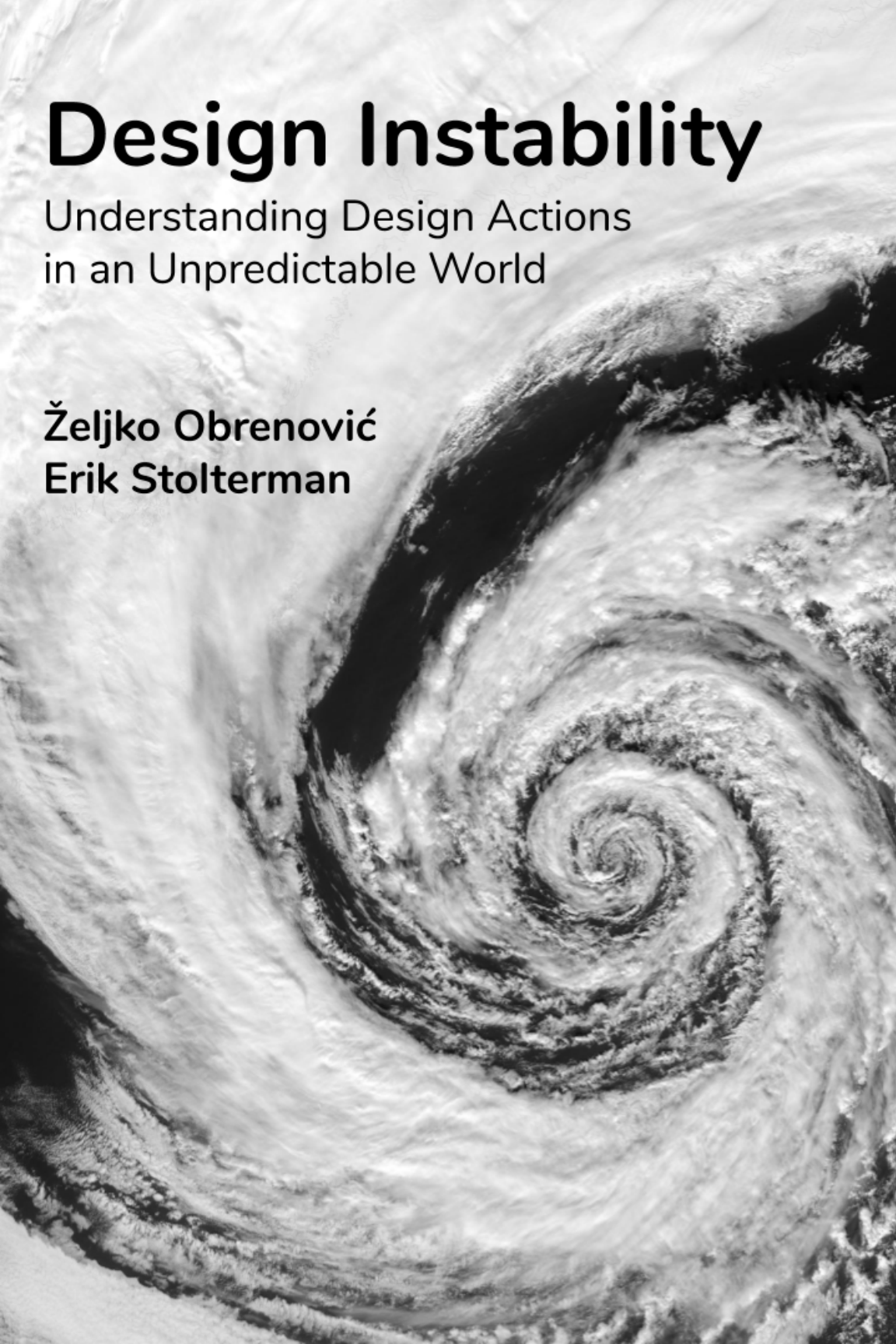


# Design Instability

Understanding Design Actions  
in an Unpredictable World

Željko Obrenović  
Erik Stolterman



# Design Instability

## Understanding Design Actions in an Unpredictable World

Željko Obrenović and Erik Stolterman

This book is for sale at <http://leanpub.com/design-instability>

This version was published on 2020-03-01



Leanpub

This is a [Leanpub](#) book. Leanpub empowers authors and publishers with the Lean Publishing process. [Lean Publishing](#) is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2016 - 2020 Željko Obrenović and Erik Stolterman

# Tweet This Book!

Please help Željko Obrenović and Erik Stolterman by spreading the word about this book on [Twitter](#)!

The suggested tweet for this book is:

Just learned about #design\_instability. @zeljko\_obren and @estolter discuss why design is instabile and complex.

The suggested hashtag for this book is #design\_instability.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

[#design\\_instability](#)

# Contents

<b>Part I: Introduction</b>	<b>1</b>
A. Analyzing Design Dynamics: A Simple Model	3
B. Why This Book?	5
C. Who Should Read This Book?	7
D. The Structure of the Book	8
E. Acknowledgments	9
F. About the Cover	10
<b>Part II: Patterns of Dynamics</b>	<b>11</b>
A. Patterns of Dynamics Between Design Situations and Design Outcomes	15
Co-Evolution of Problem-Solution	17
Puzzle Solving	23
A Solution Looking for a Problem	27
Pandora's Box	32
The Force Awakens	37
B. Patterns of Dynamics Between Design Outcomes and Resources	43
Your Design Is Your Reflection	45
You Are Reflection of Your Design	55
Design-by-Buzzword	61
C. Patterns of Dynamics Between Design Situations and Design Resources	65
Conformity	67



## CONTENTS

Commitment . . . . .	73
Cherry Picking . . . . .	78
D. Emergence of Complex Patterns of Dynamics . .	82
<b>Part III: Sources of Dynamics . . . . .</b>	<b>86</b>
A. The Moving Target Problem . . . . .	87
B. Increasing Interconnectivity . . . . .	92
C. Learning and Creativity . . . . .	96
D. Tacit Knowledge and Skills . . . . .	100
E. Social Dynamics . . . . .	103
F. Lack of Strong Principles, Theories, and Laws . .	108
<b>Discussion and Conclusions . . . . .</b>	<b>112</b>
<b>Bibliography . . . . .</b>	<b>120</b>
<b>Appendixes . . . . .</b>	<b>135</b>
A. Patterns of Dynamics: Summaries . . . . .	136
B. Patterns of Dynamics: Mottos . . . . .	138
C. Sources of Dynamics: Summaries . . . . .	139
D. Questions to Consider . . . . .	140
<b>About the Authors . . . . .</b>	<b>148</b>

# Part I: Introduction

In our professional career, both as practitioners and academics, we have seen and been involved in hundreds of design projects in diverse disciplines. We needed to understand what was going on, why some decisions have been made. We learned a lot and decided to share some of these lessons. We have taken an extra step to generalize these lessons and connect them to existing knowledge. The main result of this effort is a framework that we hope will help you to understand a broad range of design activities in our unpredictable world.

We, and many others, have experienced challenges of design, namely that designers deal with fast-changing and overwhelmingly rich reality, undetermined requirements, difficult social conditions, lack of information, limited resources, and time restrictions. Every designer is faced and challenged by these circumstances. Design activities are characterized by messy, dynamic, highly interdependent and mostly unpredictable interactions among design situations, design outcomes, and design resources<sup>1</sup>. Design situations, design outcomes and design resources are complex structures that are not independent of each other. They are continuously changing, in significant part due to forces that are beyond designers' control. Constant change and unpredictability are natural characteristics of any design activity.

In this book, we explore the dynamics of design activities

---

<sup>1</sup>We use terms *design situations* / *design outcomes* instead of *design problems* / *design solutions* because we do not view design as a simple problem-solving activity. We define a design resource in a broader sense, as tools, knowledge, processes, and organizational structures designers rely on during design activities.

and the complexity that these dynamics bring. The main contribution of this book is a structured description of design dynamics through *patterns of dynamics*. These patterns of dynamics illustrate different ways of how designers deal with ambiguity, uncertainty, and complexity when they attempt to change the world. We also believe that these patterns are universal to all design disciplines. Design disciplines may differ in situations they address, the outcomes they create and the resources they use. But the way how people deal with ambiguity, uncertainty, and complexity when designing is cross-disciplinary.

In our discussions, we address design in a broader sense, in line with the Herbert Simon's (1996) definition of design as "*courses of action aimed at changing existing situations into preferred ones.*" Due to our backgrounds and interests, we limit our overview and examples, to some degree, to discussions about complexity in software design and interaction design. In particular, we are biased towards those contributions that base their claims in observations of design practice (e.g., Brooks 1995; Glass 2006). However, we also provide connections to more general design theory (e.g., Schon 1983; Lawson 2005; Simon 1996).

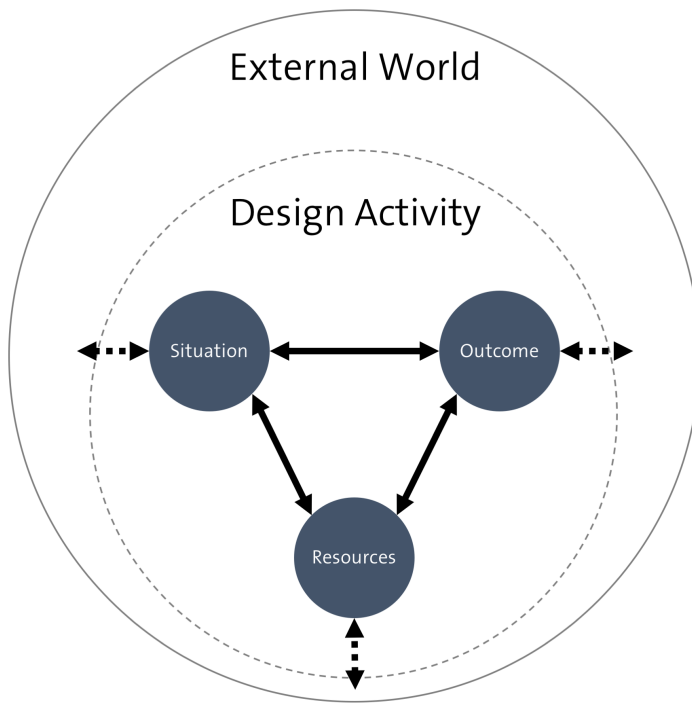
## A. Analyzing Design Dynamics: A Simple Model

Our main goal is to provide a novel view on why it is challenging to analyze, understand or explain the design process and design activities.

To structure our discussion, we adopt a simple model. This model provides a structure for analyzing dynamics of design activities as a consequence of interactions among design situations, design outcomes and design resources. Our analytic model views the dynamics of design activities as originating from designers' efforts to:

- understand and manage design situations (design problems),
- create design outcomes (design solutions), and
- understand and adequately use design resources (tools, knowledge, methodologies, procedures, and organizational structures supporting designers during a design activity).

Later in the book, we use our analytic model to structure the description of the [patterns of dynamics](#) among design situations, outcomes, and resources.



**Our analytic model for design dynamics: Design situations, design outcomes, design resources and relations among them.**

It is important to remember that the elements in our model are abstract analytical concepts. In reality, these components are not entirely separated. Nevertheless, the concepts of design situations, outcomes and resources are regularly used in design literature. We believe that they are useful concepts to structure our discussions.

## B. Why This Book?

With this book, we want to describe and provide a structured definition of the dynamics of design activities. We believe that a more formal overview can help researchers and practitioners to understand better some of the experiences designers are going through while designing. We also think that in the education of designers it is useful to picture design dynamics, and resulting complexity, as an essential part of design activities. Such design complexity has to be accepted and dealt with rather than treated as a problem to avoid. We believe that a more structured understanding of design dynamics can facilitate such education.

Lots of previous work has addressed individual elements of design activities, such as inherent complexity of design outcomes or systems (e.g., [Simon 1996](#), [Booch 2008](#), [Mens 2012](#)), or design tools. However, complex and dynamic interdependencies of elements of design activities have received relatively little attention. With this book, we focus on these interdependencies. We believe that they are primary sources of complexity characteristic for design activities. In other words, while design situations, outcomes, and resources are themselves complex structures, what makes a design activity additionally and characteristically complex are interactions among these structures. Such interactions challenge designers' ability to deal with complexity as the structures are unstable and subject to unpredictable change.

Our ambition is to provide a novel and more structured view of design dynamics. However, we do not aim at introducing a new theory about design, nor are the topics we discuss entirely new (e.g., [Stolterman 2008](#)). Instead, we

are re-examining existing work on design from a new and partially different point of view. We are also bringing together some related but previously unconnected pieces of literature. We have found similarities in the works of authors from different domains. But we have also noticed that most authors do not reference authors outside of their field of study. We believe that there is a value in connecting the contributions from different areas. By creating such connections, we hope to create a helpful overview for design practitioners and researchers interested to further study this subject.

The motivation for our work builds on the ideal sketched by Herbert Simon (1996) in *The Sciences of the Artificial*. Simon argued that all design disciplines are fundamentally similar, and that design professionals from diverse disciplines should be able to carry on mutually rewarding conversations about the content of each other's professional work (page 137). Another source of inspiration for our work comes from Fred Brooks (2010), who similarly noted that design processes in different media are strikingly alike. Reflecting on his experiences in designing in five media (computer architecture, software, houses, books, and organizations), Brooks elaborated that the mental processes, the human interactions, the iterations, the constraints, the labor - all have significant similarities (page xii). Patterns on design dynamics are our attempt to support these views and create a cross-disciplinary overview of the dynamics of design processes.

## C. Who Should Read This Book?

Our book is intended to be readable and useful for anyone who designs or is interested in design in a broader sense. That includes all types of designers and architects, but also other people that do not have such titles but are creating an intentional change in our unpredictable world ([Nelson & Stolterman 2012](#)).

We think that our book can help design researchers and students, as it makes a structured summary of and creates connections among diverse design disciplines.

We also think that patterns of design activities do have a practical value. The patterns can be used both as an analytic tool and as a management tool. As an analytic tool, patterns provide a structure and vocabulary to describe and communicate what is going on in design activities. Having a common structure and vocabulary also enables comparison and correlation of different design activities. As a management tool, our patterns offer hints on when, why and how to stimulate desired (or prevent undesired) dynamics in design activities.



## **D. The Structure of the Book**

The rest of this book is organized as follows.

In Part II, we describe patterns of dynamics in detail. We argue that the dynamics of design activities are emerging from elaborate and not entirely known interactions among the elements of design activities. With examples, we illustrate that dynamics between design situations, design outcomes and design resources are diverse and bidirectional.

In Part III, we discuss sources of interactions among the elements of design activities. We address the moving target problem, learning, creativity, tacit knowledge, social dynamics, as well as the lack of robust theories and underlying principles. Here we also discuss how are these aspects related to the patterns of dynamics.

We conclude the book with a discussion.

## **E. Acknowledgments**

We would like to thank Mathias Funk and students of the TU/e master course “Beautiful Data: Designing InfoProducts” for providing useful feedback during the lecture based on the early draft of this book. Mathias Funk also proposed the name for the “Pandora’s Box” pattern.

## F. About the Cover

The cover shows an [extratropical cyclone](https://en.wikipedia.org/wiki/Cyclone)<sup>2</sup> near Iceland on September 4, 2003. We selected this image as it symbolizes the central concept of this book - the dynamics of design activities. A cyclone is a complex, dynamic, and partially unpredictable phenomenon. Still, as with design dynamics, it is not an entirely chaotic phenomenon, as some patterns are visible.

---

<sup>2</sup><https://en.wikipedia.org/wiki/Cyclone>

# Part II: Patterns of Dynamics

*“The real power of patterns is not to hand us exotic solutions, but to give us a way to remember the simple, ordinary, basic solutions that we know but forget”. [Rising 2010](#)*

In this part of our notes, we discuss our central thesis. Namely, that design activities are characterized by inherent instability - complex and dynamic relations among design situations, design outcomes and design resources. We believe that these active ties are primary sources of complexity characteristic for design activities. As mentioned in the introduction, while design situations, outcomes, and resources are themselves complex structures, what makes a design activity additionally and characteristically complex are relations among these structures. Dynamics of these relations challenge designers' ability to deal with complexity as any of the structures are subject to change, in a difficult to predict ways.

We want to achieve two goals. Firstly, we want to show that, in design, it makes no sense to treat design situations, design outcomes and design resources as stable elements and in isolation of each other. Furthermore, we want to challenge the conventional idea that the design process moves from a problem to a solution linearly and causally. Instead, the dynamics of the network connecting design situations, design outcomes, and design resources are highly diverse. Of course, there already exists a significant diversity in how design literature describes these dynamics. However, we still find it

necessary to contribute to this ongoing debate by providing another perspective and hopefully useful, more structured model.

Secondly, we want to show that regardless of the seeming arbitrariness of interactions between design situations, outcomes, and resources, it is still possible to constructively portray, investigate and discuss such dynamics. In other words, we want to argue that in many cases these dynamics show signs of “organized complexity,” where we may observe some patterns. As a mean to achieve this goal, we introduce a concept of *patterns of dynamics* between elements of design activities. A pattern is a recurring arrangement of interactions among elements of design activities. By studying contemporary design literature, we have found several common patterns of these dynamics. It is important to note that these patterns are primarily conceptual tools. The purpose of patterns is to support analytic and reflective thinking for practical purposes and not to serve as strict scientific definitions of existing elements.

The patterns of dynamics illustrate how people deal with ambiguity, uncertainty, and complexity when they attempt to change the world. As such, these patterns are universal to all design disciplines. Design disciplines may differ in which situations they address, which outcomes they create and which resources they use. But the way how people deal with ambiguity, uncertainty, and complexity is cross-disciplinary. Consequently, we have found examples of all patterns of dynamics in all of the design disciplines we analyzed.

With eleven patterns, we provide an initial categorization and description of dynamics that we have identified between design situations, outcomes, and resources. We are aware that it is not possible to capture all patterns that may occur in

design processes. By examining findings from contemporary seminal design research, we abstracted some core patterns into a format that can support discussions about design complexity.

None of the described patterns is universally good or bad. Each pattern may be desirable in one situation, and undesirable in another. Patterns may be a consequence of a conscious effort to create particular design dynamics, or they may occur accidentally.

For convenience and clarity, we describe each pattern of dynamics with the same format:

- a brief descriptive name,
- a “motto,”
- optional alternative names,
- a summary,
- a short list of topics related to the pattern,
- a brief description of when the pattern is desirable and undesirable, and
- a more detailed discussion based on examples from the design literature.

We structured our review of patterns, based on [our model](#), around three “axis” of relations between “elements” of design activities:

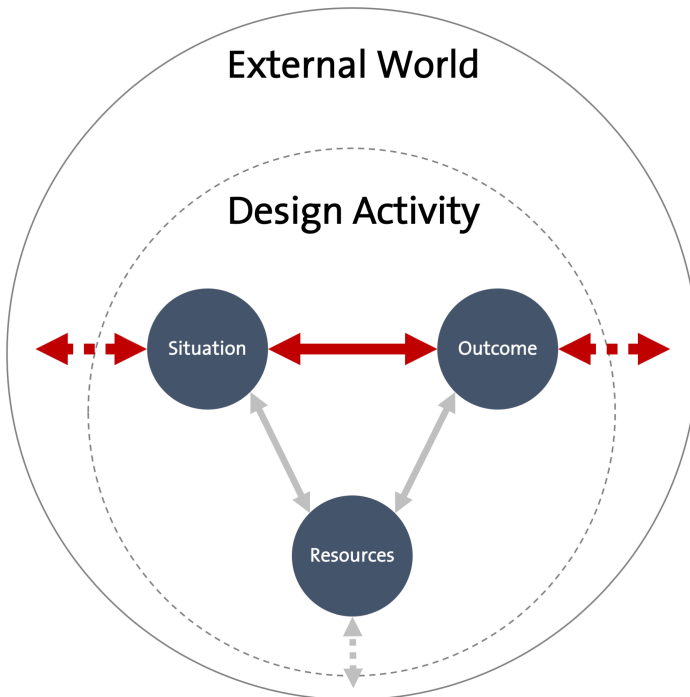
- dynamics between design situations and outcomes,
- dynamics between design outcomes and resources,
- dynamics between design situations and resources.

Each pattern may be viewed as the description of dynamics that shape design decisions during design activities. In any design activity, designers make many choices, trying to answer the following questions (Obrenovic 2008):

- How will the design process advance?
- What needs and opportunities will the design address?
- What form will the resulting outcome take?

