# Design Templates

# Design Templates

Create and test a design prototype that demonstrates the imagined “to-be” solution.

## Keystone Activities

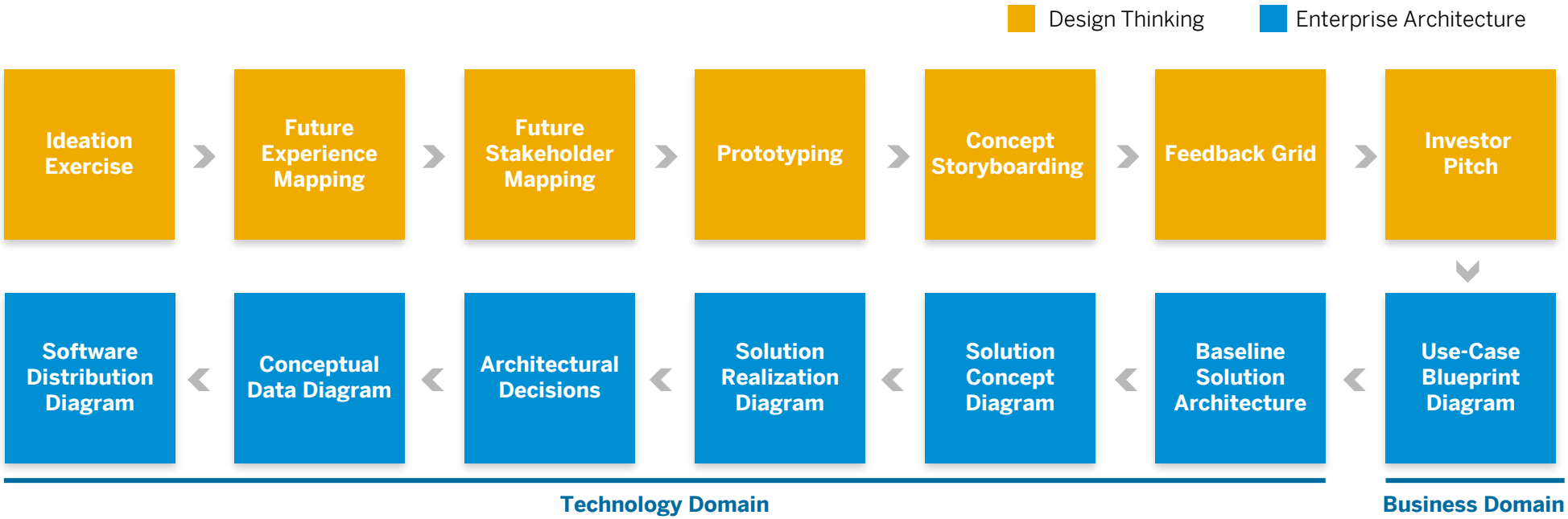
Ideation, prototyping, testing & iteration are critical for a complete design cycle.

## Preparation

People: Include end users **who have been observed**, key stakeholders and business IT during the process.
Place: Dedicate a creative space for both co-design activity

## How-To

Design Templates will help you generate, prototype, and test ideas for your selected innovation use cases. We curated the  templates in the following order based on hundreds of customer engagements. Feel free to use the way you find it most useful.



# Table of Contents

Ideation Exercises ..... 5

Future Experience Mapping ..... 14

Future Stakeholder Mapping ..... 19

Prototyping ..... 25

Concept Storyboarding ..... 28

Feedback Grid ..... 32

Investor Pitch ..... 36

Use Case Blueprint Diagram ..... 40

Baseline Solution Architecture ..... 44

Solution Concept Diagram ..... 48

Solution Realization Diagram ..... 53

Architectural Decisions ..... 58

Conceptual Data Diagram ..... 62

Software Distribution Diagram ..... 67

# Ideation Exercise

Template | Instructions | Example | Protocol

Helping your team generate ideas in structured yet creative ways.

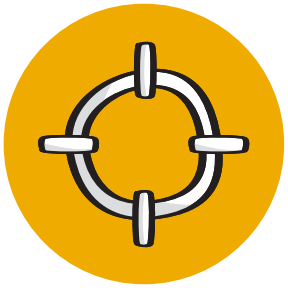
# Brainstorming Mindsets

**Yes, and…**
Basis for any idea generation

**No, but…**
Inhibitor of idea generation

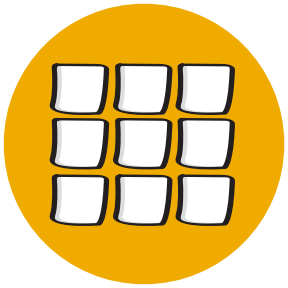
# Some Brainstorming Rules

Be visual



Stay on topic

Build on the ideas of others



Go for quantity

Defer judgments

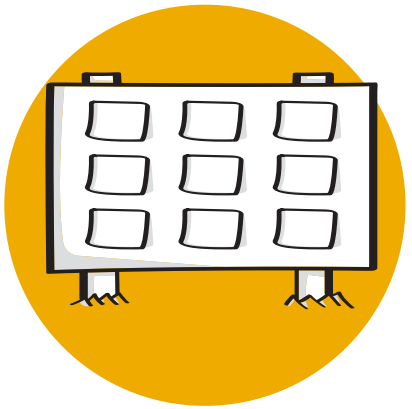
One conversation at a time

Encourage wild ideas

# Three Ideation Techniques

**Free Flow**

**Brainwriting**



**Technology Wall**

Three Ideation Techniques

# 1. Free Flow Ideation

Individually brainstorm ideas. One idea per post-it. (2 min)

Share your post-its with team using Lightning Sharing. (10 min)

- Lightning sharing: Cluster as you go. Subsequent sharers only add new content.

Vote for your favorite ideas & select the winning idea(s) for your team.

Three Ideation Techniques

# 2. Brainwriting: Building on the ideas of others


**Choose one "How might we..?" question**

*Each team member chooses one "HMW…?" questions and notes that down on the template.*


**Start with 3 ideas**

*Each team member starts with 3 ideas on the brainwriting template.*


**Rotate after 3 minutes**

*After 3 min the template will rotate clockwise, and each team member generates the next 3 ideas that are building on the previous ideas on the template.*


**Back to step 3**

*Repeat step 3 until each brainwriting template has fully rotated.*

Three Ideation Techniques

# 2. Brainwriting: Building on the ideas of others

How might we...?

