Whitepaper

# The solution design process

The importance of quality and speed in the solution design process to reduce project risks and time to market



aixigo
Driving the New Wealth Experience.

# Introduction


Executive summary I ..... 4

Intro ..... 5

Background ..... 7

The solution ..... 11

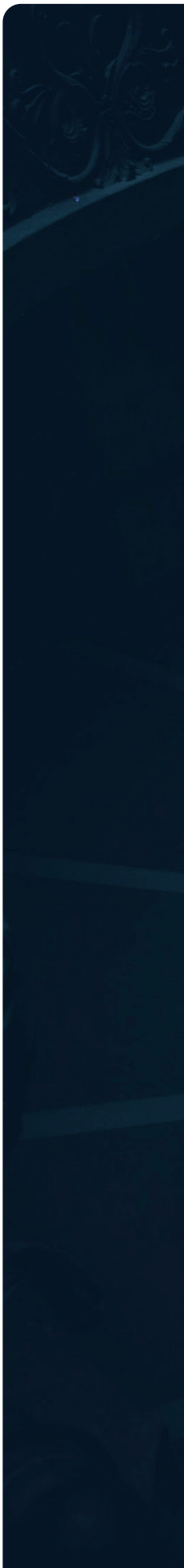
Example ..... 13

Executive summary II ..... 15

Appendix ..... 16

Sources and links ..... 16

Books ..... 17





## Only a third of projects are fully successful (completed in time & budget).[1]

**34%**

of projects were fully successful

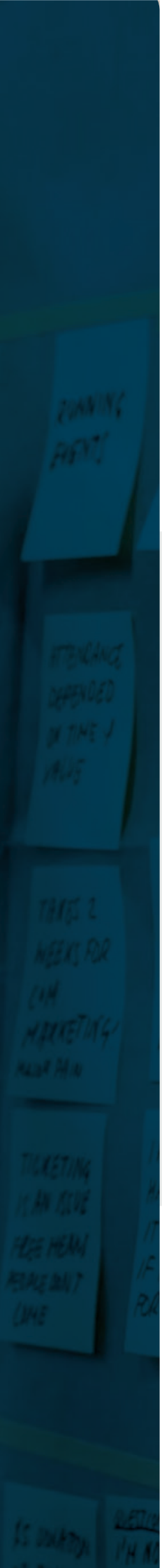
**19%**

of projects will be cancelled before they get completed

**47%**

are challenged (over budget, behind schedule, ...)

[1] [Source: The Standish Group – CHAOS Report 2020]



# Executive summary I

- Only a third of projects are fully successful (completed in time and budget).
- Unclear requirements are a major reason for failure and high costs of change.
- User centricity is important, but not sufficient. Complex projects in a regulated environment need the commitment of all stakeholders.
- The cone of uncertainty can only be narrowed in early project phases by a comprehensive solution design.
- Design is not just what it looks like. Design is how it works.
- The challenge is to design as much as possible while maintaining high quality.
- The only way to escape this dilemma is to avoid superfluous work.
- Imaginable solution design is a high-performance approach for the analysis and design phase to a project. It saves time and costs while providing high quality and speed.

# Intro

Nowadays, we already have a great deal of knowledge, experience, and methodologies to run a successful software development project. What are the reasons why we still see many projects that go out of time, over budget or fail completely?

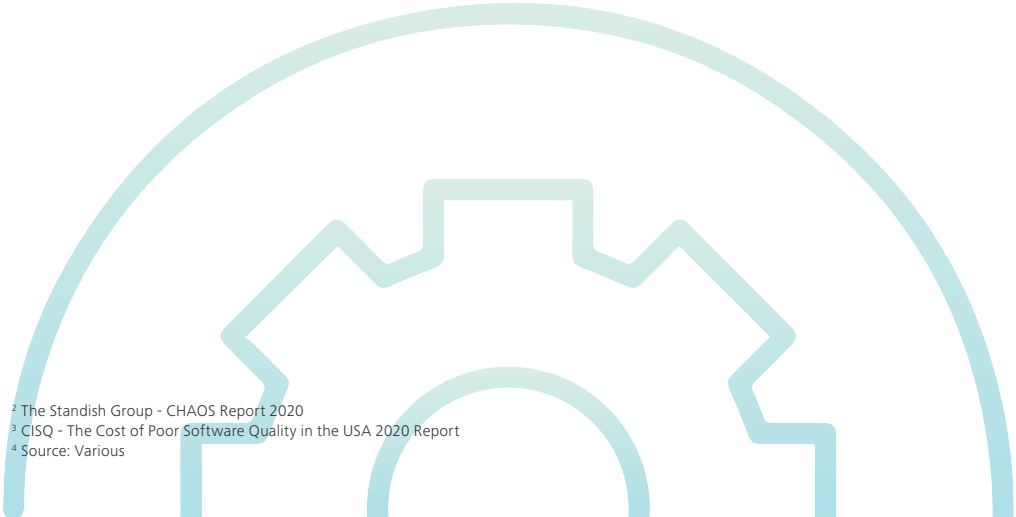
A report by Standish Group published in March 2020 stated, that only **34% of projects were fully successful** in terms of time and budget. They show that a staggering **19% of projects will be cancelled** before completion and **47% are challenged** (over budget, behind schedule, and produce low-quality deliverables).[2]