Our patterns describe dynamics that influence decisions that designers make when trying to answer these questions. When we look at the result of some design activity, with hindsight, we usually can describe what the design outcome is, how it has changed the original design situation, and what design resources have been used. However, a route to the design result may be very complicated, and with very diverse starting points. As we will illustrate with our patterns, sometimes, we may start with a problematic situation that we want to improve. Sometimes we may not have a problem to solve but a technology that people suddenly start to use. And sometimes the sheer availability of some design resource may be the primary driver of change. And frequently, multiple of these patterns of dynamics will co-occur.

## A. Patterns of Dynamics Between Design Situations and Design Outcomes



The dynamics between of a design situation and a design outcome seem to be the most extensive in any design activity. In the design literature, we have found several common and often interconnected views on dynamics of this relation, which we describe in the following five patterns:

- Co-Evolution of Problem-Solution,



- Puzzle Solving,
- A Solution Looking for a Problem,
- Pandora's Box, and
- The Force Awakens.

## Co-Evolution of Problem-Solution

**Motto:** *“If you want to change something, you need to understand it, if you want to understand something you need to change it.” (Gravemeijer and Cobb 2006)*

**Summary:** *A design situation and designer’s understanding of the design situation (problem definition) changes and evolves in parallel and under the influence of design outcome and designer’s understating of potential design outcomes (design solutions).*

**Alternative names:** *Problem–Solution Flux.*

**Topics:** *problems solving, problem setting, prototyping, iterative development, agile software development, reinventing the wheel*

**Desirable:** *In most complex real-world situations and new domains, as this pattern facilitates learning, stimulates communication among stakeholders, and minimizes the risk of failure.*

**Undesirable:** *In well-known domains, where it may lead to the “reinventing-the-wheel” and “[not-invented-here](https://en.wikipedia.org/wiki/Not_invented_here)<sup>3</sup>” anti-patterns.*

---

Design authors generally agree that design situations are never clearly defined problems. Instead, the model commonly used is that of a co-evolution (Dorst & Cross 2001). In this model designers’ understanding of what the actual problem is evolving together with their attempts to create a solution.

---

<sup>3</sup>[https://en.wikipedia.org/wiki/Not\\_invented\\_here](https://en.wikipedia.org/wiki/Not_invented_here)

Donald Schön (1983) was among first to argue that design is not a straightforward problem-solving activity. He elaborated that real-world problems do not present themselves to practitioners as givens (page 40). Instead, problem understanding must be constructed from the materials of problematic situations, which are puzzling, troubling, and uncertain<sup>4</sup>. Practitioners must make sense of unclear descriptions of situations that initially makes no sense. Schön emphasized that in design practice, *problem setting* is as crucial as *problem solving*.

Design authors also emphasize that a design situation cannot be fully understood unless some attempts have been made to change or improve the situation. Bryan Lawson, for example, claimed that many components of a design problem could not be expected to emerge until some attempt has been made at generating solutions (Lawson 2005: p 120). He argued that design situations are often full of uncertainties when it comes to the objectives and their priorities. These goals and priorities are likely to change during the design process as the solution implications begin to emerge. Lawson concluded that we should not expect a comprehensive and static formulation of design problems. Instead, we should see design problems as in dynamic tension with design solutions.

Herbert Simon (1996) stated that a goal of design might be to understand the problem and to generate new goals. He elaborated that the idea of final goals and a static problem definition is inconsistent with our limited ability to foretell or determine the future (page 162). Simon claimed that by designing without a fixation on final goals, and by adopting

---

<sup>4</sup>Rittel and Webber (1973), for instance, claimed that real-world problems are “wicked” problems, where a problem cannot be defined until a solution has been found. Such problems are also sometimes called ill-structured (Reitman 1965; Simon 1973) or complex (Funke 1991). For readers interested in discussion on wicked problems in design, we recommend Buchanan 1992, Coyne’s (2005) and Farrell & Hooker’s (2013).

the stance to welcome new, emergent and originally unintended goals, design becomes a powerful tool for discovering new previously unforeseen goals (Chua 2015). As an example, Simon used an extensive renewal program in the city of Pittsburgh, where a principal goal was rebuilding the center of the town, the so-called Golden Triangle. Simon noted that the main consequence of the initial step of redevelopment was to demonstrate the possibility of creating an attractive and functional central city on this site. This demonstration was followed by subsequent construction activities that have changed the whole face of the city and the attitudes of its inhabitants. Each step of implementation created a new situation. The new situation provided a starting point for fresh design activity (Simon 1996, page 162).

Fred Brooks noted that it is impossible for a client to specify completely, precisely, and correctly the exact requirements of a software product before trying some versions of the product (Brooks 1995). Brooks claimed that the most challenging part of software design is deciding what to build. He went further to argue that a core service offered by a designer is not only providing a solution but helping clients to discover what they want (Brooks 2010, page 23). Empirical studies of software projects confirm these observations, as inadequately defined system requirements are among leading causes of software project failures (Charette 2005).

Classical design practices, such as sketching and prototyping, are mechanisms that stimulate co-evolution of solutions and problems (see Buxton 2007). Schön and Wiggins (1992), using the example of an architect, described that architect use sketching to engage in the “seeing-moving-seeing” sequence. This sequence consists of creating a drawing to represent an initial idea, observing a drawing, discovering “certain

unintended consequences,” and reacting to this discovery by further transforming the drawing<sup>5</sup> (page 139). With sketching, designers can, in relatively short time and with low costs, significantly improve their understanding of a design problem and possible design solutions (Obrenovic 2013).

In modern software development, a co-evolution is the primary pattern of dynamics behind the [agile software development](#)<sup>6</sup> movement, nowadays a mainstream software development methodology. Agile software development promotes iterative and incremental development, using continuous feedback to refine and deliver a software system. One of the central values behind agile software development is *responding to change over following a plan*. This approach makes explicit the expectation that requirements (problem definition) cannot be fully known in advance, and will change during the project.

Because it stimulates learning and exploration, the *co-evolution of problem-solution* pattern is particularly useful in new domains, for which there are little known and documented experiences. The co-evolution of problem-solution, however, may be an anti-pattern when applied in well-known areas, where it may lead to the “reinventing-the-wheel” and “not-invented-here” anti-patterns (e.g., see [Obrenovic 2017](#)). In well-known domains, there usually are lots of documented experiences and best practices, and there are often standard solutions that can be reused. For example, for most organizations, it is costly and time-consuming to implement their custom solutions for tasks such as human resources management or communication (email, messaging). Typically, it is much

---

<sup>5</sup>Goldschmidt (1991) further articulated this process, making a distinction between *seeing-as* and *seeing-that* moves in a “dialectics of sketching.”

<sup>6</sup><https://www.agilealliance.org/agile101/>

easier and more cost-effective to adapt the organizational process to use standard off-the-shelf software solutions for these tasks.

The *co-evolution of problem-solution* pattern may be connected to the *puzzle solving* pattern. There it may occur during co-evolution of problem-solution as a way to find answers to specific well-scoped questions. The *co-evolution of problem-solution* pattern may also be related to the *design-by-buzzword* pattern. There it may occur as a way to develop a better understanding of useful applications for new design resources.



### Questions to Consider

- Do you continuously investigate, experiment, and iteratively refine your designs?
- How do you resolve uncertainties about design objectives and their priorities?
- Do you use sketching and prototyping extensively?
- How do you estimate and plan projects with the co-evolution pattern?
- How do you decide to design something new versus buying it off-the-shelf?
- Are you reinventing the wheel by co-evolving solutions for well-defined problems with off-the-shelf solutions?
- Are you guilty of the not-invented-here syndrome, routinely building yourself things for any issue you face?
- Have you ever used the off-the-shelf solution that was not appropriate for your domain? What issues have you faced?



**Pittsburgh's Golden Triangle.** In the 1950s, an extensive renewal program began in the city of Pittsburgh, with a principal goal of rebuild the center of the town, the so-called Golden Triangle. Credit: U.S. National Archives and Records Administration / Wikimedia Commons.

## Puzzle Solving

**Motto:** *“A problem well put is half solved.” (John Dewey, The Pattern of Inquiry)*

**Summary:** *A design situation is viewed as a clearly defined and static problem (i.e., a problem that can be clearly stated and where it is known what form the solution should have). A design outcome is seen as a solution for this problem. Design is viewed as a problem-solving activity.*

**Topics:** *problem-solving, the puzzle trap, education, well-scoped design sub-tasks, test-driven development*

**Desirable:** *In education, as well as in a limited number of practical cases for well-scoped design sub-tasks.*

**Undesirable:** *In most real-world situations, because few design problems are fully understood before the start of design activity.*

---

The *puzzle solving* patterns views design as a problem-solving activity, an activity that leads to a solution for a precisely defined existing problem.

This pattern is of limited use in design practice because very few of design problems are fully understood before a design activity starts (see [Section Co-Evolution of Problem-Solution](#)). Lawson even called such view on design “the puzzle trap” (Lawson 2005, page 221). Norman (2010) similarly explained that interactions design is sometimes naively viewed as a problem-solving activity. In such naïve view, interaction designers are capturing and understanding the hidden unmet



needs of potential users/customers. The intention of designers is then to design an outcome that fulfills those requirements. Norman noted that this view, however, does not reflect the everyday practice of interaction designers.

The *puzzle solving* pattern has its value in a limited number of practical cases. A designer can approach some of the design sub-tasks as problem-solving activities. Design of computing algorithms, for example, can be viewed as an occurrence of the *puzzle solving* pattern. Here, the goal is to find an efficient algorithmic solution for a precisely defined computing problem<sup>7</sup>. In software design, *test-driven development*<sup>8</sup> (TDD) promotes an approach to designing and programming of software systems through series of mini problem-solving activities. In a TDD approach, a developer should write every line of new code in response to a test the developer writes just before coding (Melnik & Jeffries 2007). Creating such tests may be seen as an attempt to define a problem (problem setting). And writing new code may be seen as an attempt to find and implement a solution that is good enough to pass the test.

Teachers can also use the *puzzle solving* pattern in education, where problems can be artificially clearly defined so that students can focus on developing a particular design technique. For example, it is a common practice to teach computer programming through exercises where students need to de-

---

<sup>7</sup>Design of algorithms, however, is only a small part of the software design, and designing whole systems requires a different approach. Butler Lampson nicely described this issue in his article on system design Lampson (1984). Lampson argued that designing a computer system is very different from developing an algorithm because the requirements are less precisely defined, more complex, and more subject to change. The measure of success is much less clear. And the designer usually finds himself in “a sea of possibilities,” where it is not clear how one choice will limit his freedom to take other decisions or affect the size and performance of the entire system.

<sup>8</sup>[https://en.wikipedia.org/wiki/Test-driven\\_development](https://en.wikipedia.org/wiki/Test-driven_development)

sign an algorithm for an explicitly defined problem (such as calculating the number of days between two dates). Similarly, Bill Buxton (2007) promoted an approach to interaction design education based on copying the classics, drawing a parallel with the traditional art education. In traditional art education, drawing antiquities (a clearly defined problem), is often used to lay the foundation for building up the student's skills. In such exercises, students are not overwhelmed by trying to deal with issues such as thinking about an appropriate subject or composition (problem setting). Instead, they can focus on learning how to master the pencil, a prerequisite to high art.

The *puzzle solving* pattern may be present within the *co-evolution of problem-solution* pattern, as a way to find answers to specific well-scoped questions.



## Questions to Consider

- In your design activities, do you have lots of well-defined design sub-tasks, such as algorithm implementation?
- Do you use a problem-solving approach for skills development in education and training?
- Have you ever attempted to solve a problem only to realize that the problem is not well defined?
- Have you ever successfully “solved” the wrong problem?
- Have you ever created an unnecessarily complicated solution because you had not well understood the problem?



A jigsaw puzzle. Solving problems, such as puzzles, is often seen as an anti-pattern in design but may have the value in education. Credit: MeHe / pixabay.

## A Solution Looking for a Problem

**Motto:** *“When humans possess a tool, they excel at finding new uses for it.” (Nye 2006)*

**Summary:** *Understanding of a design outcome and its possibilities leads to innovative usages of the design outcome in situations and for problems that were not initially envisioned.*

**Topics:** *novelty, creativity, curiosity, unordinary usages of ordinary things, early adopters, bitcoin, and blockchain*

**Desirable:** *As an innovation strategy, when an organization wants to be an early adopter of emerging technology and where the organization can afford associated risks.*

**Undesirable:** *In most other situations as it requires significant investments without tangible benefits.*

---

In this section we address examples of design activities where there was a significant investment in the development of a design “solution,” but without clear idea (or with the wrong idea) about the problem that such “solution” should solve.

The *solution looking for a problem* pattern may lead to successful results because people often find new uses for existing things. David Nye (2006) noted that when humans possess a tool, they excel at finding new applications for it. He argued that necessity is often not the mother of invention and that in many cases the opposite was the case (page 2). The tool usually exists before the problem to be solved. Nye concluded that latent in every tool is unforeseen transformation. Fred Brooks similarly noted that as a software product is found to

be useful, people try it in new cases at the edge of, or beyond, the original domain (Brooks 1995, page 12).

Norman explored this issue as well, claiming that many major innovations came from technologists who had little understanding of the problem domain (Norman 2010). He stated that this is true for almost all of the outstanding inventions that have changed society: the airplane, the automobile, the telephone, the radio, the television, the computer, the personal computer, the internet, SMS text messaging, and the cellphone. Norman claimed that with these inventions the technology came first, the products second, and then the needs slowly appeared. New applications are first considered to be luxuries, after a while they become “needs,” and finally, they may end up being seen as essential. In his prior work, Norman (2008) similarly explored how unordinary usages of ordinary products may lead to new requirements and new products. Norman provided an example of Nokia’s designers. They added a penlight to some of their phones after visiting China where they noticed that people were using displays of their mobile phones as a source of light.

Nowadays, bitcoin and blockchain<sup>9</sup> are examples of technologies that [some see as solutions looking for a problem](#)<sup>10</sup>. Blockchain opens a range of new possibilities for businesses in which value needs to be directly transferred between participants over the Internet without intermediaries or centralized points of control (Aste et al. 2017). While proponents argue that the blockchain technology has enormous potential, practical applications of blockchain were limited. The blockchain

---

<sup>9</sup>Blockchain is a technology that uses community validation to keep synchronized the content of ledgers replicated across multiple users. While blockchain derives its origins from techniques introduced decades ago, it has gained popularity with Bitcoin.

<sup>10</sup><https://qz.com/844507/bitcoin-and-blockchain-seem-more-and-more-like-solutions-looking-for-a-problem/>

technology is still looking for more practical problems to solve.

A *solution looking for a problem* requires typically significant investments without tangible benefits. This approach is usually an anti-pattern, especially for smaller organizations that invested most of their resources in the development of such solutions. The [analysis of Forbes<sup>11</sup>](https://www.forbes.com/sites/groupthink/2016/03/02/top-20-reasons-why-startups-fail-infographic/), for instance, discovered that 42% of unsuccessful startups had failed due to the lack of marketing need for “solutions” they offered.

While the *solution looking for a problem* pattern introduces significant risks, it may be a successful innovation strategy. Being an early adopter of some technology may be a crucial competitive advantage. Waiting to invest in the technology after its value has been demonstrated on the market may be too late. Consequently, organizations are investing in the development of innovative solutions, not because of immediate marketing needs, but to be in a good starting position if such marketing needs appear.

The *solution looking for a problem* pattern is related to the *co-evolution of problem-solution* pattern. Once a new usage of a design “solution” emerges in a new situation, this new situation may start to influence further evolution of the original “solution.”

---

<sup>11</sup><https://www.forbes.com/sites/groupthink/2016/03/02/top-20-reasons-why-startups-fail-infographic/>



## Questions to Consider

- Have you ever designed something just for fun?
- Do you reserve time for experimentation and exploration of new technologies and methods?
- Do you experiment with new technologies and practices even if you do not have a business case nor a clear idea about its usefulness?
- Do you experiment with new technologies enough to get an opinion about them?
- Are you always playing it safe, using only well-proven technologies and methods?



A DynaTAC 8000X (first commercially available mobile phone from 1984). Martin Cooper of Motorola made the first publicized handheld mobile phone call on a prototype DynaTAC model on April 3, 1973. Credit: Hinman, Rachel. 2012. *The Mobile Frontier*. New York: Rosenfeld Media. / Flickr / CC-BY-2.0



## Pandora's Box<sup>12</sup>

**Motto:** *“Everything we design has the potential not only to solve problems but also to create new ones.” (Lawson 2005)*

**Summary:** *The consequences of a design outcome create new situations that are perceived as problematic and may ask for another design to improve it. A design outcome may become more known for the problems it caused than for the solution it provided.*

**Alternative name:** *Problematic Solution, Collateral Damage.*

**Topics:** *unexpected and undesired effects, innovative domains, expectation management, the year 2000 problem*

**Desirable:** *In a limited number of real-world situations, when a goal is to stimulate further development of some solution without immediately benefiting from it.*

**Undesirable:** *In most other real-world situations.*

---

Any design outcome may, in good faith, introduce unexpected and undesired effects. Design literature provides numerous examples of designers having to deal with issues of unintended consequences of their designs (e.g. Janlert and Stolterman 2008). Lawson nicely illustrated this issue on an example of a motorcar:

*“Design solutions are not panaceas and most usually have some undesirable effects as well as the intended good effects. The modern motor car is a wonderfully sophisticated*

---

<sup>12</sup>Pandora's box is an artifact in Greek mythology, taken from the myth of Pandora in Hesiod's Works and Days. In modern times an idiom has grown from it meaning “Any source of great and unexpected troubles”, or alternatively “A present which seems valuable but which in reality is a curse”. [https://en.wikipedia.org/wiki/Pandora's\\_box](https://en.wikipedia.org/wiki/Pandora's_box)

*design solution to the problem of personal transportation in a world which requires people to be very mobile over short and medium distances on an unpredictable basis. However, when that solution is applied to the whole population and is used by them even for the predictable journeys, we find ourselves designing roads, which tear apart our cities and rural areas. The pollution which results have become a problem in its own right, but even the car is now beginning not to work well as it sits in traffic jams! This is a very dramatic illustration of the basic principle that everything we design has the potential not only to solve problems but also to create new ones!"* (Lawson 2005: p 122)



Traffic Jam in Maputo. A car is an excellent design solution to the problem of personal transportation, but when applied to the whole population it creates other problems, such as traffic jams. Credit: A Verdade / Wikimedia Commons / CC-BY 2.0.

In software design, one of the most prominent examples of problems caused by previous design solutions was the “the year 2000 (Y2K) problem” (e.g. [Neumann and McCullagh 1999](#); [Berghel 1999](#); [Pescio 1997](#)). A shortcut taken by early software designers cause this problem. To conserve memory space, early software designers used a solution that recorded the year using the last two digits rather than four. Computers using this system would then recognize the year 2000 as the year 1900 instead, potentially causing severe problems in many sectors. It was estimated that costs of fixing the Y2K problem exceeded \$3 trillion worldwide ([Brown et al. 2000](#)).

Spam is another example of problems created by the success

of messaging and social media technologies, such as email, SMS, Twitter, and Facebook. Spam misuse the ease and low price of sending messages (a solution) to distribute huge volumes of messages that propagate malware, disseminate phishing exploits, and advertise illegal products. Spam messages generate significant costs for users and network operators (Fonseca et al. 2016). Furthermore, spam has required significant investments in spam detection and prevention tools, which are now core tools of any messaging platform.

The pandora's box pattern is a consequence of designers' inability to predict all effects of design outcomes. Unexpected and undesired effects of design outcomes are more likely to appear in more innovative domains. Sometimes this is a consequence of attempts to speed up a design process, leaving insufficient time for exploration and learning (the *co-evolution of problem-solution* pattern). For instance, construction of the Sydney Opera House, due political pressures, started too early, before final designs were completed. This forced early start led to significant later problems. For instance, constructors discovered that the podium columns were not strong enough to support the roof structure, and had to be re-built (Murray 2004). Due to a number unexpected and unintended effects, the Sydney Opera House project was completed ten years late and with the 1,357% over-budget.

If people will perceive a design outcome as problematic is also a function of their expectations. Marketing and other expectation management techniques may have a significant impact on how people will accept the design outcome. Don Norman, in his [talk at Main Event in 2004](#)<sup>13</sup>, said that expectations are connected to hope, fear, satisfaction, and anger. Norman elaborated that delivering on positive expectations

---

<sup>13</sup><https://mprove.de/script/04/nng/expectationdesign.html>

leads to people experiencing pleasure. But failure to deliver on positive expectations may result in negative emotions and perception of a product as problematic.

The *pandora's box* pattern is related to the *force awakens* pattern, as both are a consequence of unexpected and undesirable effects of design decisions.



### Questions to Consider

- How do you manage the risk of encountering unforeseen issues?
- How do you react when unexpected issues occur? Do you have a process for it?
- Have you ever designed something that has caused significant unforeseen problems?
- Have you ever regretted starting your design activity because of these unforeseen problems?