HMW 1

| Round | Idea 1 | Idea 2 | Idea 3 |
|---|---|---|---|
| *1st Person* | ... | ... | ... |
| *2nd Person* | ... | ... | ... |
| *3nd Person* | ... | ... | ... |

Three Ideation Techniques

# 3. Technology Wall

**Write down the key technologies and trends, you would like to relate to the ideation process**

*(Machine learning, Sensors, …)*

**Pick one technology**

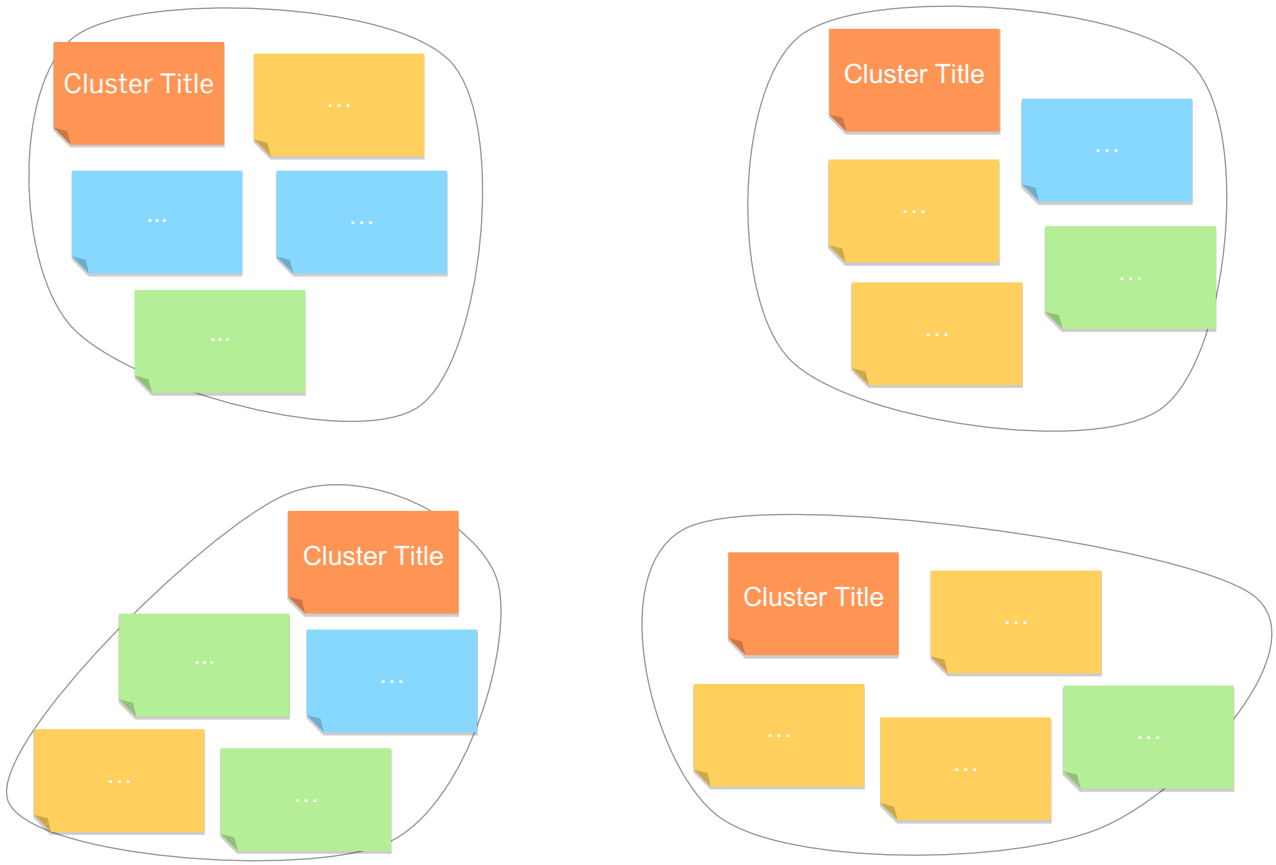
**Brainstorm on solution ideas that include the technology picked.**

*“What could this technology mean for the to be designed solution?”*

**Start again with another technologies / trends**

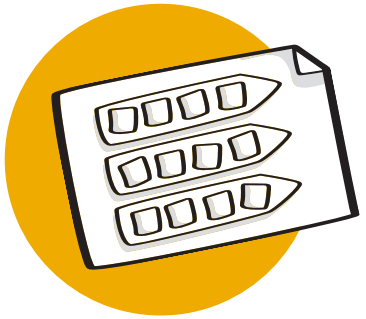
# Clustering
## Ideas

- Sort your ideas into categories or buckets
- Which ideas are related to each other in some way?



# Future Experience Mapping

Template | Instructions | Example | Protocol

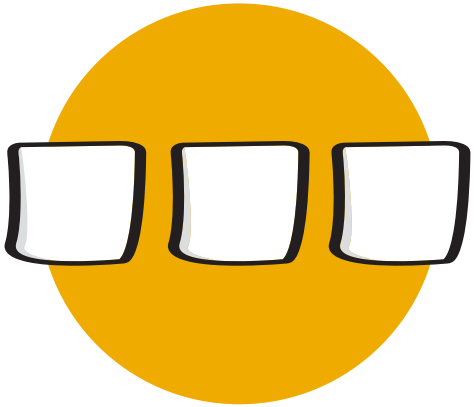


Mapping the "to-be" process based on the  proposed idea through the eyes of the user. It allows a team to gain a common understanding of the proposed idea.

# Future Experience Mapping (To-be process)

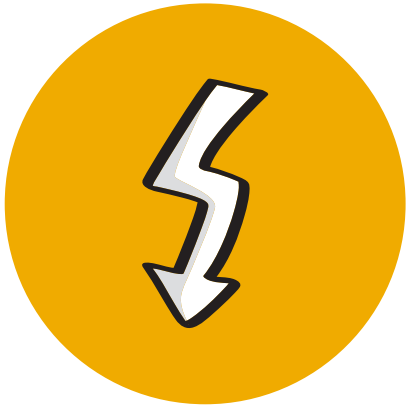
## Slide for Presentation

Make the idea concrete by mapping the experience in context of the persona



**Write down the actions step by step**

**Write down the corresponding mindset and touch points**



**Mark the magical moments and moment of truth**

# Future Experience Mapping Instructions

Duration

**60-120 minutes**

Number of Participants

**3-5 participants**

## Why & What

It is an exercise used to layout the user experience in a chronological order, step by step, on a whiteboard or on a big poster. Knowledge about the design challenge and the user is key to conduct this exercise.

We use this activity both to structure the To-Be Process of the Design phase, and to capture the As-Is Process during the Discover phase.

## How to use it

1. Center lane: Write down the actions step by step. What actions does the user take while trying to achieve their goal and/or fulfill their tasks?

2. Top lane: Write down the corresponding mindset. What is on the user's mind during this journey? How do they feel at each step of their journey?

3. Bottom lane: Write down the corresponding touch points. What touch points does the user have? What do they engage with while on the journey (tools, devices, conversations, other people, etc.)?

4. Mark the magical moments where the proposed idea addresses the pain points and mark the moments of truth.

## Tips & Tricks

Moment of truth

A "moment of truth" describes a situation when something could go wrong and/or in which critical decisions have to be made.

Magical Moments

Where the proposed idea addresses the pain points and flips them to positive experiences.

Pain points

Situations that the user finds uncomfortable, frustrating or difficult are called "pain points".