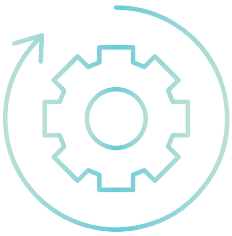
According to CISQ, the **probability of cancelling** bigger projects such as the typical software projects in wealth management is **5-6 times higher** if the process is not focused on quality.[3]

Looking at research about the reasons for the failure of software projects, numerous factors can be identified. **Unclear requirements** are cited as the most important reason in many of the sources. That seems to be a logical consequence as requirements constitute the basis for the whole organisation. Clear requirements form the foundation for quality. All kinds of problems arise from a lack of clear and complete requirements.[4]

**Bad effort and time estimations, wrong resource allocation, poor planning, buggy coding, insufficient testing** – these are the obvious reasons for project failure. But behind them hide unclear requirements. If you don't know what you are supposed to do, it leads to problems.

IBM has found that an **unclear or incomplete specification leads to a huge amount of change** effort late in the project, with the **cost of change being 200 times higher** compared to the costs incurred if the gap had been found in the early analysis or design phase.





[2] The Standish Group - CHAOS Report 2020
[3] CISQ - The Cost of Poor Software Quality in the USA 2020 Report
[4] Source: Various

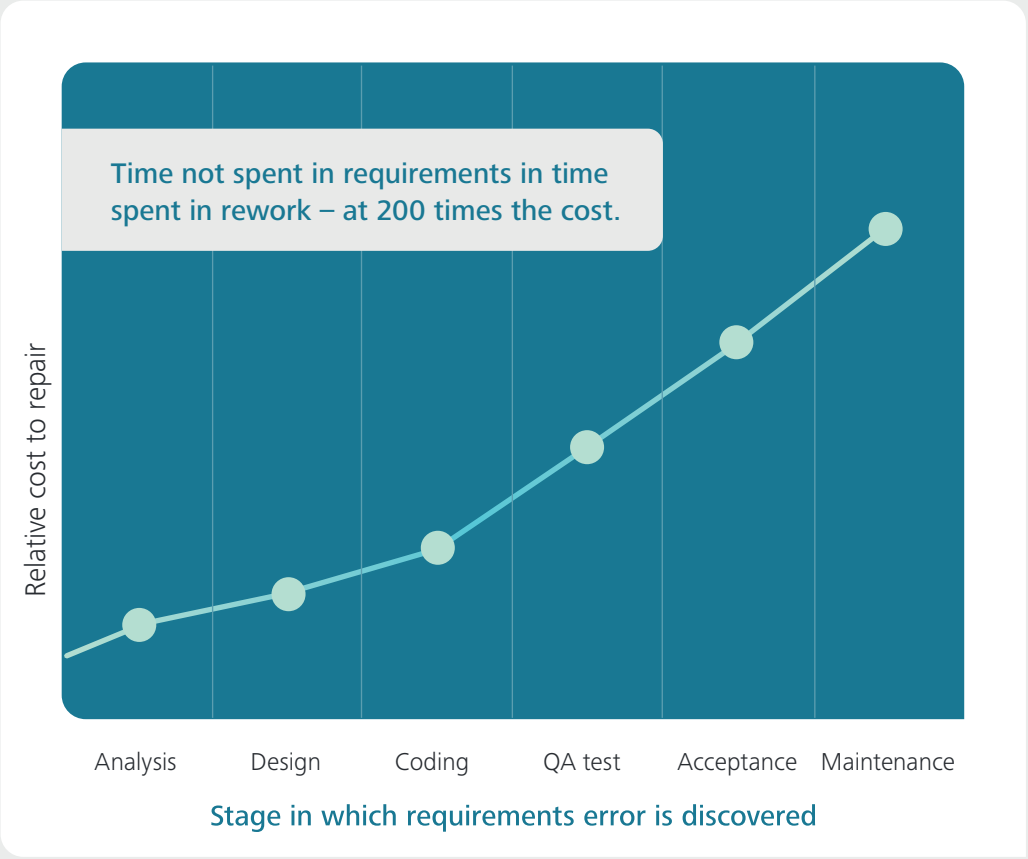



Figure 1: The later errors are discovered in the development lifecycle, the more it costs to fix them.
[Source: IBM – Reducing rework through effective requirements management]

But don't we already have advanced requirements engineering methods? Why don't all of us use requirements engineering, design thinking or design sprints successfully?