## The Force Awakens

**Motto:** *“How can something seem so plausible at the time and so idiotic in retrospect?” (Bill Watterson, Calvin, and Hobbes)*

**Summary:** *Designer’s efforts to address some situation triggers reactions that radically change the original design situation. Often such reactions make the intended design outcome or a contribution of a designer obsolete or irrelevant. Such responses would typically not occur without the design activity.*

**Alternative names:** *Boomerang, Ricochet, Own Goal, Shoot Oneself in the Foot.*

**Topics:** *social and political issues, mass market, competition, IBM PC, vaporware, one laptop per child (OLPC) project*

**Desirable:** *In a limited number of real-world situations, when a goal is to stimulate further development of some solution without benefiting from it.*

**Undesirable:** *In most other real-world situations.*

---

The force awakens pattern described dynamics of design activities in which designers’ efforts to create a design outcome lead to radical changes of the original design situation. Often such change can make intended design outcomes unsuccessful, obsolete or irrelevant. This type of patterns of dynamics can be very elaborate if the design enters the mass market or can have a significant influence on the society.

An example of the *force awakens* pattern is the “one laptop per child” (OLPC) project ([Kraemer et al. 2009](#)). The OLPC project created a novel technology, the XO laptop, developed with

close attention to the needs of students in poor rural areas. The plan failed to reach many of its original goals, in huge part due to the aggressive reaction of the PC industry. The PC industry perceived the OLPC project as a threat of a \$100 laptop distributed in emerging markets (Kraemer et al. 2009). Soon after the OLPC project started, the PC industry began to introduce their educational computers with low prices. This reaction of the PC industry would not occur without the OLPC project. And OLPC project probably would not start or would have different goals, if there were available educational PCs with low price.

A similar story is the development of the IBM PC (Goth 2011). IBM underestimated the influence that PC would have for their future. On the one hand, the introduction of the PC changed the market and led to less demand for IBM main-frame computers. IBM led the PC market in the beginning. But as the IBM PC market grew, IBM's influence diminished. In November 1985 PC Magazine stated *"Now that it has created the [PC] market, the market doesn't necessarily need IBM for the machines"*. The IBM management was also not able to predict the impact of their design, as reported by Greg Goth:

*"And management said, 'Okay, fine. It won't hurt anything. It's not going to mess up our business. You're not trying to replace us. You have 10 or 12 people. When you find it's really not much of anything, you'll come back, and we'll do real computing.' That was the attitude. They weren't afraid. They thought it would be, maybe, interesting. It might actually be a follow-on to Displaywriter and Datamaster. We'd sell maybe 100,000 or 200,000 units, and then we'd keep doing computing as we had always done it. So that's why they left us alone. We were off the radar."* (Goth 2011, page 26)

The *force awakens* pattern may also occur even before any observable design activity has started. A decision to work on some design situation may cause some reactions that may change that situation. Lawson noted that once a design situation has been identified, it is no longer possible to avoid deciding on a design outcome (Lawson 2005, page 115). He elaborated that in many real-life design situations it is not possible to take no action because the very process of avoiding or delaying a decision has an effect. For example, if a new road is planned, but the route remains under debate for a lengthy period, the property in the region of the various paths will likely change the value. Lawson concluded that procrastination as a strategy in design is deeply flawed.

In software design a similar effect we may find in “*vaporware*”<sup>14</sup>. Vaporware is a practice of announcing a product that does not exist to gain a competitive advantage and keep customers from switching to competing products (e.g. Heiko 2004; Richard 1995). Announcements of the competitors may seriously harm designers’ intention to address a particular situation. Such, often very simple announcements, may radically change the market and clients’ expectations. Vaporware is another example of the *force awakens* pattern, as here a design activity triggers forces that are significantly changing the design situation and are undermining the design effort. Again, these forces would not be “awakened” without the design activity.

The *force awakens* pattern is connected to the *pandora’s box* pattern, as both of these patterns describe dynamics that are a consequence of unexpected and undesirable effects of design decisions. The *force awakens pattern* may lead to the *solution looking for a problem* pattern. A failure to create an outcome

---

<sup>14</sup><https://en.wikipedia.org/wiki/Vaporware>



in one design situation may lead to usage of a (part of) design outcome in another situation.



## Questions to Consider

- How do you analyze the market and respond to changes in it?
- How do you anticipate the reaction of competition to your design efforts?
- Have your design efforts ever caused other players (e.g., competition) to react strongly against your activities?
- Have you ever responded strongly to the design efforts of others?
- Have you ever experienced the effect vaporware, a practice of announcing a product that does not exist to gain a competitive advantage and keep customers from switching to competing products?

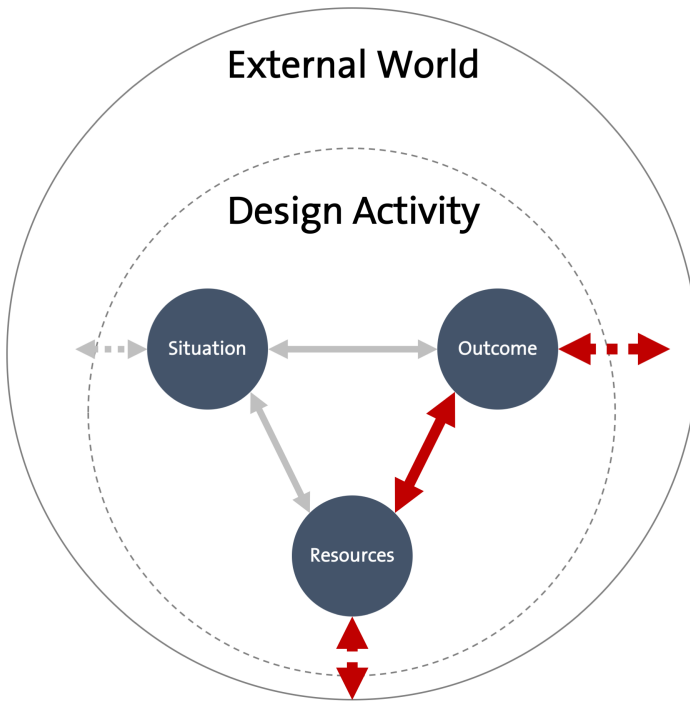


**IBM Personal Computer, 1981.** IBM introduction of PC changed the market and led to less demand for IBM mainframe computers. Credit: Wikimedia Commons.



The U.S. Justice Department accused IBM of intentionally announcing its System/360 Model 91 computer (pictured) three years early to hurt sales of its competitor's computer. Credit: Wikimedia Commons.

## B. Patterns of Dynamics Between Design Outcomes and Resources



Similar to the dynamics between design situations and outcomes, the dynamics between design outcomes and design resources are in design literature described as sophisticated and bidirectional. The way how designers set the design process when it comes to steps, phases, and activities has direct consequences on what they can design. On the other hand, design outcome may also influence design resources. We discuss the following patterns of dynamics:

- Your Design Is Your Reflection,
- You Are a Reflection of Your Design, and
- Design-by-Buzzword.

## Your Design Is Your Reflection

**Motto:** *“Every contact leaves a trace.” (Edmond Locard)*

**Summary:** *Design resources, as well as the way how working with these resources is organized, are leaving a characteristic signature on a design outcome. It is often possible to guess, from the design outcome, which design resources designers used, and how the working with the resources was organized.*

**Alternative names:** *Resource Signature, Design Style.*

**Topics:** *constraints, consistency, templates, frameworks, convention-over-configuration, Conway’s Law*

**Desirable:** *In situations where consistency among design outcomes is essential.*

**Undesirable:** *In complex real-world situations and in new domains where it may lead to unnecessary limited or complex solutions.*

---

The characteristics of design resources directly influence the design outcome’s form and possibilities. This influence can happen at all levels, from simple design tools to complex organizational structures. Selection of design materials, for instance, leaves a clear signature on a design outcome. But all design resources, including tools, methods and organizational structures supporting a design activity may leave a typical signature on a design outcome.

The following table illustrates levels at which this pattern may occur.

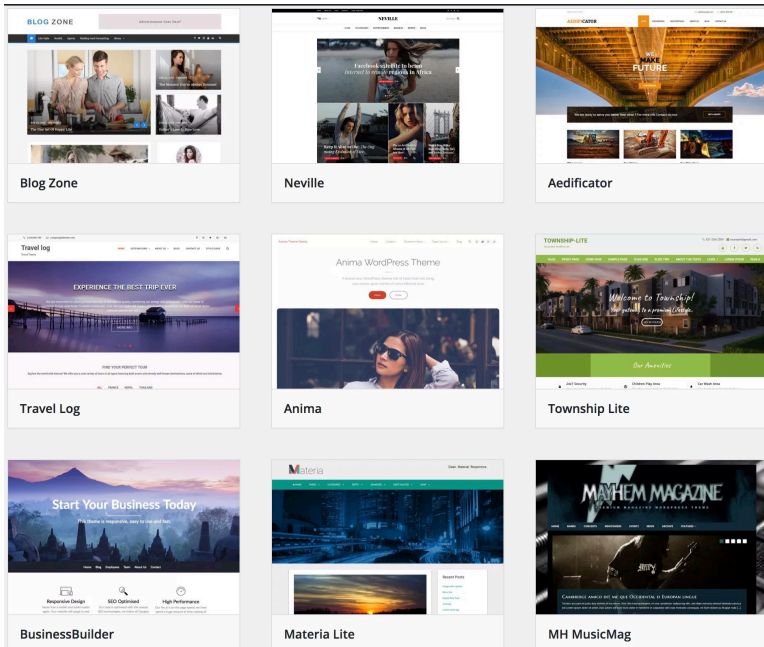
Level	Examples
Technology and tools	Templates, material characteristics or default tool settings drive the shape of outcome
Individual designers	Designer’s style, skills and preferences drive the shape of outcome
Team and organization	Team organization and communication structures drive the shape of outcome (Conway’s law)

At the tool level, the *your design is your reflection* pattern is a consequence of possibilities and limitations of used design tools. Design tools empower designers but at the same time, they may constraint designers’ actions. These restrictions often leave a typical signature on a design outcome. Mary Collins, in her post “[Web Design Trends: Why Do All Websites Look the Same?](#)”<sup>15</sup> noted that modern websites style-wise look very similar: “*large full-width background image or video in the header, overlaid text, followed by a block of short text, and then the obligatory icon columns.*” Collins argued that this uniformity has its origins in popularity of UI frameworks, in particular [Bootstrap](#)<sup>16</sup>, as well as the availability of pre-designed website themes, such as those in [WordPress](#)<sup>17</sup>.

<sup>15</sup><https://www.friday.ie/journal/why-do-all-websites-look-the-same/>

<sup>16</sup><http://getbootstrap.com/>

<sup>17</sup><https://wordpress.com/>



The screenshot of several popular WordPress themes with similar styles.

In software design, developers need to use software frameworks and libraries. These frameworks often have recommended ways of working and their conventions. Some software frameworks follow the [convention-over-configuration](https://en.wikipedia.org/wiki/Convention_over_configuration)<sup>18</sup> approach, making it easier to do things that mimic particular conventions. Sometimes, such frameworks are called “[opinionated](https://stackoverflow.com/questions/802050/what-is-opinionated-software)” frameworks<sup>19</sup> as they embed and promote “opinions” about how things should be done. Such “opinionated” frameworks encourage designers into doing things their way. By design, such frameworks are leaving a strong signature on a design outcome. For example, [Maven](https://en.wikipedia.org/wiki/Apache_Maven)<sup>20</sup>, one of the most popular software project management tools, strongly

<sup>18</sup>[https://en.wikipedia.org/wiki/Convention\\_over\\_configuration](https://en.wikipedia.org/wiki/Convention_over_configuration)

<sup>19</sup><https://stackoverflow.com/questions/802050/what-is-opinionated-software>

<sup>20</sup>[https://en.wikipedia.org/wiki/Apache\\_Maven](https://en.wikipedia.org/wiki/Apache_Maven)



encourages applications to follow standard Maven directory structure and file name conventions. In practice, that means that virtually all Maven projects will have a very similar source code organization. While this may be limiting in some cases, the significant advantage is that anyone who worked on one Maven project will be able to work in another Maven project more efficiently.

One advantage of the design signature pattern is that it facilitates consistency and better understandability of design outcomes. Myers argued that since all user interfaces created with the same tool will be similar, such tools help to achieve a consistent look and feel (Myers et al. 2000). For instance, [Bootstrap](http://getbootstrap.com/)<sup>21</sup> and [Google Material Design](https://material.google.com/)<sup>22</sup> are frameworks for creating unified user experiences across diverse platforms and devices. While being very flexible, these frameworks come with predefined elements and resources, such as material icons, which can give applications a distinctive Bootstrap or Google material look and feel. This consistent look and feel, in turn, can make it easier for users to reuse their experiences from other applications across diverse channels.

Having design tools that impose strong constraints, while limiting designers, also can simplify the design of new design outcomes. Tools without such constraints may lead designers to create unnecessarily complex and difficult to use designs. Nielsen's 2010 report on the usability of iPad applications nicely illustrates this issue:

*“The first crop of iPad apps revived memories of Web designs from 1993 when Mosaic first introduced the image map that made it possible for any part of any picture to become a UI element. As a result, graphic designers went wild: anything*

---

<sup>21</sup><http://http://getbootstrap.com/>

<sup>22</sup><https://material.google.com/>

*they could draw could be a UI, whether it made sense or not. It's the same with iPad apps: anything you can show and touch can be a UI on this device. There are no standards and no expectations.” (Nielsen 2010)*

The main disadvantage of using tools with predefined elements is that they constraint the design outcome and may limit options and creativity of designers. Websites that use predefined templates may lack clear distinction in style from other sites, which may cause user dissatisfaction or users' perception of the company as unoriginal or amateurish. Consequently, tools with fewer constraints may lead to more innovative designs. As argued by Shneiderman et al. (2007), the success of Flash<sup>23</sup> user interface tool was partially based on the lack of predefined widgets. It stimulated innovative designs because it encouraged designers to explore different ways to control the interaction, instead of just using buttons and scroll bars. We may argue that lack of predefined elements in a user interface was a typical signature of the Flash tool.

The *your design is your reflection* pattern may also occur as a consequence of individual designer's background and preferences. In his examination of forces that generate a style in architecture, Chiu-Shui Chan (2001) argued that an architectural style is not only a similarity of physical features in a designed object. Instead, a style also reflects designers' personal aspects, including operations of cognitive mechanisms, utilization of repeated procedures, personal preference for specific images, and manipulation of certain seasoned design knowledge. Designers have their own unique sets of skills, knowledge, and experiences, and often they have developed their design styles. Moreover, sometimes a designer may be invited to design something exactly because of this style. That

---

<sup>23</sup>[https://en.wikipedia.org/wiki/Adobe\\_Flash](https://en.wikipedia.org/wiki/Adobe_Flash)

means that different design outcomes produced by the same designers will often have some standard features.

The *your design is your reflection* pattern may also be visible at the level of design processes and organizations. Established processes and organizational structures may influence the form of design outcomes. Melvin Conway, in what is nowadays known as *Conway's law*, stated that organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations (Conway 1968<sup>24</sup>). In other words, the structure of a system will follow the organization of the people that built it (e.g. see Lewis & Fowler 2014). When designing a complex system, two modules A and B cannot interface correctly with each other unless the designer and implementer of A can communicate with the designer and implementer of B. MacCormack et al. 2012 similarly explored the duality between product and organizational architectures, highlighting the impact of organizational design decisions on the technical structure of the artifacts that these organizations develop. This pattern may enable organizations to create particular types of design outcomes efficiently.

If an organization frequently produces a specific kind of design outcomes, having a stable, optimized organizational structure can be a significant advantage. When established organizational structures do not match well requirements of design outcomes, this pattern may lead to negative consequences. If the organizational structure makes it difficult to introduce particular changes, this pattern may lead to missed opportunities or lower quality. Gokpinar et al. (2010), in their study of vehicle development process of a major auto company, found out that mismatches between product architec-

---

<sup>24</sup><http://www.design.caltech.edu/erik/Misc/Conway.html>

ture and organizational structure lead to a number of quality issues. In other cases, this pattern may lead to unnecessary complexity. For example, Bass et al. (2012) reported a case in which a software architect was asked by the management to add a database component to the software solution without clear functional need. The main reason was that database department was overstaffed and underworked and they needed something to do (page 59). “Design by committee” is another example of unnecessary design complexity introduced by imposed organizational structures. Fred Brooks (2010) argued that outcomes of a design by committee lack focus and result in impractical products with too broad functionality. Brooks elaborated that the people in committees, to protect their interests and avoid conflicts, are often reluctant to reject any request:

*Each player has a wish list garnered from his constituents and weighted by his personal experiences. Each has both an ego and a reputation that depend on how well he gets his list adopted. Logrolling is endemic—an inevitable consequence of the incentive structure. “I won’t naysay your wish, if you won’t naysay mine” (page 41).*

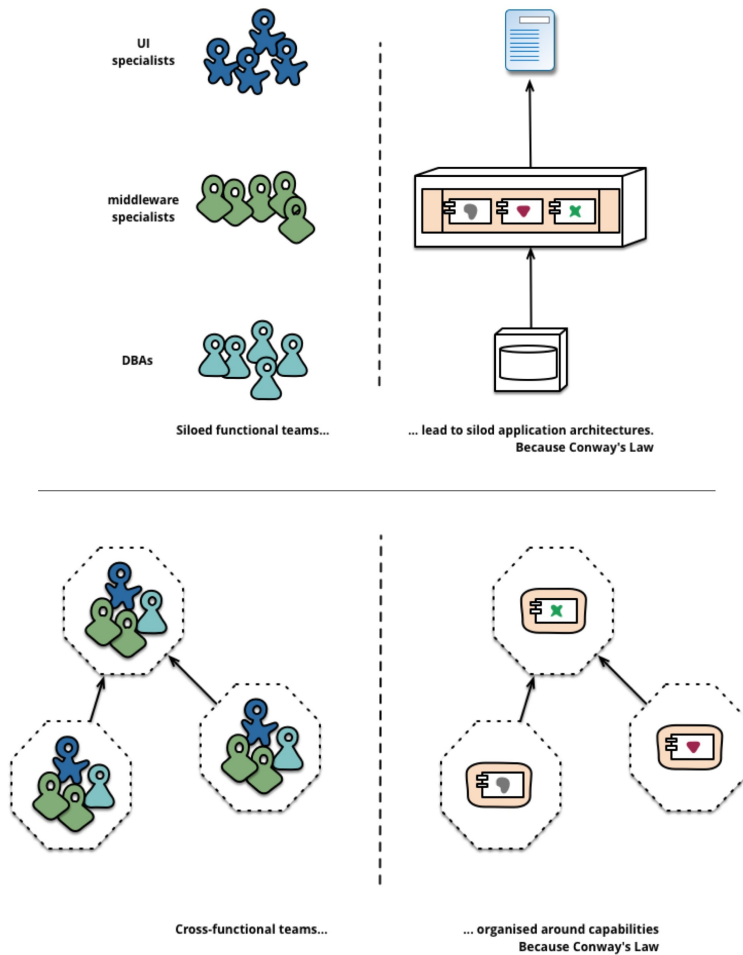


Illustration of the Conway's Law. Credit: Martin Fowler (<https://martinfowler.com/articles/microservices.html>).

The your design is your reflection pattern is a consequence of inertia (resistance to change) of design resources. The opposition to change of design resources is not necessarily

a problem. Having stable tools, habits, and organizational structures often make designing more comfortable, more predictable and faster. Such inertia may be a design decision, for example, if the goal is to create consistent design outcomes. But such inertia may happen implicitly, for instance, due to sheer size and complexity of used resources. Social dynamics in design teams and organizations also contribute to this pattern, such as in the case of the Conway's law. As noted by James Coplien (1999) in his discussion of Conway's law, *"[software] architecture is not so much about the software, but about the people who write the software. The core principles of architecture, such as coupling and cohesion, aren't about the code. The code doesn't 'care' about how cohesive or decoupled it is; ... But people do care about their coupling to other team members.."* The resistance to change means that the resources may be difficult or costly to change, often leading to a more manageable option of design outcome adaptations. However, too much inertia may lead to situations where design outcome becomes unnecessarily constraint or complex.

The *your design is your reflection* pattern is related to the *you are a reflection of your design* pattern. Design resources may get their "signature" form as a result of the need to support efficient creation of certain types of design outcomes. The *your design is your reflection* pattern is also related to the *conformity* pattern. Adopting popular design resources with established best-practices also means accepting particular constraints and ways of working with these resources.