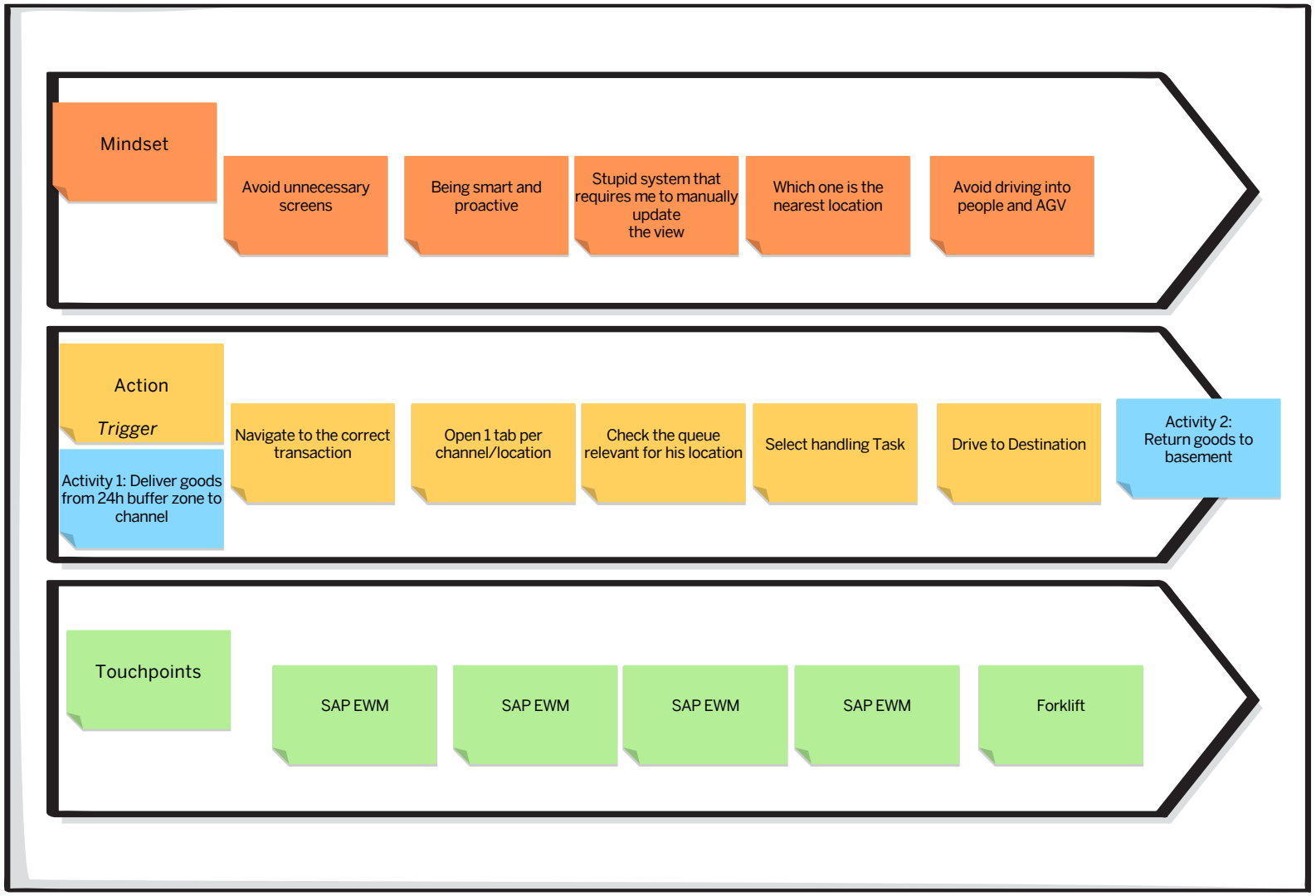
Tip

If ideas come up during the exercise, put them to an idea parking lot.

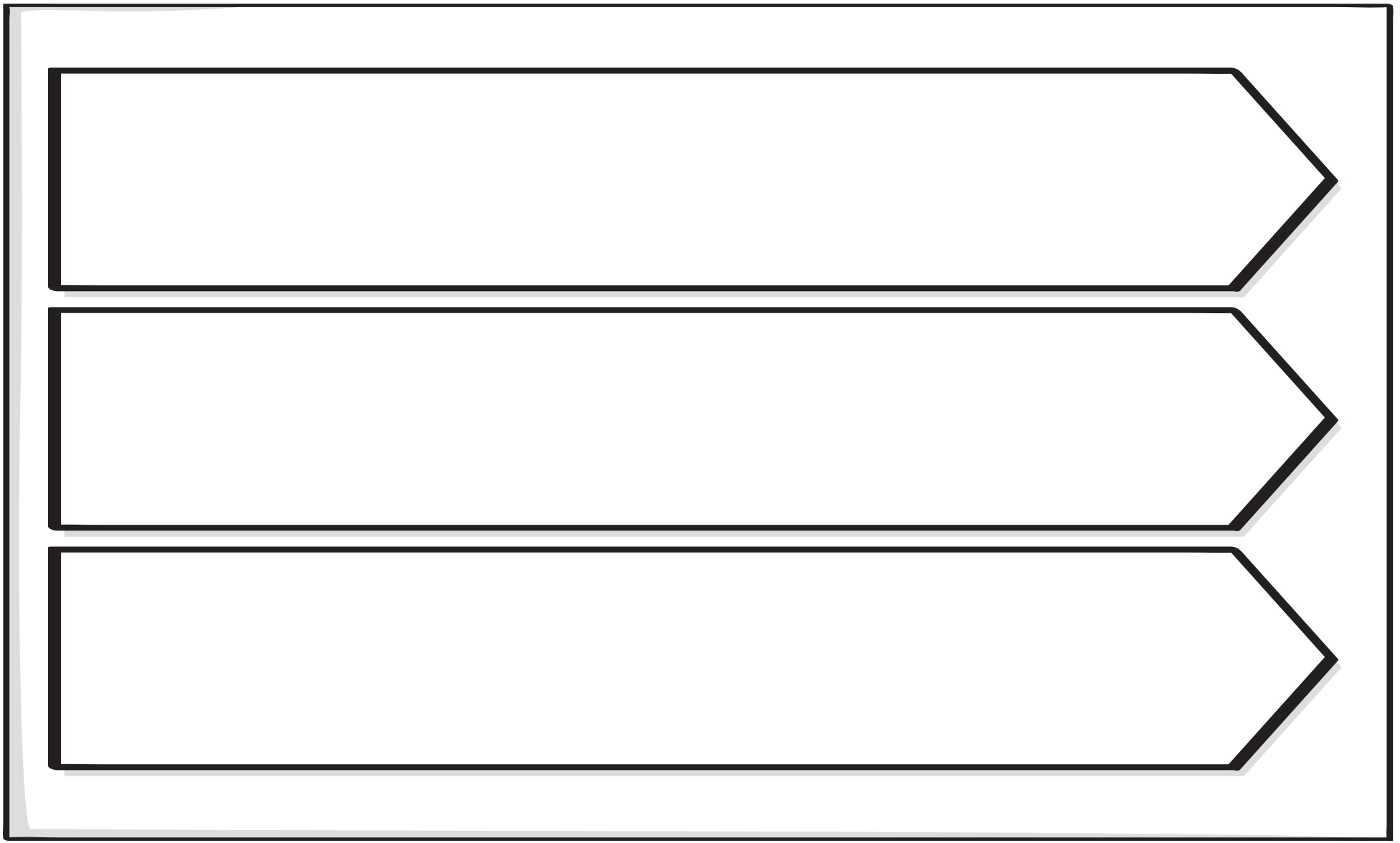
# Future Experience Mapping Example

Persona: Michael, Forklift Driver

Activity 1: deliver goods from 24h buffer zone to channel

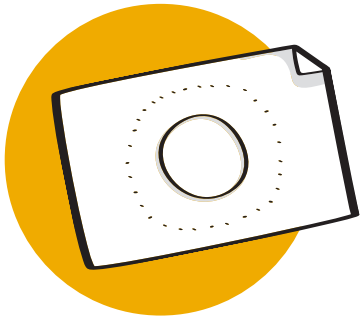


# Future Experience Mapping Template



# Future Stakeholder Map

Template | Instructions | Example | Protocol



Understand key stakeholders and how they interact and influence each other in context of the proposed idea. Bring alignment among team members around the organizational context of the proposed idea. Make assumptions transparent.