The reason is quite simple. Have you ever had enough time to properly implement what is described in theory?

We are **surrounded by constraints** like new regulations, competitors, or business plans. And the solution designer constantly gets confronted with the sentence: "Hurry up, we already spent so much time on discussions, we really need to get started!"



# Background

So, what are the drivers of success for the requirements phase of **typical software projects in wealth management** – a new advisory solution, a robo-advisor, or the discretionary portfolio management solution? Projects that obviously need to be approached in a user-centric way, but without disregarding the massive requirements of business, sales, legal and compliance.

How can you gain commitment of the various stakeholders and provide a sufficient basis for successful project development?

First and foremost, **you need to be fast**, as you lose time at the beginning of the project that you will miss in later phases. Some of you maybe remember that from your diploma thesis, don't you? But you need to be fast while **simultaneously delivering complete and robust solution designs**. How can you approach this dilemma?

The most important basis for the successful creation of a solution design is to understand the concept of design correctly. Design is not meant to be abstract, nor is it just about the way something looks. Most people end up making the mistake of thinking that design is just what it looks like.

> **"[…] That's not what we think design is. It's not just what it looks like and feels like. Design is how it works."[5]**

It is therefore crucial to have a complete and, above all, robust picture of the solution before discussing colours and pixels. To ensure this, you **need to have the commitment of all stakeholders**. Any need of a stakeholder you have not considered will lead to high costs in case of changes. According to the Carnegie Mellon Software Engineering Institute, data indicates that **60-80% of the cost** of software development **can be attributed to rework**. In fact, **requirements defects account for 70 to 85% of rework costs**. As already mentioned, errors found late in the development process can cost organisations up to 200 times more than if they were detected during the requirements analysis phase.[6]

"One of the most effective ways to cut costs, boost productivity and bring software to the marketplace faster is to reduce rework by **improving requirements management**".[7]

The different **stakeholders are very heterogeneous** and represent, in addition to the project sponsors, the various responsible departments (e.g. sales, product, compliance) as well as the users of the solution. How can you **enable** all the different **stakeholders to buy in** and **collaborate** in the requirements phase?

[5] Steve Jobs in NY Times - The Guts of a New Machine
[6] IBM - Reducing rework through effective requirements management
[7] IBM - Reducing rework through effective requirements management

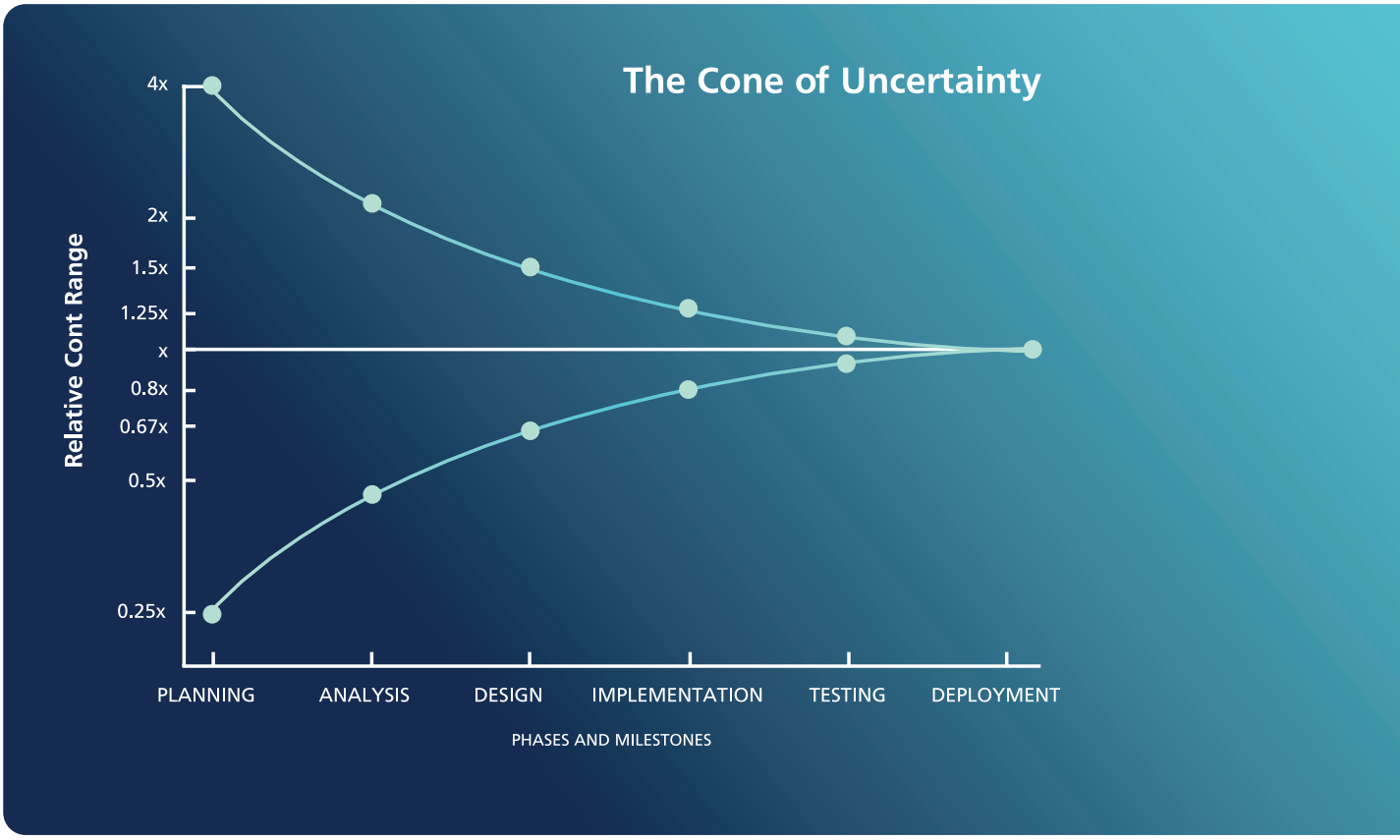
First, you need to make sure everybody involved understands the complexity and the fact that their own picture of the situation is only part of the whole system. Have you ever heard of Tom Wujec and his "how-to-make-toast-method"? It is amazingly easy to show everybody, that there are many different approaches to a simple-looking task. This is a good preparation for a group to open up for a fruitful discussion about the future solution.