## Questions to Consider

- Do you have your design style? How much do you care about it?
- Do you use templates and standard sets of technologies and materials in your designs?
- Do templates and standards help you to simplify and speed up your design efforts?
- Do these templates and standards limit your design efforts?
- How do you ensure consistency (when needed) in multiple design activities?
- Is your organizational structure aligned with the structure of your design outcomes? How?
- Have you ever experienced the effect of Conway's law, where your designs copied the communication structures of your organization?
- Have you ever build unnecessary complex designs due to wrong organizational structures?
- Have you ever experienced the effect of inverse Conway's law, changing your organization to suit the design outcome structure better?

## You Are Reflection of Your Design

**Motto:** *“We are what we repeatedly do.” (Will Durant, The Story of Philosophy)*

**Summary:** *Design resources are adapted to a design outcome, sometimes to the point that the form and organization of design resources reflects the intended shape of the outcome.*

**Topics:** *Specialization, Inverse Conway’s Law, Work Optimization*

**Alternative name:** *Outcome Signature*

**Desirable:** *In situations where efficiency of design work is essential.*

**Undesirable:** *In complex real-world situations and in new domains where it may limit designers’ ability to apply other approaches.*

---

The structure of design outcomes may influence the tools, methods, processes and whole organizations that create such outcomes. As in the case of the *your design is your reflection* pattern, this adaptation can happen at all levels, from individual design tools to complex organizational structures.

The following table illustrates levels at which this pattern may occur.



Level	Examples
Technology and tools	Tools and materials are selected and configured to facilitate the creation of particular outcome types.
Individual designers	Types of outcomes they worked on define designers' style, skills, and knowledge
Team and organization	Team organization and communication structures are defined to facilitate a particular outcome structure (inverse Conway's law).

Design tools often need to be adapted to create a particular design outcome. For example, most software design tools are configurable so that frequently used functions are more easily accessible (e.g., via shortcuts or toolbar buttons). In this way, the configured tool becomes optimized for particular tasks, making these tasks easier, and other more difficult. This tool configuration is a signature that working on a design outcome leaves on a design resource.

Working on a particular type of design outcomes also leads to improvements in designers' skills and knowledge. Working with the same tool to create similar design outcomes (e.g., mobile applications, web applications), will lead designers to develop a specific set of skills, biased at the efficient creation of such design outcomes. Once designers develop such skills, they are much more efficient in creating new similar design outcomes. This efficiency, in turn, may lead to other positive consequences, such as more rapid prototyping and

more iterations of iterative design that is a crucial component of achieving high-quality user interfaces (Myers et al. 2000; Nielsen 1993).

The *you are a reflection of your design* pattern also happens at the process and organizational levels. Bass et al. (2012), for instance, argued that software design often describes not only the structure of the design outcome. It can engrave the structure of the development project and sometimes the structure of the entire organization (page 33). They elaborated that a standard method for dividing up the labor is to assign to different groups different portions of the system to construct. Such work-breakdown structure in turn dictates:

- units of planning, schedule, and budget,
- inter-team communication channels,
- configuration control and file-system organization,
- test plans and procedures, and
- even minutiae such as how the project internet is organized and who sits with whom at the company picnic.

Sam Newman similarly provides an example of an organizational structure that grew up around the product architecture (Newman 2005, page 201). The organization was tightly aligned with the parts of the product. To make changes to the product, they first had to change the organizational structure.

The *you are a reflection of your design signature* pattern may be a consequence of the success of a product and the need to maintain and support the product in the long term. Bill Buxton (2007) claimed that the second worst thing that can happen to a new product (after failure) is that it is a huge success (page 208). Buxton elaborated that the more successful a product, the more of them are going to be produced and used.

That means that the company, clients, and successors have to deal longer with the design and architectural decisions that designers made. To be able to maintain and support successful products, organizations need to make long-term adaptations and optimizations. While such products may be profitable, they may also be a burden. Long-lived systems tend to grow in size and complexity over the time, and maintenance and support of such systems tend to consume more and more of the organization's resources. This consumption of resources may limit the ability of the organization to do other things and innovate. Adam Schneider, for instance, [noted](#)<sup>25</sup> that companies may become prisoners of their legacy systems. The need to support legacy systems prevents such companies from surging ahead in the marketplace, introducing new products, moving into new geographies, or expanding services to customers on new platforms.

The *you are a reflection of your design* pattern is related to the *your design is your reflection* pattern. Adaptations of design resources to support efficient creation of certain types of design outcomes often lead to long-term changes in these design resources. These long-term changes, in turn, may affect the form of new design outcomes in new projects where designers use the same resources. The *you are a reflection of your design* pattern is also related to the *commitment* pattern. Having tools, skills, and organizational structures optimized for creating a particular type of a design outcome may encourage designers to reuse their tools and skills in new design situations. But this optimization may also lead to the *cherry picking* pattern, and stimulate designers to choose only those situations where a specific type of design outcome is

---

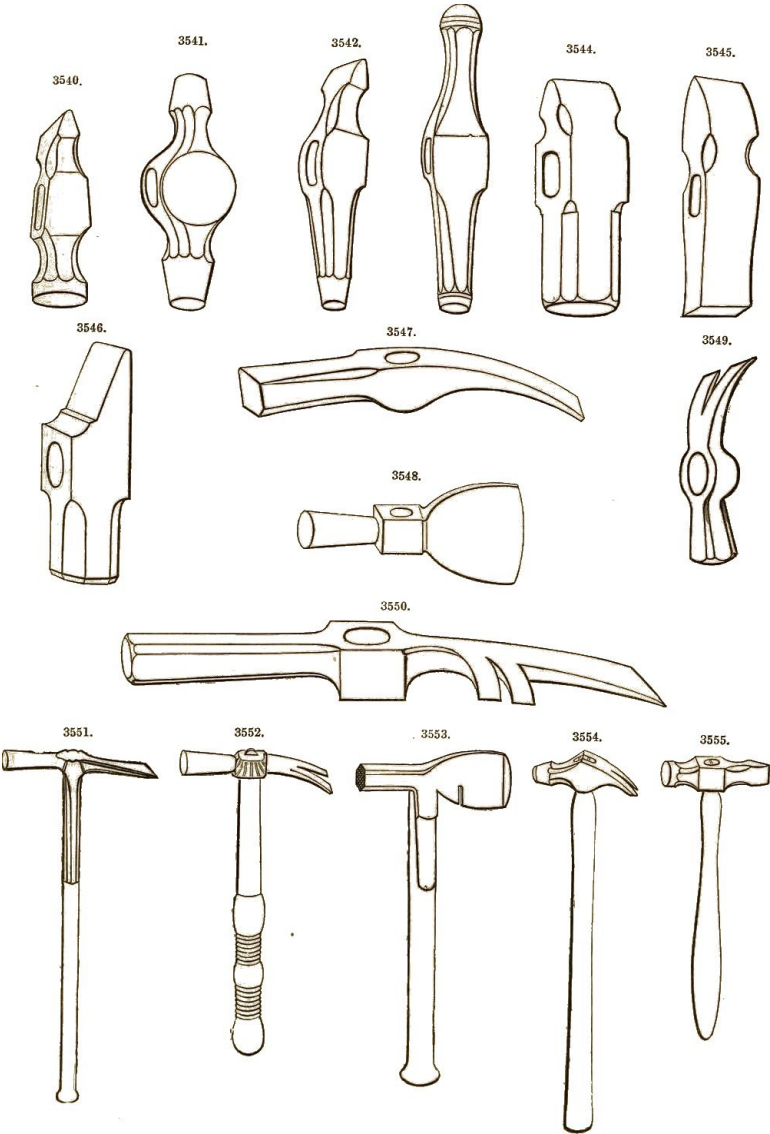
<sup>25</sup><http://deloitte.wsj.com/cio/2013/10/01/when-companies-become-prisoners-of-legacy-systems/>

required.



## Questions to Consider

- How do you choose technologies, materials, and tools for your designs?
- Do you have preferred tools and techniques?
- Have you ever obsessively used some technology or method, even though it was not the most appropriate for the task at hand?
- Do you adapt your organizational structures for your designs?
- How much effort you spend maintaining your existing designs versus creating the new ones?
- Are you being slowed down by the need to maintain your previous (legacy) designs?



Hammer-heads optimized for different purposes. Credit: Wikimedia Commons.

## Design-by-Buzzword

**Motto:** *“Projects that do not capitalize on new opportunities will generally find their products unable to compete.”* [Boehm and Bhuta \(2008\)](#)

**Summary:** *A designer is joining a growing trend in using some technology, often in an opportunist way. New possibilities of design resources are shaping the design outcome.*

**Alternative names:** *Design-by-Buzzword, Opportunistic Design.*

**Topics:** *exploiting new opportunities, hype, new trends, hackathons*

**Desirable:** *In fast-changing domains where capitalization on new opportunities is a crucial competitive advantage.*

**Undesirable:** *In real-world situations and in well-known domains where it may lead to unnecessary complex solutions.*

---

The *design-by-buzzword* pattern is a form of dynamics between design outcomes and design resources where designers use a design situation as an opportunity to try out some novel or upcoming design resource (e.g. [Ncube et al. 2008](#)).

The *design-by-buzzword* pattern is a consequence of technology developments, designer’s creativity in using and applying new resources, as well as changing expectations of users who are continually learning about technology advances. Introduction an iPhone, for instance, radically changed user’s expectations of mobile applications (e.g., see [“How iPhone Changed the World”<sup>26</sup>](#)).

---

<sup>26</sup><http://www.cultofmac.com/103229/how-iphone-changed-the-world/>



The iPhone family. Introduction an iPhone radically changed user's expectations of mobile applications. Credit: Apple.

New technology opportunities are one of the main reasons for a constant change of designs, especially in software design. [Boehm and Bhuta \(2008\)](#) argued that software projects that do not capitalize on new opportunities will generally find their products unable to compete. Earlier [Brooks \(1995\)](#) noted that new opportunities in technology are one of the primary drivers behind constant changes of software designs: *"If not new computers, then at least new disks, new displays, new printers come along; and the software must be conformed to*

*its new vehicles of opportunity*” (page 12).

In the *design-by-buzzword* pattern, the primary driver for the use of some resources is the “hype” and high expectations of some new trend. Such usage usually is not directly driven by the requirements of the situation or experiences in using the resource in similar cases. Nowadays, bitcoin and blockchain technologies are examples of such hype. Financial institutions are investing in these technologies, but the actual value and practical utility of these technologies are still unclear.

Novelty rather than experience drives the *design-by-buzzword* pattern. The lack of experiences means that there are no clear guidelines and best practices, which may lead to unnecessary complex projects that are more likely to fail. Roy Thomas Fielding (2000) noted that software projects often begin with adoption of the latest “fad” in architectural design. Only later, if at all, it is discovered whether or not the system requirements call for such an architecture. As usage of new resources may lead to more unexpected and undesired effects, projects based on the *design-by-buzzword* pattern are also more likely to fail.

Design-by-buzzword introduces a number of risks. Designers need to work with resources that are not necessarily proven in practice, and for which there are little experiences that can be reused. On the other hand, avoiding using new resources may impact the long-term competitiveness of an organization. Consequently, organizations are looking for ways to combine benefits of being an early adopter of some technology or technique, while limiting the risk. Hackathons are an increasingly more popular form of making such compromise (Komssi et al. 2015). Here the risk is minimized by time-boxing the activity, normally for several days only, with the idea that this limited time frame gives enough time to get basic insights and some



hands-on experience with a new trend.

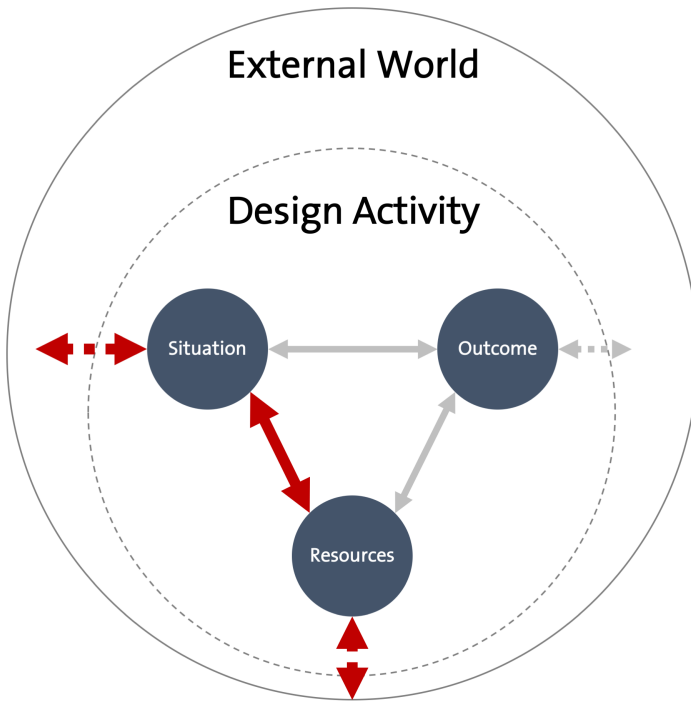
The *design-by-buzzword* pattern is related to the *solution looking for a problem* pattern. New opportunities embedded in an innovative design outcome may lead to usages of this outcome in new situations. The *co-evolution of problem-solution* pattern may also be related to the *design-by-buzzword* pattern as a way to develop a better understanding of useful applications for new design resources.



### Questions to Consider

- Do you follow the latest trends in design tools and materials, and are you quick to adopt them?
- Have you ever regretted being an early adopter of the latest fads?
- Have you ever regretted not reacting soon enough to the latest trends?
- How do you make decisions to adopt new technologies and methods?
- Do you have organized events, such as hackathons, to systematically, but with limited risk, explore new opportunities?

## C. Patterns of Dynamics Between Design Situations and Design Resources



Design situations and design resources relate to each other in many ways. A designer is often pictured as a craftsman, someone who picks and chooses tools freely based on the case and grounded in the judgment of overall benefits from using a specific resource (Stolterman 2008). Some resource selection criteria are influenced by the specifics of the design situation at hand, and the designer has limited influence on

them. Budget limitations, for instance, can affect the selection of materials and consequently tools and processes to work with these materials. The deadline imposed by clients may restrict the number of design iterations and limit the amount of testing and evaluation. Size and criticality of a situation may ask for different software development approaches (e.g. [Cockburn 2000](#)).

The choice of design resources is never trivial and never a sole function of the situation requirements. The ‘benefits’ of using some tools also depend on the level of skill and mastery required, external pressure about standards, personal style of expression. The designer is also almost never in full control about which resources will be used and how they will be adapted to a particular situation.

We discuss the following patterns of these dynamics between design resources and situations:

- Conformity,
- Commitment, and
- Cherry Picking.

## Conformity

**Motto:** *“When in Rome, do as the Roman’s do.”*

**Summary:** *To minimize risks in a new design situation, designers use popular and proven resources with established best practices. Such conformity may be driven by positive experiences of others in similar situations, mere popularity of design resources, or mandatory requirements. The usage of a design resource further contributes to establishment and reputation of the resource and its usage in new design situations.*

**Topics:** *popularity, compliance, communities, fashion, peer pressure, legal requirements, education*

**Desirable:** *When conformity is mandatory, or when designers can benefit from joining a broader community.*

**Undesirable:** *In new domains where there are no established best practices, or in areas where capitalization on new opportunities is crucial.*

---

In the *conformity* pattern, designers select popular and proven resources with established best practices. In this pattern, external factors, such as experiences of others, popularity, or regulations, are primary drivers for the selection of design resources.

The primary driver behind conforming to established resources and best practices is the wish to minimize risks. Established and popular design resources and reference projects provide some evidence that the design resources are suitable

for particular situations. Consequently, a common strategy to select design resources is to look at successful design projects addressing similar conditions and to use the same resources.

Conformity has practical advantages for designers. Well-known tools usually have active communities of practitioners and plenty of learning resources, enabling a designer to more efficiently and predictably master the resource usage. Conformity also has advantages for organizations, as it is easier to find new employees for mainstream tools.

The mere popularity of some design tool or methodology is often the main reason for its usage. For instance, while there are hundreds of libraries and frameworks for the development of web user interfaces, most new web user interfaces are built using few most popular frameworks: React, Angular and Bootstrap. In the domain of graphic design, similarly there are hundreds of available tools, but most designers use [Adobe](#)<sup>27</sup> tools (e.g., Photoshop, Illustrator, and InDesign) by far the most popular graphic design tools, often described as *de facto* industry standard for graphics editing.

The most popular design resources are often introduced at schools and universities, further contributing to the popularity and widespread use of these resources. Adobe, for example, offers [educational resources and discounts](#)<sup>28</sup> on their products for teachers and students. In software engineering schools, Java, one of the most popular programming languages, has often been used as the first programming languages that students learn ([Hadjerrouit 1998](#)). Before that, the popularity of C++ in the industry has led to its use for teaching as well ([Kölling \(1999\)](#)).

---

<sup>27</sup><http://www.adobe.com>

<sup>28</sup><http://www.adobe.com/education.edu.html>

Usage of some resources also influences the future development of that resource. By using and buying a particular resource, designers increase the popularity of the resource and the profit of the producer, which can further stimulate the use of the tool. Successful projects are excellent references for resources used in them and may be the main reason why others will select these resources for new projects.

Conformity may also be a result of legal and other regulations. Physical materials, for instance, need to satisfy rules about safety and environmental impact and there is often a limited list of “certified” or recommended materials. Here conformity is more than a question of style. Failure to use recommended materials may have severe consequences and put lives of people in danger, as in the case of fire safety standards. In software design, companies often also have a list of approved resources, tools or libraries. Security concerns are often the driver of these limitations. For instance, in 2017 the US Government banned Kaspersky Security software, limiting designers of network and application security to other options. A similar situation happened in 2019 when the U.S. banned its companies from using foreign telecoms providers, in particular the Huawei equipment in 5G mobile networks.

Designers may conform to the usage of some resources due to fear of legal and other consequences. Harrison (2004) for instance, argued that if a software system fails and customers sue, it may be viewed as negligence if the designer’s organization did not follow best practices. For this argument, Harrison draws a parallel between software design and criminal trial. In a criminal trial (at least in the US), the failure to follow an established best practice could result in an acquittal.

The *conformity* pattern is particularly useful in standard,

well-known domains. As discussed in the *co-evolution of problem-solution* pattern, designing custom solutions may lead to “reinventing-the-wheel” and “not-invented-here” anti-patterns. In well-established domains, designers often can use standard solutions with minimal modifications.

Conformity also has its downsides. One negative issue is that by using established and proven resources, designers may not capitalize on some new emerging technology. As discussed in the *design-by-buzzword* pattern, projects that do not capitalize on new opportunities may find their products unable to compete. Another negative issue is the *vendor lock-in*<sup>29</sup>, as designers and customers may become too dependent on a vendor for products and services, and are unable to use another vendor without substantial switching costs. Popular tools may also constrain a designer as they need to stay relevant for many users and may decide to focus on developing only the most common features.

The *conformity* pattern is related to the *commitment* pattern, as successful usages of some new design resources require some level of commitment. Design organizations may also commit to using some standard or favorite resource. The *conformity* pattern is also related to the *cherry picking* pattern. A wish to avoid risks may lead design organizations to work only in situations where proven resources with established best practices can be applied.

---

<sup>29</sup>[https://en.wikipedia.org/wiki/Vendor\\_lock-in](https://en.wikipedia.org/wiki/Vendor_lock-in)



## Questions to Consider

- Do you use standard tools and methods with a significant user base?
- Are you an early adopter for some technologies and methods?
- Are your practices, tools, and methods similar to practices, tools, and methods of other companies or institutions?
- Do you have some unique practices, tools, and methods? Why?
- Do you regret conforming to some standard practices, tools, and methods, as they limit you?
- Are you dependent on a vendor for products and services, and are unable to use another vendor without substantial switching costs?
- Have you ever needed to change your design due to government decisions?





Three examples of buildings in the Amsterdam School style, a style of architecture popular from 1910 through about 1930 in the Netherlands. Buildings of the Amsterdam School are characterized by brick construction with complicated masonry with a rounded or organic appearance, relatively traditional massing, and the integration of an elaborate scheme of building elements inside and out. Credit: Amsterdam Municipal Department for the Preservation and Restoration of Historic Buildings and Sites (bMA) / Wikimedia Commons.

## Commitment

**Motto:** *“It is a poor craftsman that blames his tools.”*

**Summary:** *In a new design situation designers use tools they committed to beforehand.*

**Topics:** *learning, standardization of work, tool development, “eating its own dog food,” “if all you have is a hammer, everything looks like a nail”*

**Desirable:** *When designers want to master resource usage or further improve the resource itself.*

**Undesirable:** *In complex real-world situations where over-commitment may lead to worse design outcomes and lost opportunities.*

---

The *commitment* pattern is a form of dynamics of design activities in which, in a new design situation, designers commit beforehand to using a particular resource. And they persist in the usage of the resource even when they experience difficulties.