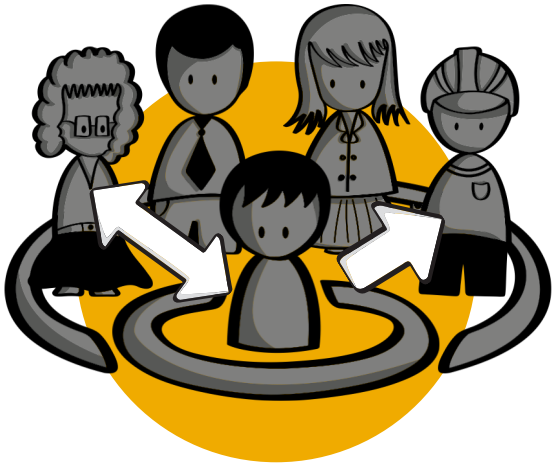
# Future Stakeholder Map

What are all the Stakeholders involved around the proposed design?


**Identify the business user / scenario**

**Capture all of the relevant stakeholders**

direct/indirect, internal/external



**Mark how they interact and influence each other**

# Future Stakeholder Map Instructions

Duration

**30-60 minutes**

Number of Participants

**2-10 participants**

## Why & What

Understand key stakeholders and how they interact and influence each other.

Bring alignment among team members around the organizational context of the project. Make assumptions transparent

Stakeholder mapping defines the roles of people and their relationships in a “human-centered system” view.

It provides a way of visualizing the many dynamics in play, including motivations, influence and relationships.

## How to use it

1. Identify the business user / scenario

2. On individual post-it notes, capture all of the relevant stakeholders that may be affected by or cause impact in the user challenge/scenario.

Discuss with your team to be sure you have an exhaustive list

3. Place stakeholders on your stakeholder map appropriately, visually displaying their relation to the problem

## Tips & Tricks

Think internal & external

Throughout the project, refer back to the map often to ensure

you are maintaining contact with all relevant parties.

Add to the map as needed, if the focus of the project shifts or new stakeholders are identified.

# Future Stakeholder Map Cheat Sheet

Duration

**30-60 minutes**

Number of Participants

**2-10 people**

## How to Run

1. Identify the business user / scenario

2. On individual post-it notes, capture all the relevant stakeholders that may be affected by or cause impact in the user challenge/scenario.

Discuss with your team to be sure you have an exhaustive list.

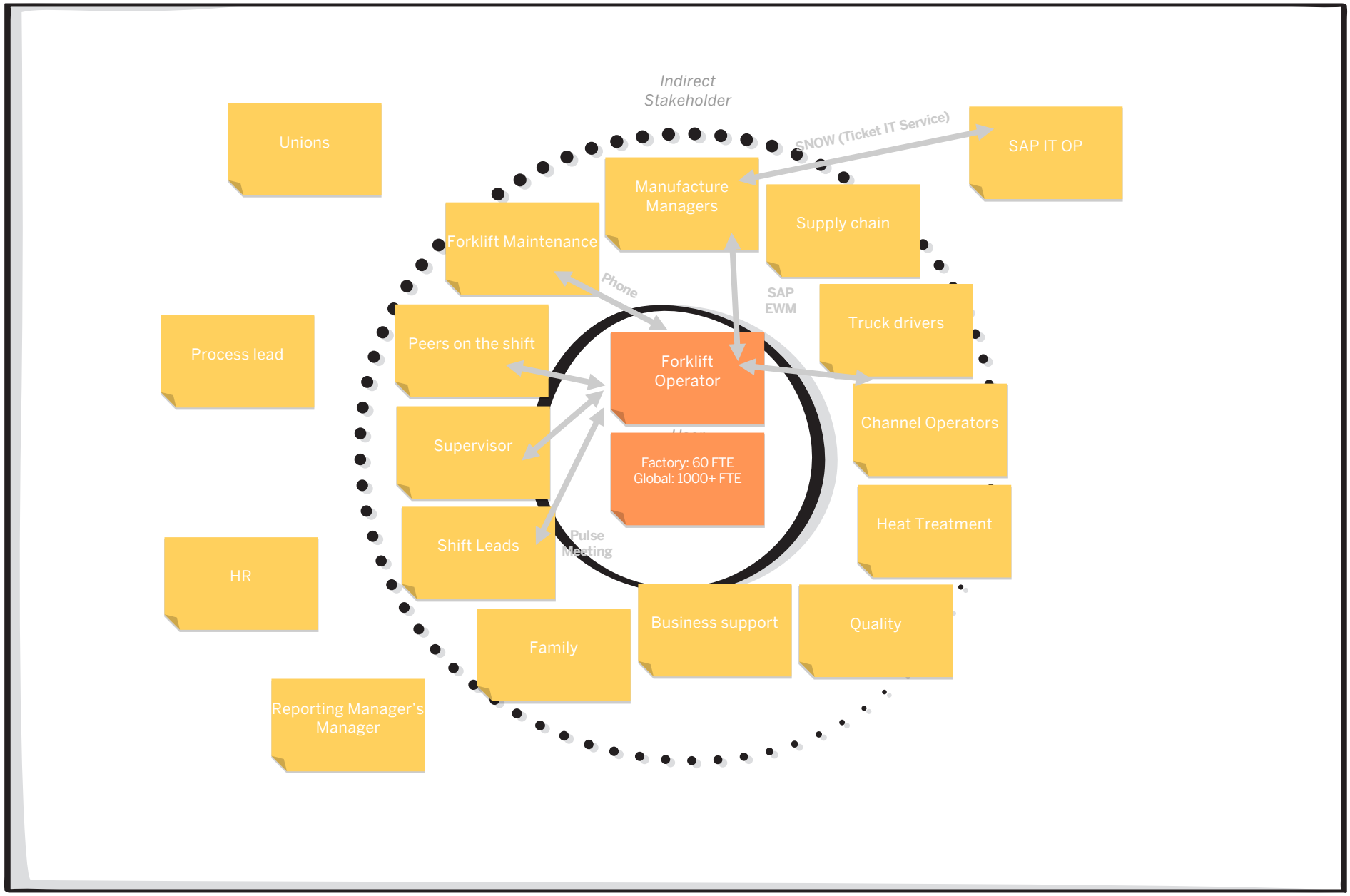
3. Place stakeholders on your stakeholder map appropriately, visually displaying their relation to the problem