The "how-to-make-toast-method" works with visualising what everybody keeps in their minds. And this works the other way round as well. You need to give stakeholders a sense of what will be viable in the future in order to concretise their imaginations. You need to present them a **picture of the future solution**, which means a prototype. A prototype is not something abstract like written down user stories. But building a prototype can be quite "expensive", as working out all the details requires a lot of time and money.

As a result, you need to choose **wisely what to design and how to gain stakeholder commitment** to create a complete as well as robust Solution Design for development.

For **complex solutions** like the already mentioned typical wealth management **advisory solution**, **robo-advisor** or the **discretionary portfolio management solution**, the question of what to design is quite clear. You need to design all the use cases, including the edge cases, to make sure that none of the **critical requirements** in terms of civil or supervisory law are not fulfilled.

A concept describing the relationship between unclear requirements and uncertainty is the **cone of uncertainty**, introduced by Barry Boehm in his book "Software Engineering Economics" and further developed by Steve McConnell in "Software Project Survival Guide".



As the preceding example illustrates, uncertainty can be **significantly reduced by proper analysis and planning**. Barry Boehm notes in his book: "Whatever the strengths of a software cost estimation technique, there is really no way we can expect the technique to compensate for our lack of definition or **understanding of the software job to be done**."

But how can you **be fast** if you are supposed to **build everything** in a way every **stakeholder can imagine**? The answer is to **avoid needless work** (waste). Limit the work spent on things that end up being thrown away. During a design phase, many things will be discarded.

It is essential to:

- Avoid designing things that are pointless, as they violate certain regulatory requirements that the designers were simply not aware about.
- Avoid designing things in a high-fidelity way before you have a stable basis. Therefore, start with wireframe prototypes.

By avoiding needless work, you can easily **save thousands of euros of the budget** and **months of valuable project time**. Moreover, you will not lose, but gain quality by spending more time on use cases and focusing on how the new system will work instead of discussing about how things should look like. We all know those endless discussions about whether the button has the right red or if another headline might be better. Just to clarify: **It is important to work on the details, but there are different phases for different kinds of details.**

**Avoiding needless work** is no new recommendation. It **strikes at the core of success** for one of the most successful car companies on the planet … **Toyota**. The Toyota Production System (TPS) initially contained the following seven types of waste: **Transportation**, **Inventory**, **Motion**, **Waiting**, **Overproduction**, **Extra-Processing** and **Defects**. In the 1990s, an eighth type was added: Waste of **Unused Talent** or 'skills' of workers.




Figure 2: 8 Types of Waste
[Source: The Lean Way – The 8 Wastes of Lean]

In the process of designing a software solution, the highlighted types of waste should be avoided.