Designers may commit to and persist in using a particular resource due to a number of reasons. Such commitment, for instance, can facilitate the development of design skills. Learning to use a new design tool requires time and effort, and initial results may be discouraging. Without the commitment and persistence, a designer may give up learning too early, and never master the tool.

Most design companies have a set of their standard or recommended resources. If a whole organization commits to using

a particular resource, it may have a positive effect on its operations. More people will share the same knowledge and skills, learn from each other, and in general easier communicate and collaborate. Ken Britton<sup>30</sup>, for instance, argued for the value of keeping the number of technologies organizations use to a minimum. With fewer technologies, the organizations can more easily train new people, maintain code, and support moves between teams. Sam Newman (2015<sup>31</sup>) similarly described that Netflix<sup>32</sup>, has mostly standardized on Cassandra<sup>33</sup> as a data-store technology (page 16). Although it may not be the best fit for all of its cases, Newman elaborated that Netflix felt that the value gained by building tooling and expertise around Cassandra was more important than having to support and operate at scale multiple other platforms that may be a better fit for certain tasks.

A special case of the *commitment* pattern is the “*eating its own dog food*” pattern (Harrison 2006). To demonstrate confidence in their products, some design companies are forcing themselves to use their own products. A famous example is the 1980’s memo sent by the Apple Computer’s president Michael Scott. In the memo, Scott asked employees to stop using typewriters and instead use Apple’s word processing software (Ditlea 1981).

An over-reliance on a familiar resource has its downsides, famously expressed as “*if all you have is a hammer, everything looks like a nail*”. Each tool and process comes with different concepts and abstractions, giving designers “lenses” through which they can view the design situation. If these “lenses” are too focused, designers may over-emphasize the

---

<sup>30</sup><http://queue.acm.org/detail.cfm?id=3131240>

<sup>31</sup>Newman2015

<sup>32</sup><https://www.netflix.com/>

<sup>33</sup><http://cassandra.apache.org/>

parts of the situation that fits nicely in their process or their tool. Or they may under-emphasize the parts where there is a less obvious match. Designers may also create unnecessary complex solutions (e.g. see [Plauger 1992](#), and discussion on the [Second-System Effect](#)<sup>34</sup> in [Brooks 1995](#)). In software design, there is a similar notion of a silver bullet or golden hammer, “*a familiar technology or concept applied obsessively to many software problems*” ([Brown et al. 1998: p 111](#)). Unified Modeling Language (UML) and Extensible Markup Language (XML) are examples of technologies that have been used almost “obsessively” in software design. These two technologies are valuable tools for software design. But in some cases, such tools are not effective or sufficient. For instance, [Bell \(2004\)](#) discussed “UML fever,” describing that it is not uncommon for people to believe that usage of UML guarantees the value of the artifacts produced. Similarly, in their article on “XML fever,” [Wilde and Glushko \(2008\)](#) argued that some view XML as a magical solution of universal interoperability of information producers and consumers. They elaborated that much more is needed to achieve interoperability than simply using XML.

In all of these patterns, there is the pre-commitment to some resource. On a positive side, this commitment may enable development of mastery in resource’s usage and stimulate improvements of the resource itself. On the negative side, the over-commitment may lead to worse design outcomes and lost opportunities.

The *commitment* pattern is related to the *conformity* pattern, as design organizations may commit to using some resource out of conformity. The *commitment* pattern is related to

---


<sup>34</sup>[https://en.wikipedia.org/wiki/The\\_Mythical\\_Man-Month#The\\_second-system\\_effect](https://en.wikipedia.org/wiki/The_Mythical_Man-Month#The_second-system_effect)

the *cherry picking* pattern, because a designer may choose situations in which resources he committed to may be more readily applied.



### Questions to Consider

- How much freedom do you have in choosing your design resources?
- Do you have some design resources you are committed to using most of the time?
- How do you learn new practices, tools, and methods? When do you give up?
- Do you “eat your own dog food”?
- Have you ever regretted committing too much to some practice, tool, or method?
- Do you have a golden hammer, applied obsessively to many problems?

Mike Scott Memo • No More Typewriters • 01 February 1980	
	<b>Inter Office Memo</b>
<u>YOU ALL BETTER READ THIS</u>	
Date:	February 1, 1980
To:	Purchasing and Everyone
From:	Mike Scott <i>ms</i>
Subject:	Typewriters
<u>Effective Immediately!! No more typewriters are to be purchased, leased etc., etc.</u>	
Apple is an innovative company. We must believe and lead in all areas. If word processing is so neat, then let's all use it!	
Goal: By 1-1-81 No typewriters at Apple. (Ken, get rid of the DEC word processor ASAP)	
Brownie Points: Typewriter users giving up their machines in favor of Apple II-Apple Writer Systems will get first priority on new Apple high performance systems. Those who can justify direct typing capabilities and will turn in their typewriter will get first Qume with Keyboard/Apple installations.	
We believe the typewriter is obsolete. Let's prove it inside before we try and convince our customers.	
cc: Executive Staff All Typewriter Users	

Apple Computer Historical Information	Page 0002 of 0002
---------------------------------------	-------------------

Mike Scott Memo - No More Typewriters. Credit: archive.org

## Cherry Picking

**Motto:** *“Do one thing and do it well.”*

**Summary:** *Design situations are selected based on how quickly designers can approach these situations with preferred or available resources. Other situations are avoided.*

**Topics:** *focusing usage of resources, work efficiency, minimizing risks*

**Alternative names:** *Low-Hanging Fruits, Selective Design.*

**Desirable:** *In new domains where there are plenty of opportunities or little competition, or when an organization needs to focus usage of scarce resources.*

**Undesirable:** *In complex real-world situations as it may stimulate solving easy instead of important problems.*

---

In domains where designers can choose situations to work on, they may decide to select only those situations where they can use their preferred resources (e.g. [Stolterman and Pierce 2012](#)). Designers may choose to specialize and avoid involving in unfamiliar situations where they may not be able to use familiar resources.

Cherry picking may have a positive effect on operations and success of designers and their organizations. [Ries and Ries \(2002\)](#), for instance, claimed that companies are likely to be more successful if they focus their energies on one specific skill instead of trying to master many different ones (page 8).

Limitation in resources may also stimulate designers to select situations where they expect more benefits or have less risk of

failure. In [the business context](#)<sup>35</sup> “cherry-picking” is used as a practice to identify and target the most profitable customers in a market, rather than serving them all. Cherry-picking the most attractive customers is a common approach for startups since they can focus their limited resources and not disperse them over the whole segment. In [the financial world](#)<sup>36</sup>, “cherry picking” is also used as a strategy by investors to choose investments that have performed well within another portfolio in anticipation that the trend will continue. Here, cherry-picking reduces the amount of time required for researching stocks because the pool of securities in which investors pick from is narrowed.

Cherry-picking also may have negative effects. One problem is that by cherry-picking designers may be solving issues that are less relevant to the client. This problem is especially visible in domains where a rigor in applying a specific method is valued more than the practical value of a design outcome, such as in academic environments. [Greenberg and Buxton 2008](#), for example, argued that the [ACM CHI conference](#)<sup>37</sup> has a methodology bias, where certain kinds of methods are considered more ‘correct’ and thus acceptable than others. The consequence is that people now likely generate ‘research questions’ and ‘designs’ that are amenable to a chosen method, rather than the other way around: *“That is, they choose a method perceived as ‘favored’ by review committees, and then find or fit a problem to match it. ... That is, researchers first choose the method (e.g., controlled study) and then concoct a problem that fits that method.”* In this case, the selection of situations is more influenced by the tools and techniques that researchers like than by the importance of the

---

<sup>35</sup><https://strategichinker.wordpress.com/cherry-picking/>

<sup>36</sup><http://www.investopedia.com/terms/c/cherrypicking.asp>

<sup>37</sup><https://sigchi.org/conferences/>



problems.

The *cherry picking* pattern is related to the *commitment* pattern, because a designer may choose situations in which resources he committed to may be more readily applied. The *cherry picking* pattern is also related to the *conformity* pattern. A wish to avoid risks may lead design organizations to work only in situations where proven resources with established best practices can be applied.



### Questions to Consider

- On what types of design situations you work on usually?
- Do you specialize in any way?
- Do you say no to design opportunities outside your specialization?
- How do you make sure not to overspecialize and become irrelevant once your specialization is not anymore needed?



Cherry picking. Designers may choose problems based on their specialization or due to resource limitations. Credit: Charles Nadeau / Wikimedia Commons / CC-BY-2.0.

## D. Emergence of Complex Patterns of Dynamics

*“Every place is given its character by certain patterns of events that keep on happening there.”* (Christopher Alexander, *The Timeless Way of Building*)

The patterns of dynamics of design activities may facilitate describing complex dynamics between elements of design activities as a combination of several higher-level arrangements of such dynamics. With patterns, we can explain, compare, and contrast, diverse and complex design activities in similar terms and on a higher level of abstraction.

We can, for instance, talk about some typical combination of patterns. When a new situation appears, designers’ attempt to change that situation may lead to the *co-evolution of problem-solution* pattern. Designers may need new or better resources for this situation, which will lead designers to change their ways of working and to the emergence of the *you are a reflection of your design* pattern. Through *commitment* and *eating its own dog food* patterns the new resources can further develop and gain popularity. Once stable and popular the resources may become a *de facto* standard, and lead to the *conformity* pattern. The *conformity* pattern may lead to the *your design is your reflection* pattern. The *your design is your reflection* pattern, on the other hand, may trigger a reaction of customers wanting something new and original, which may result in new design situations.

The patterns of dynamics may be useful to discuss different levels of difficulties of design. Or, to paraphrase the Christopher Alexander’s quote, we can argue that every design

activity is given its character by patterns of dynamics that keep on happening there. Similar to Gero (1990), we may talk about routine design vs. innovative design, later generally considered more difficult. In routine designs, designers normally have better-defined situations, and they are using familiar and proven tools. There we may primarily expect to see the *commitment* pattern and possibly the *puzzle solving* pattern, with little of other patterns occurring. A routine design may also be the consequence of the *cherry picking* pattern. In innovative designs, on the other hand we may expect to see the *co-evolution of problem-solution* pattern, with possibly the *design-by-buzzword* and *solution looking for a problem* patterns. The innovative or creative design is also more likely to lead to the *force awakens* and *pandora's box* patterns, as we are dealing with more novel and unknown elements.

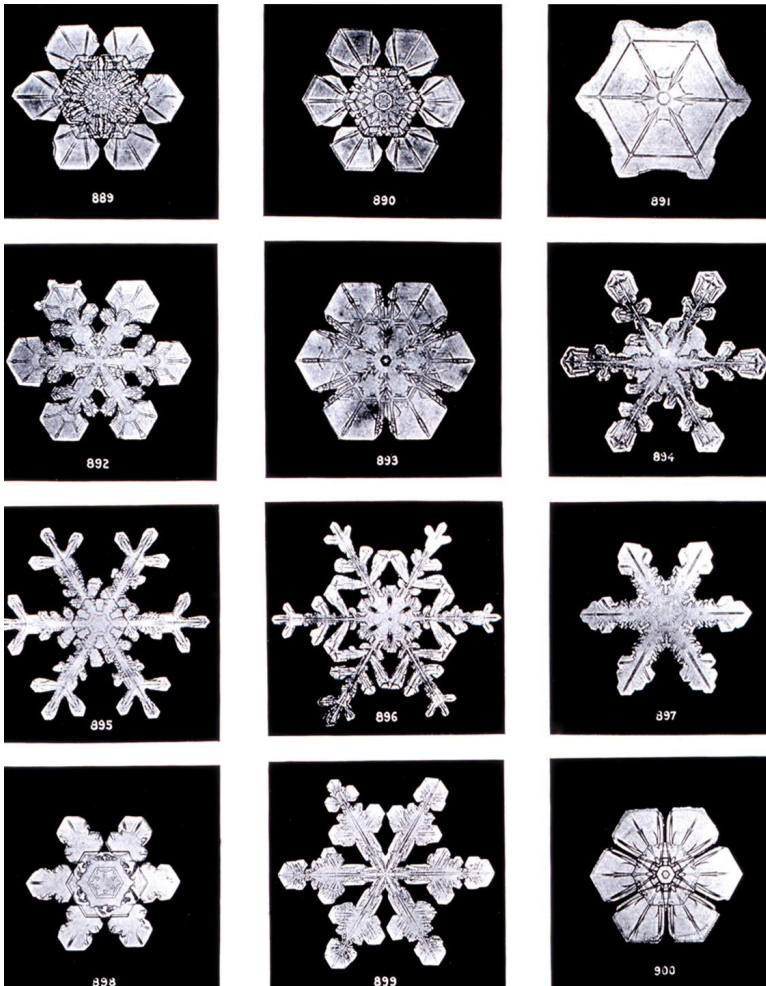
The patterns of dynamics strongly suggest that designers need techniques to estimate both the short-term and long-term impacts of design. In this aspect, we support the view of Penzenstadler et al. (2014), who argued that software designers could considerably improve civilization's sustainability by taking into account not just the first-order impacts of software systems but also their second- and third-order impacts. We believe that our framework could support such reasoning. The patterns illustrate that design activities have both a short-term and long-term dynamics. The short-term dynamics are typically localized within a particular project and are occurring in a limited period. The long-term dynamics are happening over a more extended period and among multiple projects. For example, the *co-evolution of problem-solution* pattern usually is intrinsic to the design activity and people involved in it, and the pattern is occurring during a concrete project.

The *force awakens* pattern, on the other hand, is primarily a consequence of indirect interactions through many elements outside the main design activity. Some patterns reflect both short-term and long-term dynamics. For example, the *eating its own dog food* pattern has an immediate short-term effect on the project where it is being used. But because it promotes and stimulates the development of specific design resources, on the long-term it may influence other design projects.



### Questions to Consider

- Which patterns of dynamics typically occur in your design activities?
- Are there some patterns that occur more often than the others?
- Are there some patterns that do not occur at all?
- How do you estimate the potential impact of your design outcomes, both short and long term?



The formation of complex symmetrical and fractal patterns in snowflakes exemplifies emergence in a physical system. In philosophy, systems theory, science, and art, emergence is whereby larger entities arise through interactions among smaller or simpler entities such that the larger entities exhibit properties, the smaller/simpler entities do not display. Credit: Wikimedia Commons.

# Part III: Sources of Dynamics

Dynamics of design activities are caused and influenced by diverse factors. Designers have to make decisions in a limited time based on the imperfect information they have. They are working in situations that are constantly changing due to forces beyond their control. Designers are using and relying on knowledge and skills that they do not fully understand and cannot explain. And at the same time, they do not have any strong underlying principles, theories, and laws to guide them.

In this part, we discuss our view on the main reasons behind occurrence of patterns of dynamics between design situations, design outcomes and design resources. We address the following topics:

- The moving target problem,
- Increasing interconnectivity,
- Learning and creativity,
- Reliance on tacit knowledge and skills,
- Social dynamics, and
- Lack of strong theories and principles.

Our primary goal is to provide an initial categorization of sources of design dynamics and to relate them to the patterns from the previous chapter. Our overview of sources of design dynamics is not intended to be complete, but rather a starting point and the proposed classification is not a clear-cut.

## A. The Moving Target Problem

**Summary:** *Design situations, resources, and outcomes change more rapidly than designers' understanding of them. Consequently, designers need to work without having significant experience with, and understanding of, design situations, resources, and outcomes. Waiting to obtain more knowledge and better understanding is risky as the issues may quickly become less relevant or obsolete.*

**Related Patterns:** *the force awakens, co-evolution of problem-solution, design-by-buzzword.*

The world surrounding a design activity is continuously changing. New materials, devices, computers and software products are continuously appearing, introducing new opportunities. Because these changes may happen rapidly, designers have limited time to create a useful design outcome. Design outcomes, especially those involving computing technologies<sup>38</sup>, have a limited lifespan, and may quickly become irrelevant or obsolete.

The constant and rapid change also means that designers who spend too much time on understanding a design situation and are hesitant to make decisions may end up creating a solution for an outdated problem. Myers, for example, described the problem that constant development of new technologies poses for the design of user interface tools (Myers et al., 2000). They argued that it is difficult to build tools without having significant experience with, and understanding of, the tasks they support. However, the rapid development of new interface technologies and techniques can make it difficult for tools to

---

<sup>38</sup>See <http://pages.experts-exchange.com/processing-power-compared/> for the effective visualization of the 1 trillion-fold increase in processing power from 1956 to 2015.



keep pace. By the time a new user interface implementation task is understood well enough to produce useful tools, the task may have become less critical, or even obsolete.

A similar note we may find by Harlan Mills, who wrote about difficulties of designing new programming techniques with high-pace of development of hardware ([Glass 2006](#)):

*“It’s too bad that hardware grew so fast. You know, if we’d had these new processors ... for 50 years at least, mankind could really have learned how to do sequential programming. But the fact is, by the time that [Edsger] Dijkstra comes out with structured programming, we’ve got all kinds of people using multiprocessors with interrupt handling, and there’s no theory behind it, but the IBM’s, the DEC’s, the CDCs, and so on, they are all driving forward and doing this even though nobody knows how to do it.”* (page 241)

The importance of making timely decisions in software design was discussed by [Richardson \(2011\)](#). He claimed that a software architect could spend a great deal of time designing what he thinks is a perfect system, only to present it when it is irrelevant. To illustrate the challenges of design decisions in a changing environment, Richardson uses an analogy of a hurricane meteorologist:

*“If you give a hurricane meteorologist a giant pile of data about a storm spinning in the middle of the Atlantic Ocean and ask her to determine exactly where it will come ashore, she can analyze the data, construct a detailed and accurate model of the atmospheric conditions and weather patterns, run some simulations, and come up with a forecast — two months after the storm hits. It’ll probably be wrong, but not by much, a moot point for the people in the storm’s path. The problem isn’t with the forecast’s accuracy but with the time needed to prepare*

*it. ... The model favored by hurricane meteorologists is to do just enough data acquisition and analysis to be reasonably certain what the storm will do and then start telling people to get ready.”*

In addition to the high pace of external changes, constant changing of design outcomes may also create internal difficulties for designers. Changeability of software (e.g., see [http://sebokwiki.org/wiki/The\\_Nature\\_of\\_Software](http://sebokwiki.org/wiki/The_Nature_of_Software)<sup>39</sup>, for instance, leads to continually evolving products. This evolution almost always comes with increased volume and internal complexity of such products. At some point, the sheer size and pace of growth of our systems may overpower our ability to understand them. Ebert and Jones (2009), for example, found out that premium cars in 2009 have 20 to 70 electronic control units with more than 100 million code instructions, totaling close to 1 Gbyte of software. Ebert and Jones also reported that similar embedded systems grow in size 10% to 20% per year. The larger the system, the more significant the number of both, its static elements (the discrete pieces of software, hardware) and its dynamic elements (the couplings and interactions among hardware, software, and users; connections to other systems). The increase of the number of system's components increases the possibility of errors because at some point no one can completely understand all the interacting parts of the whole or can test them (Charette 2005). Or, as noted by Bob Colwell (2005) it is always possible to design something too complicated to get it right.