## Tips & Tricks

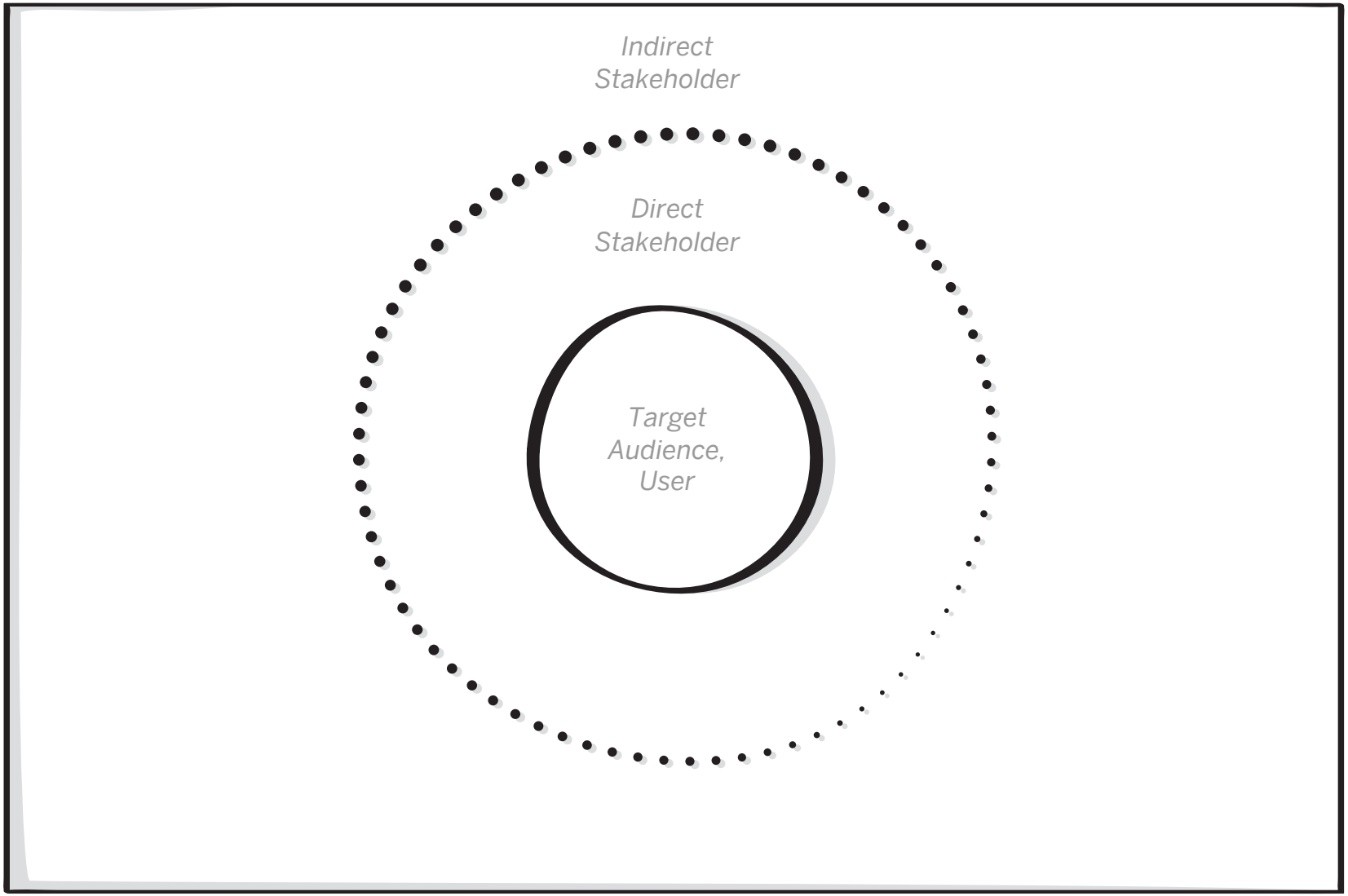
Think internal & external. Throughout the project, refer to the map often to ensure you are maintaining contact with all relevant parties.

Add to the map as needed, if the focus of the project shifts or new stakeholders are identified.

# Future Stakeholder Map Example

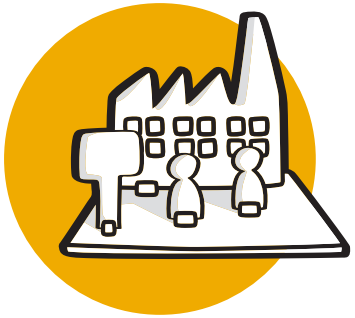


# Future Stakeholder Map Template



# Prototyping

Template | Instructions | Example | Protocol



Making ideas tangible by creating a model of how the proposed idea might work. This helps the team to test the idea with end users and provide alignment between team members.

# Why Prototyping?

**Develop and iterate ideas**

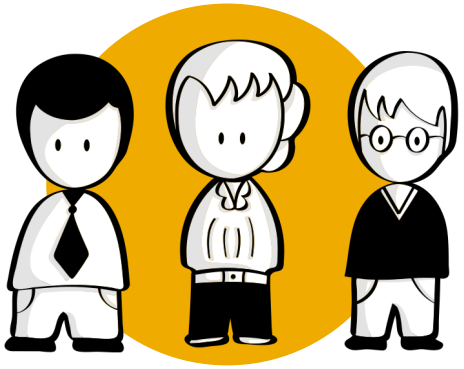
**Test and Learn**

**Communication and common understanding**

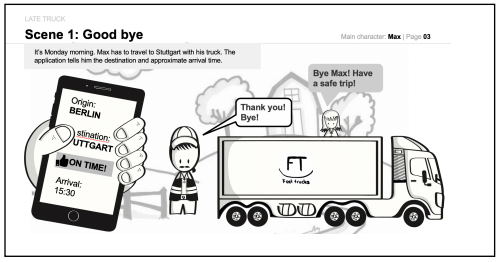
# Other Types of Prototypes

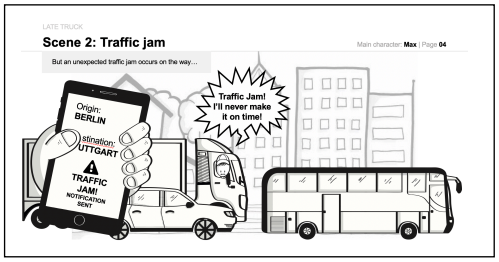
Physical

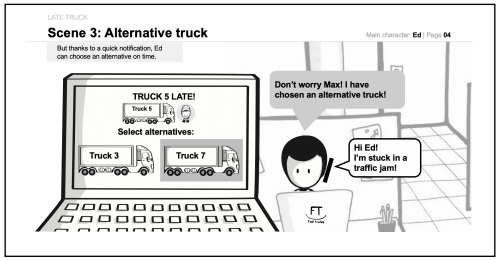
Paper



Role Play/Acting



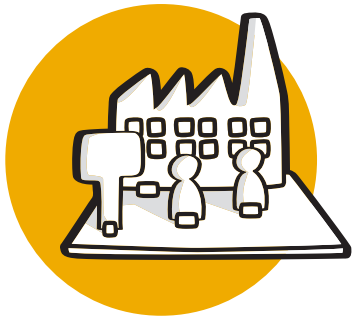




Storyboard

# Concept Storyboarding

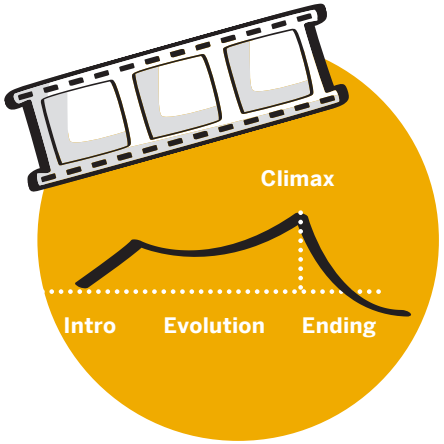
Template | Instructions | Example | Protocol