Does all that mean we recommend returning to the classic waterfall project approach, where everything is designed in detail before the first line of code is being produced? Not at all. For us, it is **not a question of waterfall against agile**, but rather a question of combining the **strengths of both approaches**. You might **call it hybrid** or just **agile**.

Our typical wealth management software projects are complex and even an MVP easily exceeds the scope of a simple and trivial project. This is nothing you can deliver within only 2 or 3 sprints. And on the other hand, the result is not completely open and can be decided and designed in all facets by a project team. To enable the project team to go agile, **a high-level solution design** can offer **orientation** for the entire **agile development process**. It represents a **NorthStar** for the project team.

Our recommendation is to start with a high-level design of the complete solution before switching to agile to specify, design the details as well as develop and test.

# The solution

So, how can all this be condensed into a proposal for the so-called **imaginable solution design**? Let's start with some prerequisites to establish the team and the process. Our mantra is **quality and speed**, and we have learned over the last 14 years that you need six key ingredients:

1. You need a **team of experts** who are highly skilled in both **wealth management and solution design.** A designer trying to understand the complex requirements of advice or portfolio management in our regulated world would not succeed.

2. You need **experience** to create a solution design that is **robust, yet flexible** to deal with the **complex and changing ambient conditions.** "It took me a few seconds to draw it, but it took me 34 years to learn how to draw it in a few seconds." (Paula Scher, Designer)

3. You need **empathy** for the **different stakeholders** who naturally hold **opposing but legitimate needs and requirements.** "You can't understand good design if you don't understand people. Design is made for people." (Dieter Rams, Designer)

4. You need to **put the needs of stakeholders at the center of your approach.** Even if all the different requirements further complicate and challenge a designer's work. Nevertheless, the focus is not on the designer in the first place, but on designing something that fulfils the needs of the stakeholders.

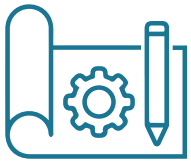
   **"Design works better without ego."** (Jasper Morrison, Designer)

5. A team that is too large should be avoided to prevent excessive coordination efforts. **A well-rehearsed team of 2 or 3** provides the ideal conditions for working fast.

6. You need to place your **complete focus** on the team and the stakeholders. Resist the temptation to handle other "important" things at the same time. We are simply not made for real multitasking.

If you have a **high-performance team** to do the job, they also need a **high-performance methodology**. Design thinking and the design sprint were two great inspirations for us. We adopted these to match the requirements we are faced with when creating solution designs in wealth management.

Always starts with a **comprehensive conversation to understand individual requirements or constraints besides** the **known business or regulatory constraints**. We really believe that conscious listening results in better understanding. Therefore, consciously listen to all stakeholders and ensure that you have understood them correctly by echoing what you have heard. This can even lead to questioning the status quo, because "we have always done it this way" is sometimes not the ideal starting point for a new solution.



**Save time and effort** by working **with wireframe prototyping** in a comprehensive way. Comprehensive way implies designing wireframe prototypes with a lot of details, showing the different screens in many different states and providing detailed screen flows for all relevant use cases. That sounds a lot, but it can be done easily with wireframing, as opposed to high-fidelity prototyping like Figma, etc. "High-fidelity prototyping requires a lot of design effort. You can't justify this effort at this stage of the project, mainly because the project's scope has not yet been signed off, and the internal and external stakeholders haven't yet agreed on the deliverables."



Focus on creating a solution based on **robust concepts** that can be **used for different tasks** and can easily be **adapted to new requirements** (e.g. new types of products, etc.).

**Iterate the draft 3 to 5 times** with all stakeholders (incl. three representative users of the new solution) to **gain broad agreement**. This might include user tests based on clickable prototypes for specific targets, which can also be done based on a wireframe prototype. You really do not need high-fidelity prototypes for this.[8]

As a result, you will get **robust concepts within 4 to 6 weeks**. You **keep momentum** und **save time and budget**. Including all stakeholders and giving them the opportunity to envision the future solution will raise their **commitment** to the project. Completeness of the solution design **reduces your cone of uncertainty**, which leads to better estimations. Better estimations are, in turn, the prerequisite for **reliable splitting into MVP (Minimum Viable Product) and subsequent cycles**. Designing how the solution is supposed to work enables **quality driven development** that significantly **reduces the cost of change** and **raises the probability of a successful project**.

**Imaginable solution design** as described, is a **booster for the project**, as a clear consensual high-level picture of the future solution speeds up everything.

[8] balsamiq, Nielsen

# Example

At aixigo we developed this approach over the last years in the field while delivering numerous projects. We call it the **aixigo Project Booster**. It is a flexible approach allowing imaginable solution design to be developed with high quality and speed. We adjust the length of the different process phases to the complexity of the future solution. The process as such stays the same.