Constant change contributes significantly to the dynamics of design activities because it forces designers to work without extensive experience with and deep understanding of a design situation. The *force awakens* pattern, for instance, is often

---

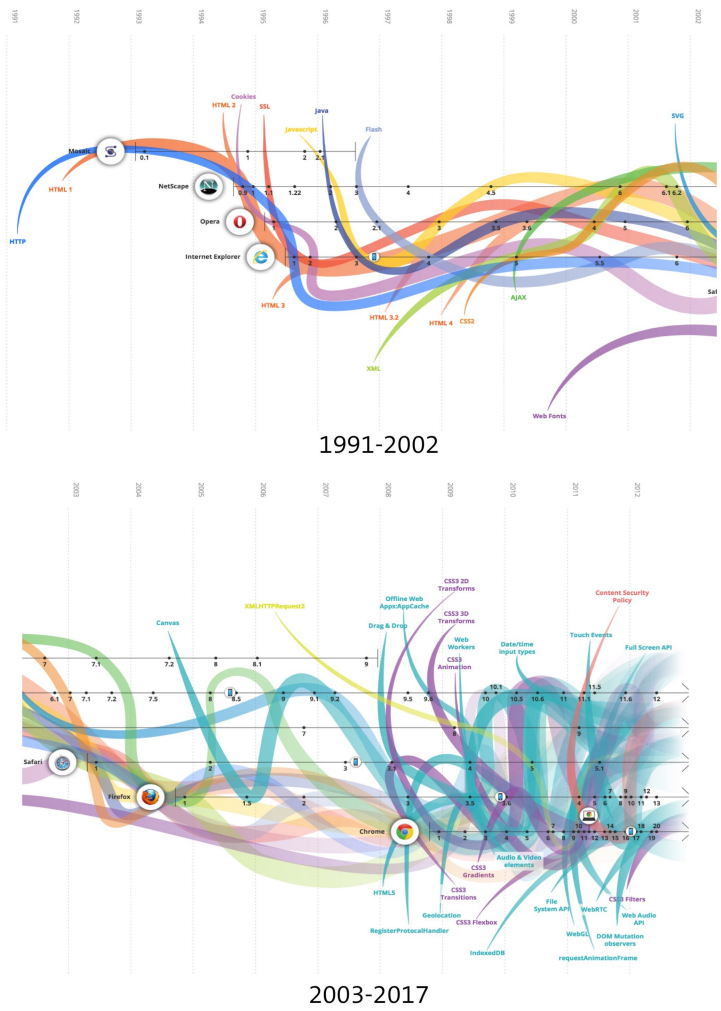
<sup>39</sup>[http://sebokwiki.org/wiki/The\\_Nature\\_of\\_Software](http://sebokwiki.org/wiki/The_Nature_of_Software)

a consequence of rapid changes in the external world as a reaction to a design effort, as well as designer's lack of understanding and experience with similar situations. The *co-evolution of problem-solution* pattern partially happens because, due to the fast pace of change, designers are dealing with new situations for which there is little previous projects and experiences. On the other hand, the *design-by-buzzword* pattern, may contribute to the constant change, as it aims at introducing new opportunities that may significantly change design situations.



### Questions to Consider

- Are you working in a dynamic domain where things change quickly?
- How do you balance the need to spend more time understanding the situation and designing the outcome versus producing results rapidly?
- How do you deal with uncertainty in design situations?
- How do you react to change?
- How do you respond to significant unforeseen problems?
- Have you ever canceled your design effort due to significant changes in design situation or resources?



The snapshot from the <http://www.evolutionoftheweb.com/> website. The website illustrates the constant evolution of existing web technologies, and frequent introduction of new web technologies.

## B. Increasing Interconnectivity

**Summary:** *Design situations, design outcome, and design resources are increasingly more globally interconnected. Increasing interconnectivity means that there are more ways or channels through which a design activity may be influenced.*

**Related Patterns:** *the force awakens, pandora's box, conformity, design-by-buzzword.*

Design situations are never isolated from the outside world. Our competition is closely monitoring the market developments, including our activities, and is reacting to them. Markets are more globalized. Furthermore, events in remote parts of the world, not directly related to design activity, may have a significant impact on it. One extreme example is [the global financial crisis of 2008](https://en.wikipedia.org/wiki/Financial_crisis_of_2007%E2%80%932008)<sup>40</sup>. While started as a consequence of the bursting of the U.S. (United States) housing bubble, the crisis affected almost all countries in the world. The crisis played a significant role in the failure of key businesses, declines in consumer wealth estimated in trillions of U.S. dollars, and a downturn in economic activity. It led to the Great Recession of 2008–2012 and contributed to the European sovereign-debt crisis. Impact of this global event on design activities has been significant, with many canceled and postponed projects, and budgets changed.

With the increase in the number of globally connected devices and advanced in communication technologies, design outcomes are more and more connected. Many of the current software systems, for example, are globally accessible web applications. Most of the current websites, in addition to being globally available, also offer sharing buttons, connecting their

---

<sup>40</sup>[https://en.wikipedia.org/wiki/Financial\\_crisis\\_of\\_2007%E2%80%932008](https://en.wikipedia.org/wiki/Financial_crisis_of_2007%E2%80%932008)

content to social media sites, further increasing their interconnectivity. And with initiative such as Internet-of-Things (IoT), even small devices such as temperature sensors in our houses, can now be connected to a globally accessible network (Gubbi et al. 2013). Anticipating behavior of design outcomes in a globally connected network is an increasingly more difficult task.

People themselves are more interconnected. Clients and users are following the news from all around the world, and are continually learning about new technological possibilities, adapting their expectation. Designers are also internationally connected to each other. Design education programs are mostly international. The rise of online education enables learning of new design tools and methods by a broad geographically distributed audience.

Increasing interconnectivity on a global scale may lead to significant changes in short time. Bruce Schneier's described how one simple message that was naively distributed caused \$2.54 billion in market capitalization to disappear and reappear within just a few hours:

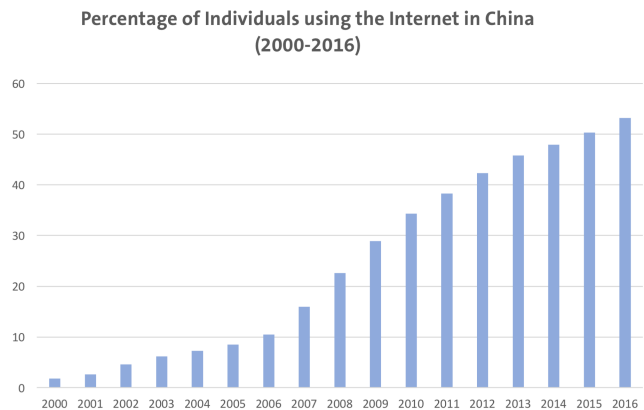
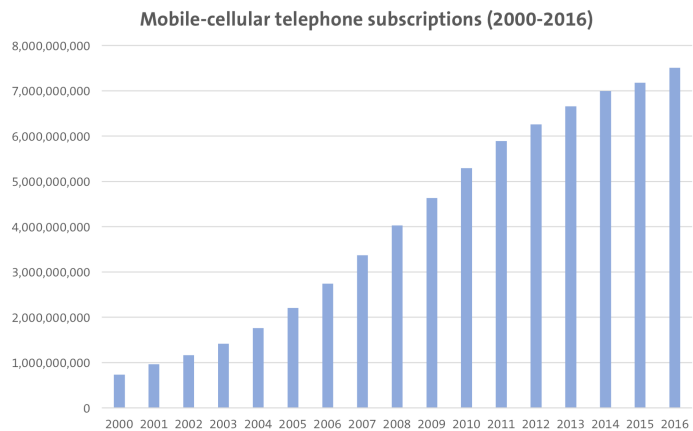
*On August 25, 2000, Internet Wire received a forged email press release seemingly from Emulex Corp., saying that the Emulex CEO had resigned and the company's earnings would be restated. Internet Wire posted the message, without verifying either its origin or contents. Several financial news services and Web sites further distributed the false information, and the stock dropped 61% (from \$113 to \$43) before the hoax was exposed. This was a devastating network attack. Despite its amateurish execution (the perpetrator, trying to make money on the stock movements, was caught in less than 24 hours), \$2.54 billion in market capitalization disappeared, only to reappear hours later.*

Increasing interconnectivity leads to more dynamic design activities, as there are more ways or channels through which a design activity may be influenced. The *force awakens* pattern is often a direct consequence of the impact of design outcomes on external, globally connected context. As illustrated by the OLPC project (see the *force awakens* pattern), a project in a rural area in an undeveloped region is not a mere local project. It is also an action in a segment of an emerging market. The *pandora's box* pattern is a consequence of a chain of unforeseen consequences of design outcomes in interconnected environments. Increasing connectivity also means that design professional can quickly learn from each other. The *conformity* pattern is in part a result of the ability of design professionals to quickly obtain globally accessible knowledge about some popular design resources. And the *design-by-buzzword* pattern may lead to successful design outcomes because of the existence of globally connected environments. Global environments enable more people to interact with design outcomes and find novel applications of them.



### Questions to Consider

- How many people are involved in your design activities?
- How are they connected?
- How many people are interacting with your design outcomes? Via which channels?
- In which geographical zones are these people located?
- How likely is that a change in remote regions will influence your design activities?



**Increased mobile phone and internet usage. Source: International Telecommunications Union.**



## C. Learning and Creativity

**Summary:** *Behaviors and expectations of people are continually changing due to their learning and creativity.*

**Related Patterns:** *co-evolution of problem-solution, the force awakens, solution looking for a problem, design-by-buzzword, you are a reflection of your design, commitment, cherry picking.*

In a design activity, all involved stakeholders are continually learning and discovering new things. Designers learn about new technologies, user wishes, or new design methods, and they may change their designs because of that. Challenging or innovative designs require extensive investigation, experimentation, and iterative improvement based on lessons learned. Fred Brooks even argued that in the design of radically new systems, because of the intensive learning process, designers should plan to throw away the first system that they build. They should then use the gained knowledge to implement the system properly in the next run (Chapter “You won’t get it right the first time anyway!”, in [Brook 1995](#)). Moreover, through design, we can learn lessons that are usually difficult to obtain by other means (e.g. [Obrenovic 2011](#)).

Clients also learn about new technologies and other designs. And if such learning is extensive, it may lead to significant changes in their perceptions and expectations.

Even the design outcomes are nowadays becoming more intelligent, being able to learn about the environment and the user and to show some form of agency and intelligent behavior ([Holmquist 2017](#)). The behavior of such systems is

dependent on the history of its usage and the context. It becomes challenging to anticipate responses of such systems in all situations. Because of learning and creativity designers do not work with “mechanical” systems that react the same to the same “stimulus.” Rather, because of learning and adaptation, such “systems” may act and react or “behave” differently each time they are approached.

Learning is also related to discovery and creativity. Boden (2004) noted that people expect three elements of any creative contributions: novelty, utility, and surprise. And designers are expected to come up with new and surprising solutions. Schön (1983) used the notion of ‘surprise’ in his theory of creative design. For Schön surprise leads to design situation framing and reframing keeps a designer from routine behavior, and drive the originality streak in a design project. However, as a consequence, creativity and surprise, are also sources of dynamic and arbitrary change – leading to additional dynamics of design activities.

In addition to designers’ creativity, users and clients may use design solutions in unexpected ways. As discussed in the *solution looking for a problem* pattern, people may find new, often surprising usages of design tools and methods.

Learning and creativity lead to constant changing of behaviors and expectations of people. This change contributes significantly to the dynamics of design activities, and are one of the main sources behind many patterns of dynamics between elements of design activities. The *co-evolution of problem-solution* pattern is often a consequence of learning and creativity of both designers and their clients (e.g. Dorst & Cross 2001). As designers engage in design activities, they will discover new things about design situations. As a design outcome begins to emerge, new implications may be discovered.

When clients see initial sketches, prototypes, or some version of design outcome, they may discover what they really want, but also learn about new possibilities of used materials and technologies, and as a consequence, they may change their wishes. The *force awakens* pattern is partially a consequence of people outside the design activity learning about the value of the design outcome. This learning may lead them to react in a way that may change our design situation. The *solution looking for a problem* pattern is a consequence of people's creativity in finding new uses for existing solutions.

The *design-by-buzzword* pattern is frequently a consequence of people's curiosity and wish to learn new technologies. Shane Greenstein (2018), for instance, noted that many people enjoy acquiring expertise for its own sake, for the scientific adventure of it, and for the hope of translating that expertise into a great wave. He illustrated this point with an example of a new generation of technological analysts that are educating themselves on the nuances of the blockchain, concluding that *"some of this is about the money, but a lot of it is about enjoying the adventure and becoming expert."*

The *commitment* pattern is closely related to learning, as committing to using some design resources facilitates learning and mastering of its usage. The *conformity* pattern is partially a consequence of design professionals being able to quickly learning how to use the same popular design resource. The *eating its own dog food* pattern is an attempt to stimulate intensive learn within the design company about their designs. And the *you are a reflection of your design* pattern is a consequence of designers and organizations learning and adapting to efficiently produce a particular design outcome.

The *cherry picking* pattern may be viewed as an attempt to avoid risks and costs of extensive learning. In this pattern,

designers know limitations of their resources and are applying them only in situations where the resources have proved their value and no surprises are expected.



### Questions to Consider

- How do you come up with new ideas?
- How do you discipline your creativity?
- How do you stimulate your individual and group creativity?
- How do you prevent being unnecessary creative?



Creative usage of a pen as an “emergency” capo. People often find new and surprising usages of artifacts. Credit: Larry Jacobsen / Wikimedia Commons / CC-BY-2.0.

## D. Tacit Knowledge and Skills

**Summary:** *Designers are relying on tacit knowledge and skills that they do not fully understand and cannot fully explain. Because of its implicit nature, it is hard to predict what effects tacit knowledge will have in design.*

**Related Patterns:** *co-evolution of problem-solution, commitment.*

The term “tacit knowing” or “tacit knowledge” was first introduced by Michael Polanyi, with his central assertion being that “*we can know more than we can tell.*” (Polanyi 1966, page 4). Polanyi analyzed diverse forms of tacit knowledge, such as tradition, inherited practices, implied values, and prejudices.

Design is closely related to tacit knowledge. Design aims at exploiting implicit, tacit knowledge and skills of designs and other people involved in the design activity. Schön (1987) calls such knowledge *knowing-in-action*, revealed only in the way we carry out tasks and approach problems (page 25). Schön elaborated that such knowledge can only be revealed by the skillful execution of the performance. We are characteristically unable to make it verbally explicit. In other words, while we cannot explain such knowledge and skills, we can demonstrate them by being engaged in a particular activity. Cross similarly refers to “natural intelligence” of design ability (Cross 1999), and “designerly ways of knowing” (Cross 2006).

A design activity can set in motion our intuitive and tacit knowledge accumulated through years of research and experience. And by engaging users in design, we may employ their intuitive knowledge about their domains. Much of such

valuable knowledge is not captured in existing theories and guidelines. Glass, for example, noted that actions of designers are often implicit and intuitive. He defined intuition as: “*a function of our mind that allows it to access a rich fund of historically gleaned information we are not necessarily aware we possess, by a method we do not understand*” (Glass, 2006; page 125). Glass elaborated that our unawareness of such knowledge does not mean that we cannot use it.

Intuition, judgment and tacit skills play a critical role in understanding and setting problems from messy and ill-defined situations (Obrenovic 2011). Tacit knowledge is often seen as one of the most important characteristics of designers. Moreover, we usually would not call some activity *design* if it does not involve tacit skills (Weth 1999). Tacit knowledge and intuition are often only available intellectual ‘tools’ by which designers can handle real-world complexity.

Tacit knowledge is crucial in design, but because of its implicit nature, it is hard to predict what effects it will have. Consequently, tacit knowledge has a significant influence on the dynamics of design activities. The *co-evolution of problem-solution* pattern is often a consequence of actions based on intuition, judgment, and tacit skills. For instance, tacit skills are crucial to identifying design problems, especially in the realm of complex human and social issues. The similar applies to the *commitment* pattern as mastering usage of tools often requires acquiring an extensive set of tacit knowledge and skills.



## Questions to Consider

- What kind of tacit knowledge and skills you use in your design activities?
- How are you developing these skills?
- How do you combine tacit (implicit) and explicit knowledge?
- How do you capture and share tacit knowledge?
- How do you plan activities that involve extensive usage of tacit skills?



A designer sketching. Skills relevant to design require tacit knowledge, not always known explicitly, even by expert practitioners, and which is difficult or impossible to explicitly transfer to other people. Credit: Hrvoje Abraham Miličević / Pexels.

## E. Social Dynamics

**Summary:** *Design situations, resources, and outcomes are complex socio-technical systems, with complex, difficult to predict dynamics.*

**Related Patterns:** *the force awakens, design-by-buzzword, conformity, commitment, your design is your reflection.*

More than 80 years ago, the Hawthorne studies demonstrated that no one interested in the behavior of people could consider them as isolated individuals (Obrenovic 2014). These studies initially aimed at studying the impact of material conditions (e.g., lighting) on the productivity of workers. They discovered that it makes no sense to study the productivity of workers without taking into account factors such as motivational and group influences, social status, informal communication, roles, norms, job satisfaction, resistance to change, group norms, worker participation, and leadership.

Everything that designers do relates to people and their social relations. Every design activity is a field of complex social dynamics. Social dynamics manifests itself through multiple mechanisms, such as culture, values, norms, ethics, peer pressure, charisma, honor, prestige, and so on. Social dynamics affect design activates at all levels. Design situations are complex social contexts, where people and their connection are of primary importance. Design outcomes are never mere material products but always change social dynamics. Sometimes changing the social dynamics may be the primary goal of design activity, such as when we are designing new processes or methods of works in organizations. And design resources are not mere material things, but they come with associated communities of producers, marketers, distributors,



practitioners, and education institutions.

Understanding the social context of design is crucial to its success. There are many examples of products that failed due to lack of understanding of the social context. The success of some product in one country or culture does not guarantee its success in another country or culture. Aaron Marcus, for example, discussed that there are significant differences between Western and Asian ways of communicating and interacting and that these differences pervade all aspects of how each group understands daily life and how each group behaves in daily activities (Marcus 2003).

Clients perception of values and culture of companies and their leaders can have a significant impact on adoption or rejection of designed products. The success of Apple products, for instance, is often associated with the charisma and legacy of Steve Jobs. But there are also negative examples. In 2014, for example, some users boycotted the Firefox Web browser because of the CEO's stance on gay rights. This boycott significantly contributed to the pressure that led to the CEO's resignation (Obrenovic 2015). Similarly, in 2017 The Uber CEO stepped down, due to the pressure caused by negative publicity, such as the #DeleteUber campaign and the reported incidents of sexual harassment<sup>41</sup>.

Design resources also have their social dynamics. Producers of design resources may use different forms of social activates and pressure to promote usage of their products. Organizing social events, such as sponsored conferences and parties, are all parts of carefully planned promotional activities. Each design resource also has a community behind it, with its own culture. Open-source projects, for instance, have an

---

<sup>41</sup><https://www.susanjfowler.com/blog/2017/2/19/reflecting-on-one-very-strange-year-at-uber>

elaborate and sophisticated network of people working and involved in them (Bach & Twidale 2010). These communities also organize various social events, such as conferences or “meetups,” and promote ideas, values and best practices through social networks, blog posts, journals, and books. Such communities also have their thought leaders and influencers. Recommendations of some of the famous and charismatic thought leader may have a stronger impact on design decision than any other factors.

The broader community of the discipline in which designers operate may have a significant impact on their design decisions. Such communities promote specific values and norms, and define what is ethical, acceptable and not acceptable (e.g. Goodman 2014, Williamson & Sundén 2016).

The structure of design outcomes is also related to social dynamics of teams creating these outcomes. As illustrates by Conway’s law (Conway 1968), communication structure of an organization may be reflected in a designed system. And there are a number of reported examples of “technical” problem, that in the end turned out to be a consequence of social issues (e.g. Matsudaira 2016, Kromhout 2018).

Social factors influence most of the patterns of dynamics of design activities. The *force awakens* pattern is often a consequence of complex social dynamics in society. The *design-by-buzzword* pattern is in many ways a consequence of social dynamics related to the adoption of new technologies. Popularity and peer pressure may be a strong motivator to buy some new product. And some users see it as a matter of prestige to be among first adopters of some new product. The *conformity* pattern may be viewed as a consequence of workings of social mechanisms, such as peer-pressure in the community of designers, popularity and fashion of some de-

sign resources. The *commitment* pattern may be influenced by social dynamics related to promotion of some design resource. And the *your design is your reflection* pattern, particularly at the organizational level (e.g., Conway's law), is related to social dynamics within design organizations and teams.



### Questions to Consider

- What kind of culture your organization has? What kind of cultures your customers or users have?
- While designing, how do you take into account cultural differences?
- What kind of political issues you face in your design activities?
- Are you a part of a broader professional community?
- What type of ethical consideration do you meet in design activities?



Floating market boats in Thailand. Designers should note consider people as isolated individuals. Understanding the social context of design is crucial to its success. Credit: Pxhere.

## F. Lack of Strong Principles, Theories, and Laws

**Summary:** *Designers are dealing with poorly understood phenomena, for which we do not have strong underlying principles, theories, and laws.*

**Related Patterns:** *the force awakens, pandora's box, co-evolution of problem-solution, solution looking for a problem, design-by-buzzword.*

One common theme about previous sources of design dynamics is that we are dealing with poorly understood phenomena. Even though there are large numbers of “theories” today about human factors, design processes, and tools, we do not have “strong” theories in the sense that they can be used to prescribe or predict design fully. Learning and creativity, for example, depend on people’s intelligence, previous knowledge and experience, motivation, context, and other factors. Even when studied in more controlled settings, such as in the classroom, learning and creativity are difficult to model. In real-world situations, it is even harder to anticipate how fast and what people will learn, how creative they will be, and how this will affect their behavior. For example, when introducing a new device, we require people to learn how to use it. This learning process is a function of device design, but other factors, such as peer pressure (e.g., number of friends who have a similar device), can have even more impact on motivation and learning. We do not have enough knowledge to predict or simulate such processes reliably.