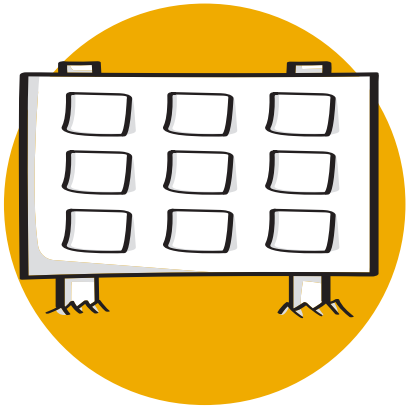
Concept storyboarding is prototyping an idea in the form of a story. Storyboarding helps your team to communicate and validate the proposed idea.

# Concept Storyboarding

Create your Story to get feedback and validate with your end-users




**Plan Your Ideal Story**



**Build Your Storyboard**

**Document it for Testing**

# Concept Storyboarding Instructions

Duration

45-60 minutes

Number of Participants

3-5 participants

## Why & What

Concept storyboarding is an effective way to prototype an idea in the form of a story. Storyboarding helps your team to communicate as well as validate the proposed idea.

Scenes

https://experience.sap.com/designservices/approach/scenes

## How to use it

1. Define your storyline

Take your experience map as a reference for a quick start. What is the biggest problem you are trying to solve? And how does your idea resolve that problem in an ideal world?

Plot points of your storyline:
- . Introduction
- . Problem
- . Evolution
- . Climax
- . Ending

2. Build your storyboard with Scenes, one of the AppHaus tools.

Customize Scenes illustrations with the special markers.

Humanize your characters by drawing their faces expressions.

3. Document your story

Capture with video and pictures each Scene of your story. Put all Scenes together and validate your storyboard.

## Tips & Tricks

Once you have aligned with your team on a storyline, build the different Scenes simultaneously in pairs to speed up the creation of the storyboard.

Use the caption markers to describe what happens in each Scene. This will make it easier for others to understand your story after the workshop.

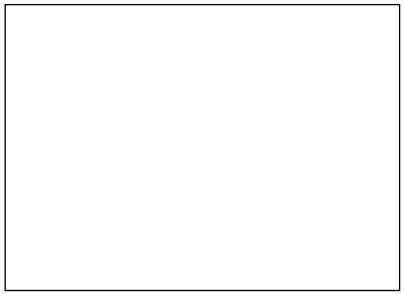
Take a picture of each of the Scenes on a white background for good documentation.

Record the presentation with your phone on video, to capture relevant details.

We recommend using Scenes, with predefined illustrations.

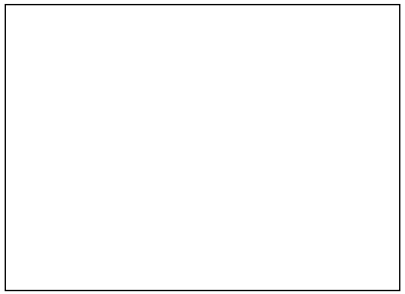
# Concept Storyboarding Template

Scene 1: Title



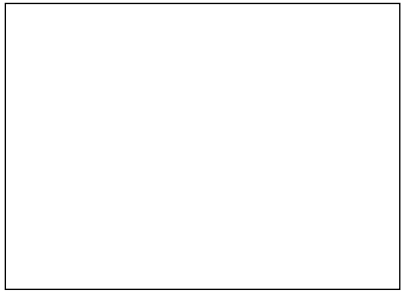
*Describe what's happening on the screen above.*

Scene 2: Title



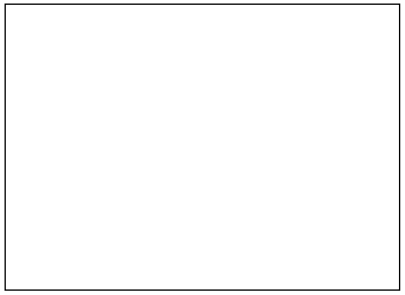
*Describe what's happening on the screen above.*

Scene 3: Title



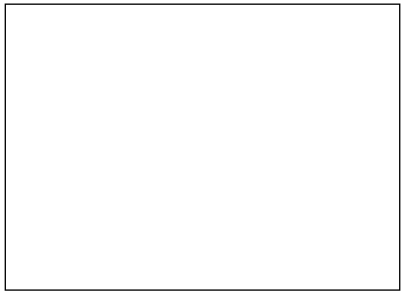
*Describe what's happening on the screen above.*

Scene 4: Title



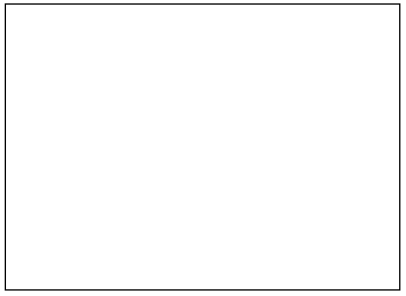
*Describe what's happening on the screen above.*

Scene 5: Title



*Describe what's happening on the screen above.*

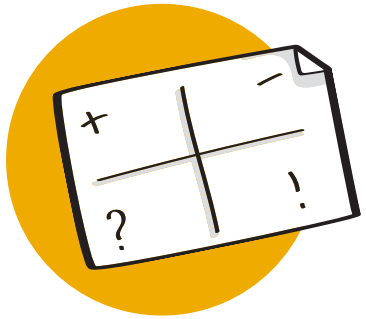
Scene 6: Title



*Describe what's happening on the screen above.*

# Feedback Grid

Template | Instructions | Example | Protocol

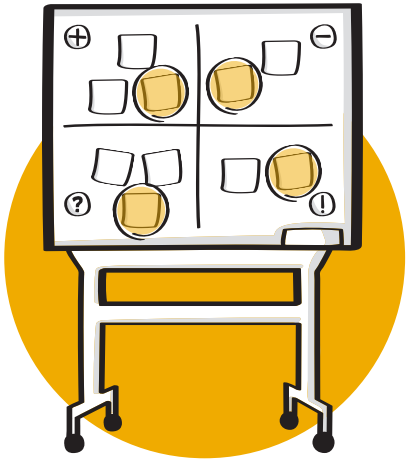



Get early feedback from users, stakeholders, experts and have a structured approach to derive learnings and insights.

# Feedback Grid

Capture feedback in a structured manner

Get feedback from your users



Capture all feedback on the grid

Incorporate the feedback

# Feedback Grid Instructions

Duration

**15-30 minutes**

Number of Participants

**3-5 participants**

## Why & What

Feedback grid to structure feedback & learnings.

Get early feedback from users, stakeholders, experts.

Fail early.

“Failure is simply the opportunity to begin again, this time more intelligently.” – Henry Ford.