The following example shows a wireframe prototype and a screenshot of the real solution. As you can see, the later solution is imaginable quite well based on the wireframe.

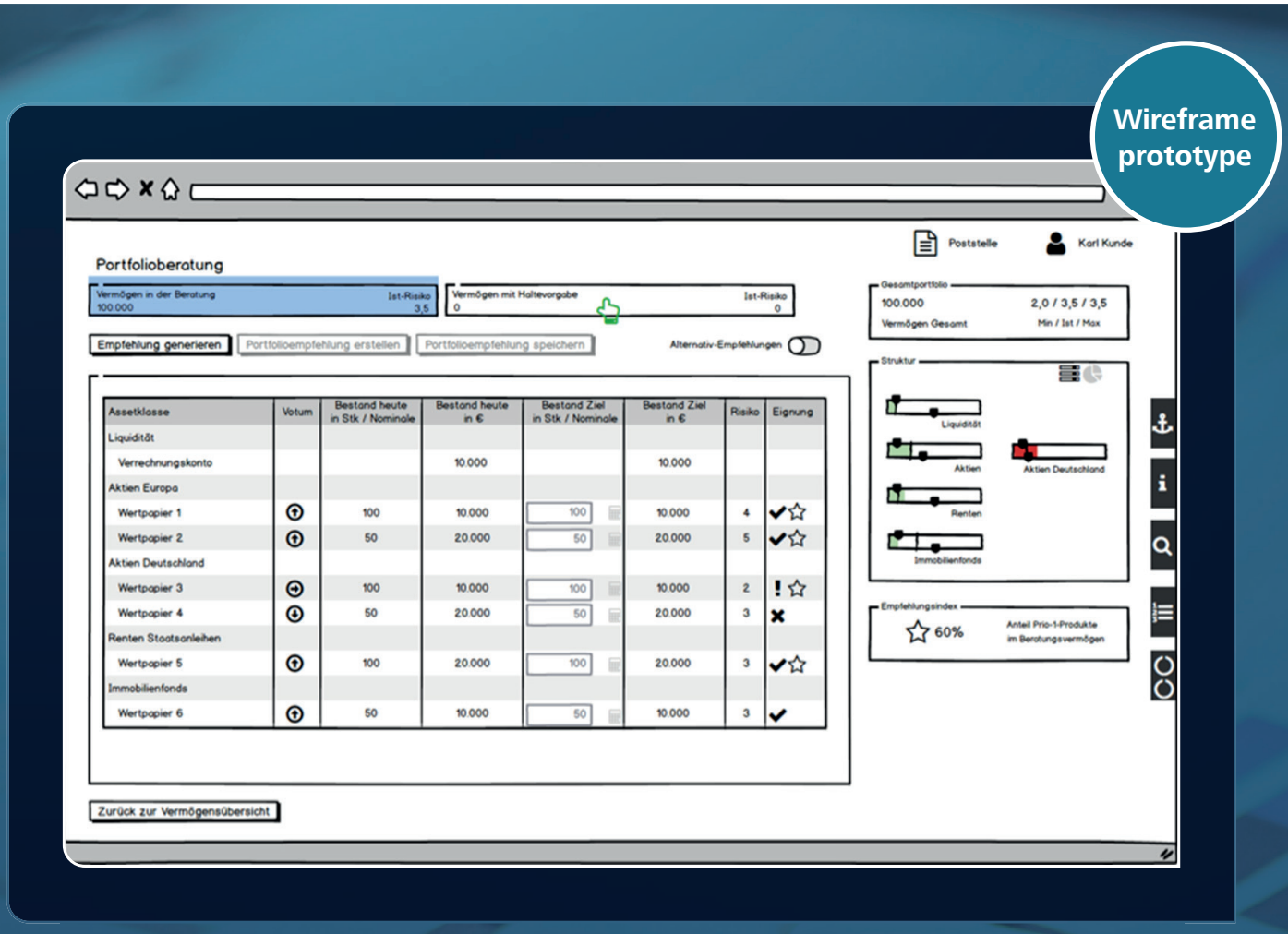



Figure 3: Wireframe prototype



The wireframe prototype for this advisory solution included 350+ slides of screens in different user scenarios organised in 12+ user flows. That made it imaginable for the different stakeholders.

We focused on understanding all the different requirements and getting every stakeholder on board of the concept. This enabled the project team to do all their work on a stable ground, facilitated estimations and planning, and saved a huge amount of expensive high-fidelity prototyping and costs for changes in late project phases.