Brooks similarly argued that most of the design complexity is arbitrary complexity, not coming from stable underlying

principles, but forced, often without apparent reason, by the human institutions and systems to which we must conform (Brooks 1987):

*“We are not alone in facing complexity. Physics deals with terribly complex objects even at the ‘fundamental’ particle level. The physicist labors on, however, in a firm faith that there are unifying principles to be found, whether in quarks or in unified field theories. Einstein repeatedly argued that there must be simplified explanations of nature ... No such faith comforts the software engineer. Much of the complexity he must master is arbitrary complexity, forced without rhyme or reason by the many human institutions and systems to which his interfaces must conform. These differ from interface to interface, and from time to time, not because of necessity but only because they were designed by different people...”*

Lack of robust theories and underlying principles means that it is not possible to reliably model or predict effects of any design decisions. Designers need to be flexible and to investigate continuously, experiment, and iteratively refine their designs.

The lack of strong theories contributes significantly to the dynamics of design activities. The *force awakens* and *pan-dora’s box* patterns, for instance, are partially a consequence of our inability to estimate the impact of our design efforts in a broader context. The *co-evolution of problem-solution* pattern partially happens due to lack of knowledge and theories about a particular design situation and our inability to develop sufficient understanding of the design situation beforehand.

On the other hand, the *solution looking for a problem* and *design-by-buzzword* patterns, practically assume that our current understanding and predicted usages of some technology

are limited, and that new, unpredicted usages will emerge.



### **Questions to Consider**

- Do you use any theories or models to analyze, predict or simulate elements of design activity?
- How reliable are such models and theories?
- How do you approach poorly understood situations and problems?



Albert Einstein, Paul Ehrenfest, Paul Langevin, Heike Kamerlingh Onnes, and Pierre Weiss at Onnes's home in Leiden, the Netherlands (1920). Einstein repeatedly argued that there must be simplified explanations of nature. Credit: Wikimedia Commons.



# Discussion and Conclusions

Dynamics between elements of design activities lead to complexity characteristic for design activities. While design situations, outcomes and resources are themselves complex structures, what makes a design activity additionally and characteristically complex are interactions among these structures. As illustrated with our patterns of dynamics, these interactions challenge designers' ability to deal with complexity as any of these structures is subject to change, in difficult to predict ways.

One of the central assumptions in this book is that design complexity, as a consequence of dynamics of design activities, *should not be seen as a problem*. We are not alone to make this argument. Fred Brooks (1987), for instance, claimed that descriptions of a (software) design that abstract away its complexity often abstracts away its essence. Consequently, we view the dynamics and unpredictability of a design process as *a defining characteristic of any design activity*. Moreover, we believe that a lack of stability is a requirement for any process to be called design. If a process is more predictable and stable, we would typically not see it as a design activity. Or, at best, we would call it a routine design.

In many ways, unpredictable dynamics of design activities is exactly the state that designers want and need. A fast-changing world, while being complex to understand and work with, is a rich source of design opportunities. Designers are

expected to create new, surprising and creative outcomes in domains where conventional, more structured approaches may not work. They are supposed to dive into new areas, to push the limits, rather than operate in standardized and well-understood domains. This expectation, by necessity, leads to an unstable process, in the sense that it is not possible to fully prescribe or control it.

The key question is not how to make design activities stable and entirely predictable, but how to deal with the resulting complexity. In general, we can deal with complexity in two fundamental ways: develop tools to reduce it when we can, or learn to tolerate it when we cannot ([Rubinstein 1986](#)). Both of these ways are relevant for designers. The design research community has been working on developing tools to deal with different forms of design complexity (e.g. [Stolterman 2008](#), [Montagne 2010](#)). In software design, for instance, lots of attention has been given to tools that can help developers to avoid creating unnecessary complex system implementations ([Brooks 1995](#)). However, less attention has been given to acceptance and tolerance of design complexity itself. Robert Glass ([2006](#)) claimed that it is unusual to find tolerance for complexity among managers, entrepreneurs, and maybe more surprisingly academics (page 104). The majority seems to be looking for simple and elegant solutions. Glass, however, argued, as we do, that we should accept complexity and ambiguity in design, rather than trying to wish them away.

Accepting design complexity does not mean ignoring it. On the contrary. Accepting complexity has to be combined with more awareness about it. There is a strong need to be more considerate about design complexity and its sources, and invest more time in understanding and coping with it. With this book, we want to increase awareness about complexity

and its origins in design. Furthermore, we want to stimulate designers and design researchers to approach the issues of design complexity more thoughtfully.

Our patterns of dynamics also highlight the need for holistic approach to designing. Many forces shape and influence design activities. Without a pro-active approach to manage and balance impacts of these forces, we may end up in design activities that are driven by hype, fashion, and politics rather than by the value delivered to users.

In addition to increasing the awareness, we believe that our framework and patterns can give some insights about why particular design approaches are more (or less) successful in dealing with design complexity. For example, most designers are well aware that due to the highly-interdependent nature of design situations, outcomes, and resources, it is not possible to work on one part of the design in isolation. For instance, two of the most recognized design scholars [Lawson \(2005\)](#) and [Schön \(1990\)](#) argued that design problems are rarely simple problems with only one or two features, but that a whole host of criteria must be satisfied and a multitude of constraints respected. Lawson concludes that *“the only way to keep them all in mind at once, as it were, is to oscillate very quickly between them like a juggler.”* (page 151). Lawson also cites Michael Wilford who compares architects to jugglers: *“[a juggler] who’s got six balls in the air . . . and an architect is similarly operating on at least six fronts simultaneously, and if you take your eye off one of them and drop it, you’re in trouble’.”* (page 151)



Juggling (combined juggling and jogging). Designers are often compared to jugglers operating on several fronts simultaneously. Credit: Owen Morse / Wikimedia Commons.

Experienced designers already understand that the “conventional” ways of dealing with complexity cannot always be applied in design. For instance, one of the most common approaches to deal with complexity is a separation of concerns often called a divide-and-conquer approach. In this approach, complex objects are divided into smaller more manageable pieces. Each of the resulting less complex pieces can then be handled more or less independently of other pieces. Hierarchical organizations of systems, in general, follow this approach. In design, however, it is normally not possible to easily decompose problems into smaller chunks that can be solved in isolation. Advanced designers are aware that it is normally not possible to be a reductionist. In design, it is usually not possible to reduce complexity by splitting the problem into smaller chunks that designers can more easily handle. Frosh nicely illustrates this issue with his view on systems engineering and systems analysis (Frosh 1969):

*“One of the key misassumptions in modern systems engineering and systems analysis is that the total problem can, and frequently is, decomposed into sub-problems; the sub-problems can be solved more or less independently, and the total solution can be synthesized by combination of the sub-solutions, treating the interactions of the parts as ‘interfaces.’ The real world is, however, highly nonlinear, and unless real attention is paid to this fact, the linear decomposition treatment will fail catastrophically, because the interaction terms may be as large as the sub-problems and not reducible to simple interfaces. The result may well remain decomposed.”*

Similarly, Rick Nason, in his discussion on complexity in business, noted that complex problems involve too many unknowns and too many interrelated factors to reduce to rules and processes (Nason 2017). Hendler et al. (2008) also noted

that complex systems have emergent properties that are not predictable by analyzing micro technical or social effects.

Design complexity, thus, needs to be addressed with other approaches than decomposition and reductionism (e.g. [Somerville et al. 2012](#); [Stolterman 2008](#)).

Experienced designers are aware that a designer never has enough information to predict any aspect of a design reliably, and that it is therefore not always a good idea to spend much time analyzing and theorizing about solutions and potential consequences. Instead, attempting a solution, making mistakes and learning from them is one of the most important strategies that any designer use (e.g. [Ball et al. 2010](#); [Gaver et al., 2009](#); [Buxton 2007](#); [Lamer 1989](#); [Linden 2010](#)). Henry Petroski, for instance, argues that the old dictum “form follows function” is false and that the basic rule of design evolution is “*form follows failure*” ([Petroski 1994](#): p 22). He argued that changes in the design, even for commonplace things like forks and paper clips, are more motivated by the things early designs do poorly than those things they do well. Such flaws guide design decisions, as each new design attempt is an attempt to correct flaws. In the domain of software design, empirical studies similarly showed that good software designers use a trial-and-error process. The purpose of this process is to create mental models of a design solution, while poor designers tend to seize on a solution with fewer trials and errors ([Curtis 1987](#); [Adelson 1986](#)). Or, as noted by Sir Francis Bacon almost 400 years ago “*truth will sooner come out of error than from confusion*” ([Bacon 1620](#)).

Probably the model that most closely describes the challenges that designers face in dealing with complexity is the famous [Simon’s \(1996\)](#) model of “bounded rationality.” It is the concept that decision makers (irrespective of their level of intelli-

gence) have to work under three unavoidable constraints: (1) only limited, often unreliable, information is available regarding possible alternatives and their consequences, (2) human mind has only limited capacity to evaluate and process the information that is available, and (3) only a limited amount of time is available to make a decision. Even individuals who intend to make rational choices are bound to make satisficing (rather than maximizing or optimizing) choices in complex situations. Decision-makers in this view are satisficing because they seek a satisfactory solution rather than an optimal one.

The bounded rationality model reflects very well the designers' work. Designers are constantly making a complex design decision, in fast-changing environments. The world is changing fast, and designers need to make decisions before they fully understand design situations to avoid creating obsolete outcomes. And increased interconnectivity makes such changes quickly propagate in many parts of the world. Lately, there has been some advancements based on Simon's ideas, such as the work of Hatchuel (2001). Hatchuel introduced the model that emphasizes, even more, the need to handle complexity as a designer, arguing that it is not possible to deal with complexity by being comprehensive. Instead, he argues for an explorative and judgment based approach.

We also see a strong movement from prediction based design management toward management that adapts to change. Initiatives, such as rapid prototyping and agile software development all emphasize the importance of constant adaptation to change. Ries (2011) noted that old management methods, such as long-term planning and forecasting, are not appropriate in dynamic contexts. As suggested by Hales (1985) designers need to be like chameleons, constantly adapting to dynamic

environments.

Complexity in design means different things to someone who is a researcher and someone who is a practicing designer. Most researchers focus on one or a few of the patterns and their relationships. However, a practicing designer does not have such a choice. When engaging in real design activity, a designer cannot decide to only pay attention to particular patterns or relationships. A designer has to deal with the full complexity. The recognition that most design research only pay attention to some specific aspects of the complexity of design practice was one of the motivations for creating a framework that would make it possible to examine and discuss design complexity without losing the overall view.

We believe that it is crucial to educate designers to be more tolerant and thoughtful about design dynamics and complexity. We can learn from successful designers who have developed a range of approaches suitable for dealing with design complexity. We hope that our book can facilitate such knowledge transfer by providing the structure and vocabulary to describe and correlate complex design experiences.



# Bibliography

- Adelson B., Soloway E. (1986):  
**A model of software design,**  
*International Journal of Intelligent Systems* 1 (3), pp. 195-213.
- Aste T., Tasca P., Di Matteo T. (2017):  
**Blockchain Technologies: The Foreseeable Impact on Society and Industry,**  
*IEEE Computer* 50(9), pp. 18-28.
- Bach P.M and Twidale M. (2010):  
**Social participation in open source: what it means for designers,**  
*interactions* 17 (3), pp. 70-74.
- Bacon, Francos (1620):  
***The New Organon.***
- Ball L.J., Onarheim B., Bo T. Christensen, (2010):  
**Design requirements, epistemic uncertainty and solution development strategies in software design,**  
*Design Studies* 31 (6), pp. 567-589.
- Bell A.E. (2004):  
**Death by UML Fever,**  
*Queue* 2 (1), pp. 72-80.
- Berghel H. (1999):  
**Digital Village: How Xday Figures in the Y2K Countdown,**  
*Commun. ACM* 42 (5), pp. 11-15.
- Boden M. A., (2004):  
***The Creative Mind: Myths and Mechanisms,***

Psychology Press, 2nd edition, expanded/revised.

Bass L., Clements P.C., and Kazman R. (2012):  
***Software Architecture in Practice***,  
(3rd Edition) Addison-Wesley Professional.

Boehm B. and Bhuta J. (2008):  
**Balancing Opportunities and Risks in Component-Based  
Software Development**,  
*IEEE Software* 25 (6), pp. 56-63.

Booch G. (2008):  
**Measuring Architectural Complexity**,  
*IEEE Software* 25 (4), pp. 56-63.

Brooks F.P. (1987):  
**No Silver Bullet—Essence and Accidents of Software En-  
gineering**,  
*Computer* 20 (4), pp. 10-19.

Brooks F.P. (1995):  
***The Mythical Man-Month***,  
Addison-Wesley Professional; 2nd edition.

Brooks F.P. (2010):  
***The Design of Design***,  
Addison-Wesley Professional.

Brown W.J., Malveau R.C., McCormick H.W., and Mowbray  
T.J. (1998):  
***AntiPatterns: Refactoring Software, Architectures, and  
Projects in Crisis***,  
Wiley.

Brown G., Fisher M., Stoll N., Beeksmma D., Black M., Taylor R.,  
Seok Yon C., Williams A.J., Bryant W., and Jansen B.J. (2000):  
**Using the lessons of Y2K to improve information systems**

**architecture,**

*Commun. ACM* 43 (10), pp. 90-97.

Buchanan R. (1992):

**Wicked Problem in Design Thinking,**

*Design Issues* 8 (2), pp. 5-21.

Buxton B. (2007):

***Sketching User Experiences: Getting the Design Right and the Right Design,***

Morgan Kaufmann Publishers Inc., San Francisco, CA.

Chan C.-S. (2001):

**An examination of the forces that generate a style,**

*Design Studies* 22 (4), pp. 319-346.

Charette R. N. (2005):

**Why Software Fails,**

*IEEE Spectr.* 42 (9), pp. 42-49.

Chua J. (2015):

**Design Without Final Goals: Getting Around Our Bounded Rationality,**

*Artifact* 3 (4), pp. 2.1-2.7.

Cockburn A. (2000):

**Selecting a Project's Methodology,**

*IEEE Softw.* 17 (4), pp. 64-71.

Colwell, B. (2005):

**Complexity in Design,**

*Computer* 38 (10), pp. 10-12.

Conway, M.E. (1968):

**How do Committees Invent?,**

*Datamation* 14 (5), pp. 28-31.

Coplien J.O (1999):

**Guest Editor's Introduction: Reevaluating the Architectural Metaphor-Toward Piecemeal Growth,**

*IEEE Software* 19 (5), pp. 40-44.

Coyne R. (2005):

**Wicked problems revisited,**

*Design Studies* 26 (1), pp. 5-17.

Cross N. (1999):

**Natural intelligence in design,**

*Design Studies* 20 (1), pp. 25-39.

Cross N. (2006):

***Designerly ways of knowing,***

Springer Science & Business Media.

Curtis B., Guindon R., Krasner H., Walz D., Elam J., and Iscoe N., (1987):

**Empirical Studies of the Design Process: Papers for the Second Workshop on Empirical Studies of Programmers,**  
*MCC Technical Report Number STP-260-87.*

Ditlea, Steve (1981-10-01):

**An Apple On Every Desk.**

*Inc*<sup>42</sup>, Retrieved April 22, 12013.

Dorst, C.H., Cross, N.G. (2001):

**Creativity in the design process: co-evolution of problem-solution,**

*Design Studies* 22(5), pp. 425-437.

Ebert C., Jones C. (2009):

**Embedded Software: Facts, Figures, and Future,**

*Computer* 42 (4), pp. 42-52.

Farrell R., Hooker C. (2013):

---

<sup>42</sup><http://www.inc.com/magazine/19811001/2033.html>

**Design, science and wicked problems,**

*Design Studies* 34 (6), pp. 681-705.

Fielding, R.T. (2000):

***Architectural Styles and the Design of Network-based Software Architectures.***

Dissertation. University of California, Irvine.

Fonseca O., Fazzion E., Cunha I., Las-Casas P.H.B., Guedes D., Meira W., Hoepers C., Steding-Jessen K., Chaves M.H.P. (2016):

**Measuring, Characterizing, and Avoiding Spam Traffic Costs,**

*IEEE Internet Computing* 20 (4), pp. 16-24.

Frosh, R.A. (1969):

**A new look at systems engineering,**

*IEEE Spectrum*, September 1969, pp. 24-28.

Funke J. (1991):

**Solving complex problems: Exploration and control of complex systems,**

*Complex problem solving: Principles and mechanisms.*

Gaver W., Bowers J., Kerridge T., Boucher A., and Jarvis N.. (2009):

**Anatomy of a failure: how we knew when our design went wrong, and what we learned from it,**

In *Proceedings of the 27th international conference on Human factors in computing systems (CHI '09)*,

ACM, New York, NY, USA, 2213-2222.

Gero J.S. (1990):

**Design prototypes: a knowledge representation schema for design,**

*AI Magazine* 11 (4), pp. 26–36.

Glass R. L. (2006):  
***Software Creativity 2.0***,  
developer.\* Books.

Gokpinar B., Hopp W.J., Iravani S.M.R (2010):  
**The Impact of Misalignment of Organizational Structure  
and Product Architecture on Quality in Complex Product  
Development**,  
*Management Science* 56 (3), pp. 468-484.

Goodman E. (2014):  
**Design and ethics in the era of big data**,  
*interactions* 21 (3), pp. 22-24.

Goth G. (2011):  
**IBM PC Retrospective: There Was Enough Right to Make  
It Work**,  
*Computer* 44 (8), pp. 26-33.

Gravemeijer K. and Cobb P. (2006):  
**Design research from a learning design perspective**,  
in *Akker et al. (Eds.) Educational Design Research*.

Greenberg S. and Buxton B. (2008):  
**Usability evaluation considered harmful (some of the time)**,  
in *Proceedings of the twenty-sixth annual SIGCHI conference  
on Human factors in computing systems (CHI '08)*,  
ACM, New York, NY, USA, 111-120.

Greenstein S. (2018):  
**The Paradox of Technological Déjà Vu**,  
*IEEE Micro* 38 (1), pp 118-120.

Gubbi J., Buyya R., Marusic S., Palaniswami M. (2013):  
**Internet of Things (IoT): A vision, architectural elements**,

**and future directions,**

*Future Generation Computer Systems* 29 (7), pp. 1645-1660.

Hadjerrouit S. (1998):

**Java as first programming language: a critical evaluation,**  
*SIGCSE Bull.* 30 (2), pp. 43-47.

Hales C. (1985):

**Designer as chameleon,**  
*Design Studies* 6 (2), pp. 111-114.

Harrison W. (2004):

**From the Editor: Best Practices—Who Says?,**  
*IEEE Software* 21 (1), pp. 8-11.

Harrison W. (2006):

**Eating Your Own Dog Food,**  
*IEEE Software* 23 (3), pp. 5-7.

Hatchuel, A. (2001):

**Towards Design Theory and Expandable Rationality: The Unfinished Program of Herbert Simon,**  
*Journal of Management and Governance* 5 (3-4), pp. 260-273.

Heiko G.A. (2004):

**Announcement, entry, and preemption when consumers have switching costs. (econometric analysis),**  
*RAND Journal of Economics (The RAND Corporation)* 35 (1), pp. 184-202.

Hendler J., Shadbolt N., Wendy Hall, Tim Berners-Lee, and Daniel Weitzner (2008):

**Web science: an interdisciplinary approach to understanding the web,**  
*Commun. ACM* 51 (7), (July 2008), 60-69.

Holmquist L. E. (2017):

**Intelligence on tap: artificial intelligence as a new design**

**material,**

*interactions* 24, 4, pp. 28-33.

Janlert L.E. and Stolterman E. (2008):

**Complex interaction,**

*ACM Trans. Comput.-Hum. Interact.* 17 (2), Article 8, 32 pages.

Kölling, M. (1999):

**The Problem of Teaching Object-Oriented Programming,  
Part 1: Languages,**

*Journal of Object-Oriented Programming* 11 (8), pp. 8-15.

Komssi M., Pichlis D., Raatikainen M., Kindstram K., Jarvinen J. (2015):

**What are Hackathons for?**

*IEEE Software* 2015 (5), pp. 60-67.

Kraemer K.L., Dedrick J., and Sharma P. (2009):

**One laptop per child: vision vs. reality,**

*Commun. ACM* 52 (6), pp. 66-73.

Kromhout B. (2018):

**Containers Will Not Fix Your Broken Culture (and Other  
Hard Truths),**

*ACM queue* 16 (6).

Lamers S. (1989):

***Programmers at Work: Interviews with 19 Programmers  
Who Shaped the Computer Industry,***

Tempus Books.