## How to use it

1. Capture your impressions right after the feedback session.

2. As a team, discuss the reactions you received. Take notes on sticky notes. Sort and cluster the feedback according to the feedback grid:

- What was positively received

- What concerns came up

- What new ideas did you find

- What questions came up

3. Prioritize the feedback:

“What is the most important to make a success?”

Sort your notes and create an overview of which feedback you want to respond to.

4. Iterate your prototype by incorporating valuable feedback into your concept.

## Tips & Tricks

Let the prototype speaks …
… but know what you want to learn.

Do not defend your idea.

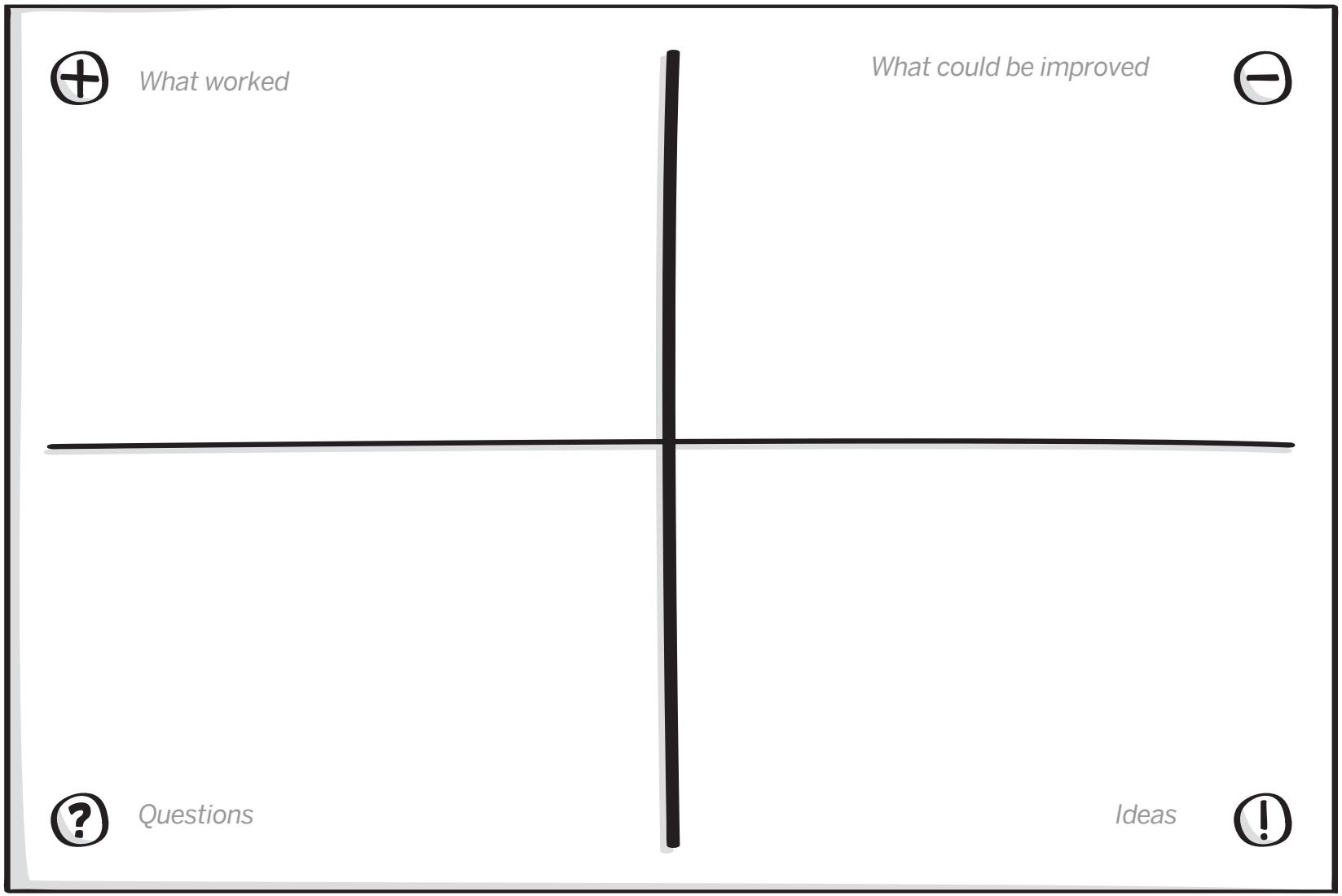
Be opened minded.

Do not fall in love with your prototype.

Do not make the prototype too finished and perfect.

Be thankful.

# Feedback Grid Template



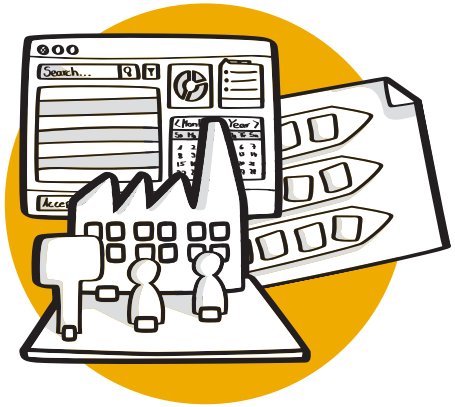
# Investor Pitch

Template | Instructions | Example | Protocol

Create a compelling story of your solution and the problem it solves that can be presented to management and key stakeholders for sign-off. A summary used to quickly and simply define a new solution and its value proposition short enough to be delivered in an investor pitch session.

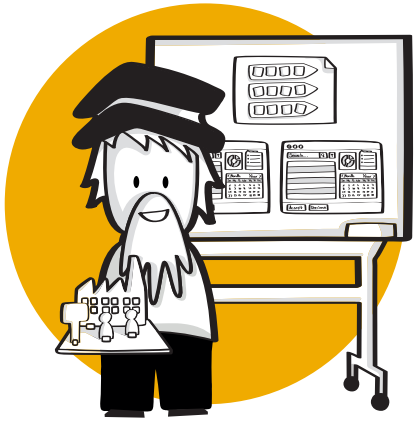
# Investor Pitch

Create a compelling description of your solution and the problem it solves



**Reflect on your current solution**
(through the 2D Journey map, Scenes 3D, Paper screens, Value proposition)

**Synthesize in one sentence**



**Pitch your solution to the investors**
(using the 2D Journey map, Scenes 3D, Paper screens, Value proposition)

# Investor Pitch Instructions

Duration

**15-30 minutes**

Number of Participants

**3-5 participants**

## Why & What

Create a compelling description of your solution and the problem it solves.

A summary used to quickly and simply define a new solution and its value proposition short enough to be delivered in an investor pitch session.

## How to use it

1. Reflect on your current solution by putting together the 2D Journey map, Scenes 3D, Paper screens, Value proposition.

2. To help you to prepare your Pitch, synthesize your current solution in one sentence using the description of the:

. Customer
. Need
. Solution
. Market Category
. Key benefit
. Competition
. Unique differentiator

3. Rehearse the pitch presentation using the 2D Journey map, Scenes 3D, Paper screens, Value proposition.

## Tips & Tricks

If the participants cannot hold the pitch in front of the high management, ask to 2 participants' colleagues outside the workshop to play the investor roles for the session.

# Investor Pitch Template

*Prepare your Investor Pitch*

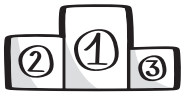
**For** .....  *(Customer)*

**Who** .....  *(Need)*

*(Solution Name)* .....