**The solution**

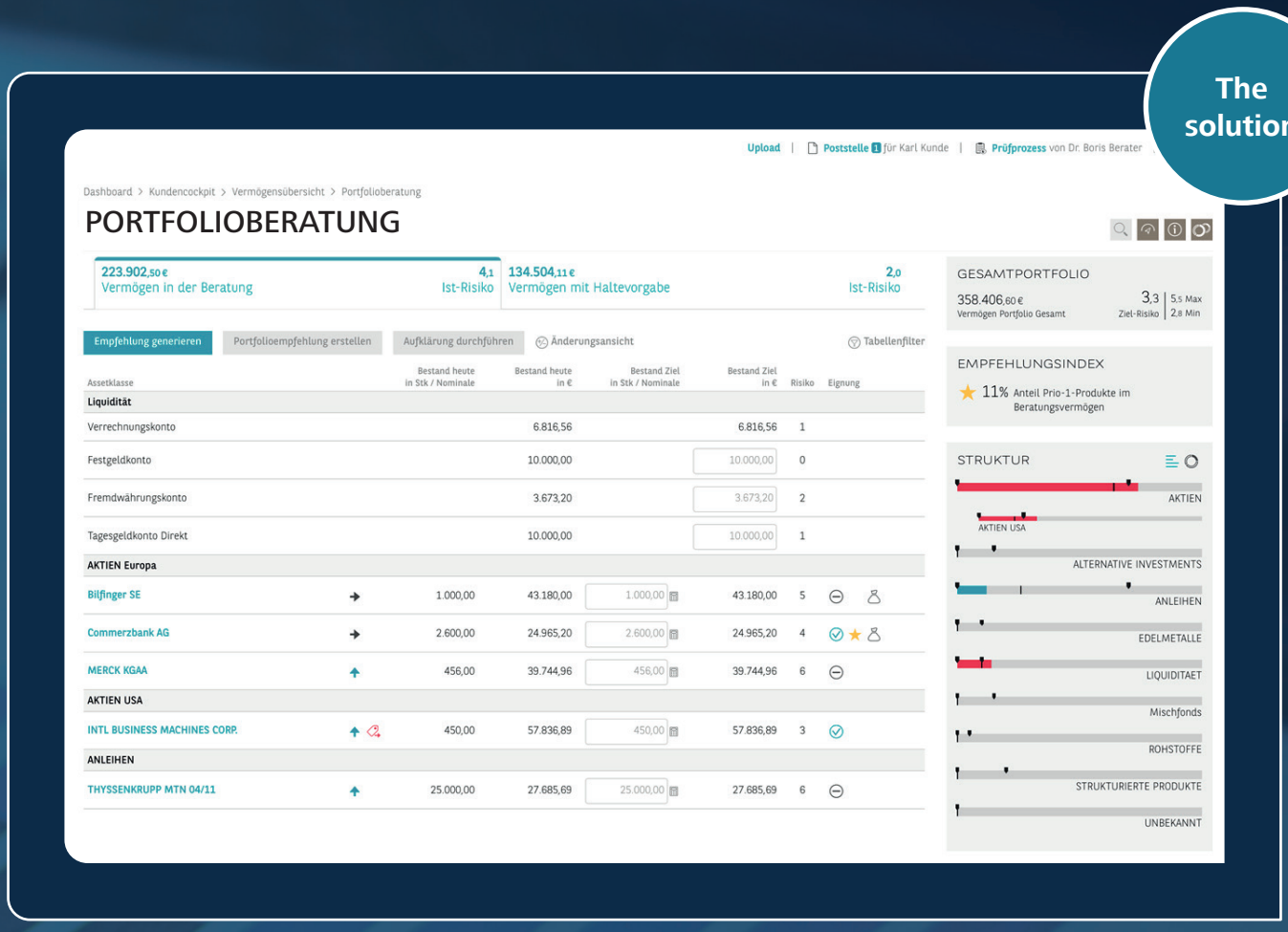



Figure 4: Screenshot of the realised solution

# Executive summary II

**Imaginable solution design** is a high-performance approach for the analysis and design phase of a project that delivers high business value:

- It delivers high quality and speed.
- It saves time and budget.
- It reduces project risks.
- It avoids avoidable work and prefers wireframe prototyping.
- It uses a well-rehearsed team of two or three people with in-depth expertise in the field combined with the capability to design solutions.
- It uses a flexible design process that can easily be adapted to different levels of scope.