Lampson B.W. (1984):

***Hints for Computer System Design,***

*IEEE Software* vol. 1 (1), pp. 11-28.

Lawson B. (2005):

***How Designers Think,***



Architectural Press, 4th edition.

Lewis J., Fowler M. (2014):

**Microservices: a definition of this new architectural term,**  
<https://martinfowler.com/articles/microservices.html>

Linden J. Ball, Balder Onarheim, Bo T. Christensen (2010):  
**Design requirements, epistemic uncertainty and solution development strategies in software design,**  
*Design Studies* 31 (6), pp. 567-589.

MacCormack A., Carliss Y.B., and Rusnak J. (2012):  
**Exploring the Duality Between Product and Organizational Architectures: A Test of the ‘Mirroring’ Hypothesis,**  
*Research Policy* 41 (8), pp. 1309–1324.

Matsudaira K. (2016):  
**Bad Software Architecture is a People Problem,**  
*Communications of the ACM* 59 (9), pp. 42-43.

Melnik G., Jeffries R.,  
**Guest Editors’ Introduction: TDD–The Art of Fearless Programming,**  
*IEEE Software* 24(3), pp. 24-30.

Mens T. (2012):  
**On the Complexity of Software Systems,**  
*Computer* 45 (8), pp. 79-81.

Montagne K. (2010):  
**Tackling Architectural Complexity with Modeling,**  
*Communication of the ACM* 53 (10), pp. 46-52.

Marcus A. (2003):  
**User-interface design and China: a great leap forward,**  
*interactions* 10 (1), pp. 21-25.

Murray, P. (2004). T

***The Saga of the Sydney Opera House,***

London: Spon Press.

Myers B., Scott E. Hudson, & Randy Pausch. (2000):

**Past, present, and future of user interface software tools,**  
*ACM Trans. Comput.-Hum. Interact.* 7 (1), pp. 3-28.

Nason R. (2017):

***It's Not Complicated: The Art and Science of Complexity in Business,***

Rotman-UTP Publishing, 2017).

Ncube C., Oberndorf P. and Kark A.W. (2008):

**Opportunistic Software Systems Development: Making Systems from What's Available,**  
*IEEE Software* 25 (6), pp. 38-41.

Neumann P.G. and McCullagh D. (1999):

**Inside Risks: Risks of Y2K,**  
*Commun. ACM* 42 (6), p. 144.

Nelson, H., Stolterman, E. (2012):

***The Design Way—Intentional Change in an Unpredictable World,***

MIT Press.

Newman S. (2015):

***Building Microservices: Designing Fine-Grained Systems,***  
O'Reilly Media; 1st edition.

Nielsen, J. (1993):

**Iterative User Interface Design,**  
*IEEE Computer* 26 (11), pp. 32-41

Nielsen, J. (2010):

**iPad Usability: First Findings From User Testing,**  
Jakob Nielsen's Alertbox, April 26, 2010.

<http://www.useit.com/alertbox/ipad.html>

Norman D.A. (2008):

***Workarounds and hacks: the leading edge of innovation,***

*interactions* 15 (4), pp. 47-48.

Norman, D.A. (2010):

**Technology first, needs last: the research-product gulf,**

*interactions* 17 (2), pp. 38-42.

Nye, D. E. (2006):

***Technology matters: questions to live with,***

Cambridge, MA: MIT Press.

Obrenovic Z. (2011):

**Design-Based Research: What We Learn When We Engage in Design of Interactive Systems,**

*interactions* 18 (5), pp. 56-59.

Obrenovic Z. (2013):

**Software Sketchifying – Bringing Innovation into Software Development,**

*IEEE Software* 30 (3), pp. 80-86.

Obrenovic Z. (2014):

**The Hawthorne Studies and Their Relevance to HCI Research,**

*interactions* 21 (6), pp. 46-51.

Obrenovic Z. (2015):

**Design as a Political Activity: Borrowing From Classical Political Theories,**

*interactions* 22 (6), pp. 56-59.

Obrenovic Z. (2017):

**Insights from the Past: The IEEE Software History Exper-**

**iment,**

*IEEE Software* 34 (4), pp. 71-78.

Penzenstadler B., Raturi A., Richardson D., and Tomlinson B.  
(2014):

**Safety, Security, Now Sustainability: The Nonfunctional  
Requirement for the 21st Century,**

*IEEE Software* 31 (3), pp. 40-47.

Petroski H. (1994):

***The Evolution Of Useful Things,***

Vintage; Reprint edition

Plauger P.J. (1992):

**The Falutin' Index,**

*Embedded Systems Programming*, May 1992, pp. 88-92.

Pescio C. (1997):

**When past solutions cause future problems [Year 2000  
problem],**

*IEEE Software* 14 (5), pp. 19-21.

Polanyi M. (1966):

***The Tacit Dimension,***

University of Chicago Press: Chicago.

Reitman, W. (1965):

***Cognition and Thought.***

New York: Wiley.

Richard H.S. (1995):

**Microsoft and vaporware,**

*IEEE Micro Magazine* 15 (2), pp. 6-7.

Richardson E. (2011):

**What an Agile Architect Can Learn from a Hurricane  
Meteorologist,**

*IEEE Software* 28 (6), pp. 9-12

Ries A. and Ries L. (2002):  
***The 22 Immutable Laws of Branding***,  
HarperBusiness; 1st edition.

Ries E. (2011):  
***The Lean Startup***,  
Crown Business.

Rising L. (2010):  
**The Benefit of Patterns**,  
*IEEE Software* 2010 (5), pp. 15-17.

Rittel, H., and Webber M. (1973):  
**Dilemmas in a General Theory of Planning**,  
*Policy Sciences* 4, pp. 155–169.

Rubinstein M.F. (1986):  
***Tools for Thinking and Problem Solving***,  
Prentice-Hall, Incorporated.

Shneiderman B. (2007):  
**Creativity support tools: accelerating discovery and innovation**,  
*Commun. ACM* 50 (12), pp. 20-32.

Schön, D. (1983):  
***The Reflective Practitioner***.  
Basic Books, New York.

Schön, D. (1987):  
***Educating the Reflective Practitioner***,  
San Francisco: Jossey-Bass.

Schön D. (1990):  
**The Design Process**,  
Howard V.A. (1990) Ed. *Varieties of Thinking: Essays from*

*Harvard's Philosophy of Education Research Center*. New York, Routledge, pp. 110-141.

Schön D., Wiggins G. (1992):

**Kinds of seeing and their functions in designing,**  
*Design Studies* 13 (2), pp. 135-156.

Simon H. (1996):

***The Sciences of the Artificial*,**  
3rd ed. MIT Press.

Herbert A. Simon (1973):

**The structure of ill structured problems,**  
*Artificial Intelligence* 4 (3), pp. 181-201.

Sommerville I., Cliff D., Calinescu R., Keen J., Kelly T., Kwiatkowska M., Mcdermid J., and Paige R. (2012):

**Large-scale complex IT systems,**  
*Commun. ACM* 55 (7), pp. 71-77.

Stolterman E. (2008):

**The Nature of Design Practice and Implications for Interaction Design Research,**  
*International Journal of Design (IJDesign)* 2 (1).

Stolterman, E., Mcatee, J., Royer, D. and Thandapani, Selvan (2008b):

**Designerly Tools,**  
*Undisciplined!*, Design Research Society Conference 2008, Sheffield Hallam University, Sheffield, UK, 16-19 July 2008.

Stolterman, E. & Pierce, J. (2012):

**Design tools in practice: Studying the designer-tool relationship in interaction design,**  
*DIS 2012*, June 11-15, 2012, Newcastle, UK

Weth, von der R. (1999):

**Design instinct?—the development of individual strate-**

**gies,**  
*Design Studies* 20 (5), pp. 453-463.

Wilde E. and Glushko R.J. 2008:  
**XML fever,**  
*Commun. ACM* 51 (7), pp. 40-46.

Williamson J.R. and Sundén D. (2016):  
**Deep cover HCI: the ethics of covert research,**  
*interactions* 23 (3), pp. 45-49.

# Appendixes

- Appendix A. Patterns of Dynamics: Summaries
- Appendix B. Patterns of Dynamics: Mottos
- Appendix C. Sources of Dynamics: Summaries
- Appendix D. Questions to Consider



## A. Patterns of Dynamics: Summaries

**Co-Evolution of Problem-Solution:** *A design situation and designer's understanding of the design situation (problem definition) changes and evolves in parallel and under the influence of design outcome and designer's understating of potential design outcomes (design solutions).*

**Puzzle Solving:** *A design situation is viewed as a clearly defined and static problem (i.e., a problem that can be clearly stated and where it is known what form the solution should have). A design outcome is seen as a solution for this problem. Design is viewed as a problem-solving activity.*

**A Solution Looking for a Problem:** *Understanding of a design outcome and its possibilities leads to innovative usages of the design outcome in situations and for problems that were not initially envisioned.*

**Pandora's Box:** *The consequences of a design outcome create new situations that are perceived as problematic and may ask for another design to improve it. A design outcome may become more known for the problems it caused than for the solution it provided.*

**The Force Awakens:** *Designer's efforts to address some situation triggers reactions that radically change the original design situation. Often such reactions make the intended design outcome or a contribution of a designer obsolete or irrelevant. Such responses would typically not occur without the design activity.*

**Your Design Is Your Reflection:** *Design resources, as well as the way how working with these resources is organized, are leaving a characteristic signature on a design outcome.*

*It is often possible to guess, from the design outcome, which design resources designers used, and how the working with the resources was organized.*

**You Are Reflection of Your Design:** *Design resources are adapted to a design outcome, sometimes to the point that the form and organization of design resources reflects the intended shape of the outcome.*

**Design-by-Buzzword:** *A designer is joining a growing trend in using some technology, often in an opportunist way. New possibilities of design resources are shaping the design outcome.*

**Conformity:** *To minimize risks in a new design situation, designers use popular and proven resources with established best practices. Such conformity may be driven by positive experiences of others in similar situations, mere popularity of design resources, or mandatory requirements. The usage of a design resource further contributes to establishment and reputation of the resource and its usage in new design situations.*

**Commitment:** *In a new design situation designers use tools they committed to beforehand.*

**Cherry Picking:** *Design situations are selected based on how quickly designers can approach these situations with preferred or available resources. Other situations are avoided.*

## B. Patterns of Dynamics: Mottos

**Co-Evolution of Problem-Solution:** *“If you want to change something, you need to understand it, if you want to understand something you need to change it.”* (Gravemeijer and Cobb 2006)

**Puzzle Solving:** *“A problem well put is half solved.”* (John Dewey, *The Pattern of Inquiry*)

**A Solution Looking for a Problem:** *“When humans possess a tool, they excel at finding new uses for it.”* (Nye 2006)

**Pandora’s Box:** *“Everything we design has the potential not only to solve problems but also to create new ones.”* (Lawson 2005)

**The Force Awakens:** *“How can something seem so plausible at the time and so idiotic in retrospect?”* (Bill Watterson, Calvin, and Hobbes)

**Your Design Is Your Reflection:** *“Every contact leaves a trace.”* (Edmond Locard)

**You Are Reflection of Your Design:** *“We are what we repeatedly do.”* (Will Durant, *The Story of Philosophy*)

**Design-by-Buzzword:** *“Projects that do not capitalize on new opportunities will generally find their products unable to compete.”* Boehm and Bhuta (2008)

**Conformity:** *“When in Rome, do as the Roman’s do.”*

**Commitment:** *“It is a poor craftsman that blames his tools.”*

**Cherry Picking:** *“Do one thing and do it well.”*

## C. Sources of Dynamics: Summaries

**The Moving Target Problem:** *Design situations, resources, and outcomes change more rapidly than designers' understanding of them. Consequently, designers need to work without having significant experience with, and understanding of, design situations, resources, and outcomes. Waiting to obtain more knowledge and better understanding is risky as the issues may quickly become less relevant or obsolete.*

**Increasing Interconnectivity:** *Design situations, design outcome, and design resources are increasingly more globally interconnected. Increasing interconnectivity means that there are more ways or channels through which a design activity may be influenced.*

**Learning and Creativity:** *Behaviors and expectations of people are continually changing due to their learning and creativity.*

**Tacit Knowledge and Skills:** *Designers are relying on tacit knowledge and skills that they do not fully understand and cannot fully explain. Because of its implicit nature, it is hard to predict what effects tacit knowledge will have in design.*

**Social Dynamics:** *Design situations, resources, and outcomes are complex socio-technical systems, with complex, difficult to predict dynamics.*

**Lack of Strong Principles, Theories, and Laws:** *Designers are dealing with poorly understood phenomena, for which we do not have strong underlying principles, theories, and laws.*

## D. Questions to Consider

### Co-Evolution of Problem-Solution:

- Do you continuously investigate, experiment, and iteratively refine your designs?
- How do you resolve uncertainties about design objectives and their priorities?
- Do you use sketching and prototyping extensively?
- How do you estimate and plan projects with the co-evolution pattern?
- How do you decide to design something new versus buying it off-the-shelf?
- Are you reinventing the wheel by co-evolving solutions for well-defined problems with off-the-shelf solutions?
- Are you guilty of the not-invented-here syndrome, routinely building yourself things for any issue you face?
- Have you ever used the off-the-shelf solution that was not appropriate for your domain? What issues have you faced?

### Puzzle Solving:

- In your design activities, do you have lots of well-defined design sub-tasks, such as algorithm implementation?
- Do you use a problem-solving approach for skills development in education and training?
- Have you ever attempted to solve a problem only to realize that the problem is not well defined?
- Have you ever successfully “solved” the wrong problem?

- Have you ever created an unnecessarily complicated solution because you had not well understood the problem?

### **A Solution Looking for a Problem:**

- Have you ever designed something just for fun?
- Do you reserve time for experimentation and exploration of new technologies and methods?
- Do you experiment with new technologies and practices even if you do not have a business case nor a clear idea about its usefulness?
- Do you experiment with new technologies enough to get an opinion about them?
- Are you always playing it safe, using only well-proven technologies and methods?

### **Pandora's Box:**

- How do you manage the risk of encountering unforeseen issues?
- How do you react when unexpected issues occur? Do you have a process for it?
- Have you ever designed something that has caused significant unforeseen problems?
- Have you ever regretted starting your design activity because of these unforeseen problems?

### **The Force Awakens:**

- How do you analyze the market and respond to changes in it?

- How do you anticipate the reaction of competition to your design efforts?
- Have your design efforts ever caused other players (e.g., competition) to react strongly against your activities?
- Have you ever responded strongly to the design efforts of others?
- Have you ever experienced the effect vaporware, a practice of announcing a product that does not exist to gain a competitive advantage and keep customers from switching to competing products?

### **Your Design Is Your Reflection:**

- Do you have your design style? How much do you care about it?
- Do you use templates and standard sets of technologies and materials in your designs?
- Do templates and standards help you to simplify and speed up your design efforts?
- Do these templates and standards limit your design efforts?
- How do you ensure consistency (when needed) in multiple design activities?
- Is your organizational structure aligned with the structure of your design outcomes? How?
- Have you ever experienced the effect of Conway's law, where your designs copied the communication structures of your organization?
- Have you ever build unnecessary complex designs due to wrong organizational structures?
- Have you ever experienced the effect of inverse Conway's law, changing your organization to suit the design outcome structure better?

**You Are Reflection of Your Design:**

- How do you choose technologies, materials, and tools for your designs?
- Do you have preferred tools and techniques?
- Have you ever obsessively used some technology or method, even though it was not the most appropriate for the task at hand?
- Do you adapt your organizational structures for your designs?
- How much effort you spend maintaining your existing designs versus creating the new ones?
- Are you being slowed down by the need to maintain your previous (legacy) designs?

**Design-by-Buzzword:**

- Do you follow the latest trends in design tools and materials, and are you quick to adopt them?
- Have you ever regretted being an early adopter of the latest fads?
- Have you ever regretted not reacting soon enough to the latest trends?
- How do you make decisions to adopt new technologies and methods?
- Do you have organized events, such as hackathons, to systematically, but with limited risk, explore new opportunities?

**Conformity:**

- Do you use standard tools and methods with a significant user base?



- Are you an early adopter for some technologies and methods?
- Are your practices, tools, and methods similar to practices, tools, and methods of other companies or institutions?
- Do you have some unique practices, tools, and methods? Why?
- Do you regret conforming to some standard practices, tools, and methods, as they limit you?
- Are you dependent on a vendor for products and services, and are unable to use another vendor without substantial switching costs?
- Have you ever needed to change your design due to government decisions?

**Commitment:**

- How much freedom do you have in choosing your design resources?
- Do you have some design resources you are committed to using most of the time?
- How do you learn new practices, tools, and methods? When do you give up?
- Do you “eat your own dog food”?
- Have you ever regretted committing too much to some practice, tool, or method?
- Do you have a golden hammer, applied obsessively to many problems?

**Cherry Picking:**

- On what types of design situations you work on usually?

- Do you specialize in any way?
- Do you say no to design opportunities outside your specialization?
- How do you make sure not to overspecialize and become irrelevant once your specialization is not anymore needed?
- Which patterns of dynamics typically occur in your design activities?
- Are there some patterns that occur more often than the others?
- Are there some patterns that do not occur at all?
- How do you estimate the potential impact of your design outcomes, both short and long term?

### **The Moving Target Problem:**

- Are you working in a dynamic domain where things change quickly?
- How do you balance the need to spend more time understanding the situation and designing the outcome versus producing results rapidly?
- How do you deal with uncertainty in design situations?
- How do you react to change?
- How do you respond to significant unforeseen problems?
- Have you ever canceled your design effort due to significant changes in design situation or resources?

### **Increasing Interconnectivity:**

- How many people are involved in your design activities?
- How are they connected?

- How many people are interacting with your design outcomes? Via which channels?
- In which geographical zones are these people located?
- How likely is that a change in remote regions will influence your design activities?

### **Learning and Creativity:**

- How do you come up with new ideas?
- How do you discipline your creativity?
- How do you stimulate your individual and group creativity?
- How do you prevent being unnecessary creative?

### **Tacit Knowledge and Skills:**

- What kind of tacit knowledge and skills you use in your design activities?
- How are you developing these skills?
- How do you combine tacit (implicit) and explicit knowledge?
- How do you capture and share tacit knowledge?
- How do you plan activities that involve extensive usage of tacit skills?

### **Social Dynamics:**

- What kind of culture your organization has? What kind of cultures your customers or users have?
- While designing, how do you take into account cultural differences?

- What kind of political issues you face in your design activities?
- Are you a part of a broader professional community?
- What type of ethical consideration do you meet in design activities?

**Lack of Strong Principles, Theories, and Laws:**

- Do you use any theories or models to analyze, predict or simulate elements of design activity?
- How reliable are such models and theories?
- How do you approach poorly understood situations and problems?

# About the Authors

Željko Obrenović is a CTO of Incision. Before joining Incision, he worked as a principal consultant with the [Software Improvement Group \(SIG\)](#)<sup>43</sup> in Amsterdam, a consultant at Backbase, an assistant professor at the Technical University in Eindhoven, and a researcher at CWI. In his work, he aims at bridging design research and practice.

Website: [obren.info](http://obren.info)<sup>44</sup>

Twitter: [@zeljko\\_obren](https://twitter.com/zeljko_obren)<sup>45</sup>

---

<sup>43</sup><https://www.sig.eu/>

<sup>44</sup><https://zeljkoobrenovic.com/>

<sup>45</sup>[https://twitter.com/zeljko\\_obren](https://twitter.com/zeljko_obren)

**Erik Stolterman** is Professor of Informatics and Senior Executive Associate Dean at the School of Informatics, Computing, and Engineering, Indiana University, Bloomington. He is also a professor at the Institute of Design at Umeå University, Sweden. Stolterman is co-Editor of the Design Thinking/Design Theory book series by MIT Press, and on several editorial boards for international journals (The HCI journal, International Journal of Design, Design Studies, Design, Economics and Innovation, International Journal of Designs for Learning, Studies in Material Thinking, Human Computation: An Interdisciplinary Journal, Artifact). Stolterman's main work is within the areas of interfaces, interactivity, interaction design, design practice, philosophy and theory of design. Stolterman has published a large number of articles and five books, including "Thoughtful Interaction Design" (MIT Press) and "The Design Way" (MIT Press) and the forthcoming "Things That Keep Us Busy—The Elements of Interaction" (MIT Press, 2017).

**Website:** [transground.blogspot.com](http://transground.blogspot.com)<sup>46</sup>

**Twitter:** [@estolter](https://twitter.com/estolter)<sup>47</sup>

---

<sup>46</sup><http://transground.blogspot.com>

<sup>47</sup><https://twitter.com/estolter>