**Doing it right can make an enormous difference in the quality of your deliveries.**

**Bernard Ehrlich**
Director Wealth Lab
LinkedIn Account Bernard Ehrlich
bernard.ehrlich@aixigo.com



# Sources and Links

## Solution Design

- https://softwaredominos.com/home/software-design-development-articles/high-level-solution-design-documents-what-is-it-and-when-do-you-need-one/
- https://softwaredominos.com/home/software-design-development-articles/write-solution-design-document/
- https://softwaredominos.com/home/software-design-development-articles/solution-design-and-its-role-in-successful-projects/
- https://softwaredominos.com/home/software-design-development-articles/a-concise-practical-and-current-evaluation-of-agile-and-waterfall-how-to-choose-a-methodology-that-works-for-you/#6-4-understanding-cost-of-change
- https://softwaredominos.com/home/business-management-articles/engineering-superior-production-processes-a-no-nonsense-guide-for-everyone/
- https://www.altexsoft.com/whitepapers/estimating-software-engineering-effort-project-and-product-development-approach/
- http://www.ittoday.info/ITPerformanceImprovement/Articles/2012-06Otero.html
- https://softwaredominos.com/home/business-management-articles/decision-making-in-a-professional-environment-techniques-and-pitfalls/
- https://adamfard.com/blog/mvp-design-sprint
- https://www.it-cisq.org/the-cost-of-poor-software-quality-in-the-us-a-2020-report.htm
- https://www.forbes.com/sites/forbestechcouncil/2020/03/31/14-common-reasons-software-projects-fail-and-how-to-avoid-them/?sh=35d7e52b798c
- https://thescalers.com/reasons-software-projects-fail/
- https://thecodest.co/blog/why-do-software-projects-fail-most-common-reasons/
- https://konradweber.medium.com/handling-with-stakeholders-in-larger-organisations-to-drive-innovation-with-design-thinking-69f6fd1ffc32
- https://blog.jambo.cloud/10-tips-improve-how-you-engage-with-stakeholders
- https://airccse.org/journal/ijsea/papers/6515ijsea02.pdf
- ftp://aix.boulder.ibm.com/software/sg/rational/podcasts/2011/RAW14192USEN.PDF
- https://ivanjureta.com/wp-content/uploads/2021/08/How-Stakeholders-Commitment-May-Affect-the-Success-of-Requirements-Elicitation.pdf

## Wireframe Prototyping

- https://balsamiq.com/learn/articles/usability-testing-wireframes/
- https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

## Toyota

- https://www.inc.com/chris-mcgoff/eliminate-waste-and-improve-efficiency-by-using-to.html
- https://www.allaboutlean.com/muda/
- https://global.toyota/en/company/vision-and-philosophy/production-system/
- https://www.apo-tokyo.org/publications/p_glossary/7-wastes-2/
- https://theleanway.net/The-8-Wastes-of-Lean
- https://kanbanize.com/blog/stop-the-line/

## How to make toast

- https://www.drawtoast.com

## Listening

- https://www.ted.com/talks/julian_treasure_5_ways_to_listen_better

## Books




**Design Thinking Das Handbuch**
Falk Uebernickel, Walter Brenner, Therese Naef, Britta Pukall, Bernhard Schindholzer



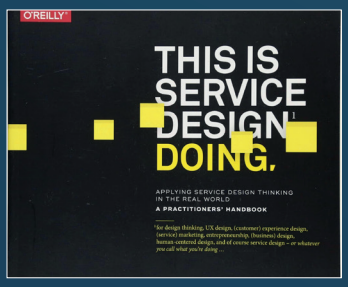

**Das Design Thinking Playbook**
Michael Lewrick, Patrick Link, Nadia Langensang




**Value Proposition Design**
Alexander Osterwalder, Yves Pigneur, Gregory Bernarda, Alan Smith, Trish Papadakos



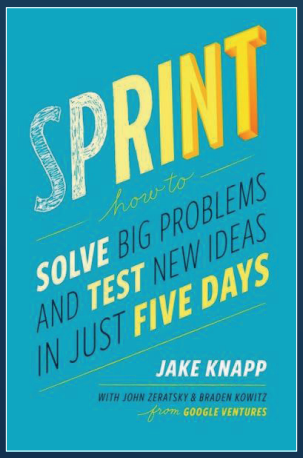

**Digital Innovation Playbook**
Dark Horse Innovation




**This is Service Design Doing**
Marc Stickdorn, Markus Edgar Hormess, Adam Lawrence, Jakob Schneider



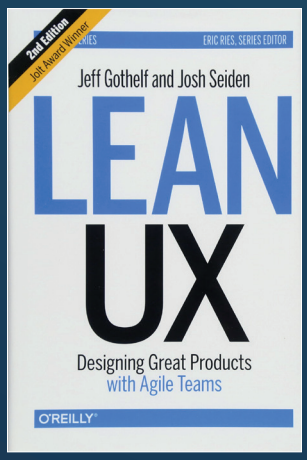

**Creative Explosion**
Henning Patzner




**Sprint**
Jake Knapp, John Zeratsky




**This is Service Design Thinking**
Marc Stickdorn, Jakob Schneider




**Lean UX**
Jeff Gothelf, Josh Seiden