**is a** .....  *(Market Category)*

**that** .....  *(Key Benefit)*

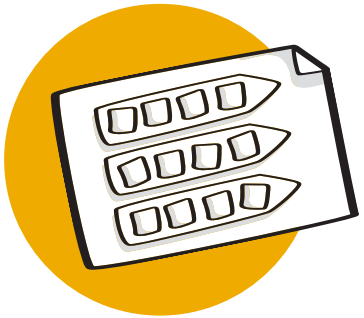
**unlike** .....  *(Competition)*

**the solution** .....  *(Unique differentiator)*

# Use Case

## Blueprint Diagram

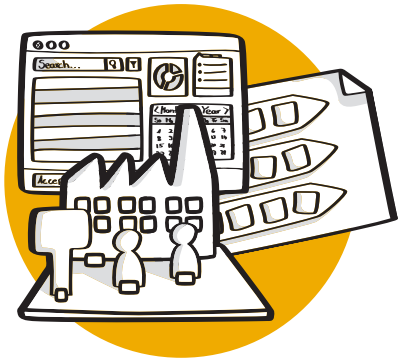
Template | Instructions | Example | Protocol



Pivot from Design Thinking to Architectural Thinking by mapping user actions to architectural requirements such as applications, data and specific technical features.

# Use-Case Blueprint Diagram

Pivot from Design Thinking to Architectural Thinking

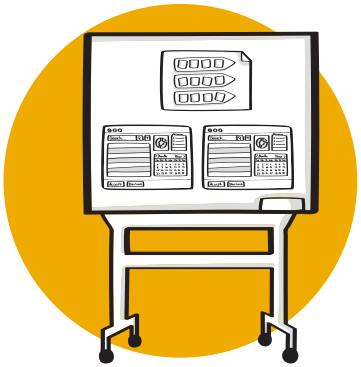


## User Story

Visualize and describe a specific use-case/ scenario of a specific persona with a storyboard and corresponding user actions.

## Data & Applications

Identify which data, what applications are required for each user-action along the storyboard.



## Technical capabilities

Identify technical capabilities that are required for each user-action along the storyboard.

# Use-Case Blueprint Diagram

## Instructions

Duration

**approx. 30-60 minutes**

Input

- **Statement of Architecture Work**
- **Stakeholder Matrix**
- **Solution Context**

### Why & What

The purpose of the Use-Case Blueprint Diagram is to pivot from Design Thinking to Architectural Thinking. User-centric actions are mapped to technical aspects of the architecture, such as data, systems, and technical capabilities.

The Use-Case Blueprint diagram, with its user centricity, is the bridge to creative thinking and the Design Thinking methodology. In case you want to add more user-centricity and usability focus to your architecture, you find additional work products in the Design Thinking Methodology (Innovation Toolkit by SAP AppHaus).

### How to use it

1. Create a storyboard by putting yourself in the shoes of a key user. Describe the desired objective or capability by disassembling the objective into individual scenes or scenarios for the key user.

2. Think about technical requirements that need to be met in order to realize the user actions: What data is required for the actions in the scenes? Which applications and IT systems are involved in the user actions?

3. Based on your experience, add technical capabilities that are required at a specific step or action, like IoT data ingestion, a chatbot or a workflow service, for example.

### Tips & Tricks

Taking the previously identified main-uses and key-objectives of the users from the Solution Context as input, put yourself in the shoes of one or several key-users. Create one Use-Case Blueprint Diagram per user and per key objective of this user. As this can clearly be a lot of work, you might want to choose the most valuable objective and user combination and will not do all possible combinations.

You will re-use information form the Use-Case Blueprint Diagram at later steps in the technical architecture domain. For example, the data identified, helps to understand the information flow, which is represented in the Solution Concept and the Solution Realization Diagram. Also, the identified data serves as input for the Conceptual Data Diagram.

The identified applications are used for the creation of Baseline Solution Architecture, the Solution Concept and Solution Realization Diagram.

The identified technical capabilities support the choice of Solution Building Blocks.

AppHaus

# Use-Case Blueprint Diagram

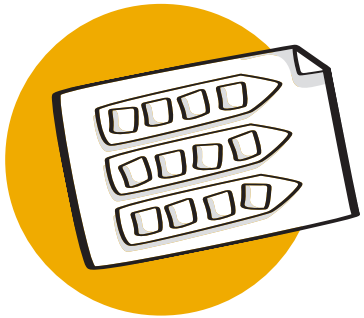
## Template & Example

| | Template: | **Scene 01**: Business developer wants to request budget for new project idea. | **Scene 02:** … |
|---|---|---|---|
| **Target Scenario**<br>How does the user work with the target solution? | *Describe the context / scene of the user* | | |
| **Actions**<br>What actions does the user perform to achieve desired results? | *Describe what the user is doing* | ▪ Open “Request Application”<br>▪ Enter project details<br>▪ … | ▪ … |
| **Required Data**<br>What data is required for the action? | *Describe the data, its attributes, and data type* | ▪ Project name + description<br>▪ Project goals + KPIs<br>▪ … | ▪ … |
| **Systems & Applications**<br>Which technical systems or applications are accessed for the action? | *List systems and applications required to perform the action* | ▪ “Request Application”<br>▪ “Central Repository”<br>▪ … | ▪ … |
| **Technical Cabability**<br>Which technical capabilties are required to perform the action? | *Highlight required technical capabilities to perform the action* | ▪ Check integrity of project details (e.g. time vs. budget, …)<br>▪ Automatically consider budget limits<br>▪ … | ▪ … |

# Baseline

## Solution Architecture

Instructions | Template | Example



Describe existing applications and IT components relevant for the use-case (request of architecture work)

# Baseline Solution Architecture

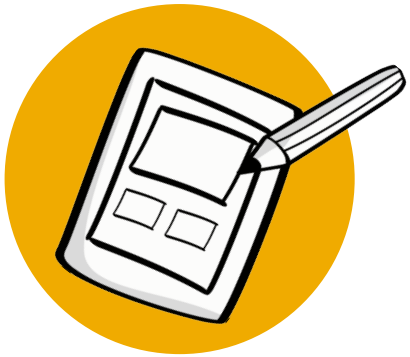
Describe existing components relevant for your architecture

## List current applications

Identify existing applications and IT components that are relevant for the request of architecture work (use-case).

## Understand dependencies

Identify the functional relationship between the identified components.



## Visualize with diagram

Draw the existing applications and IT components including their dependencies and relationships. Add users and roles.

# Baseline Solution Architecture

## Instructions

Duration

**approx. 30-60 minutes**

Input

- **Stakeholder Matrix**
- **Statement of Architecture Work**
- **Use-Case Blueprint Diagram**

### Why & What

Describes applications, software components, and functional components currently in use and relevant for the use-case/ request of architectural work.

The reason for creating the Baseline Solution Architecture is to identify building blocks that can be re-used for designing the architecture or identify components which need to be integrated via interfaces like APIs, Events or Data replication, for example.

It also shows the scope of change initiatives resulting from the aspired solution to the current IT landscape.

### How to use it

1. List all the current applications and IT components (Building Blocks) that are relevant for the use-case (scenario).

2. Understand the dependencies between the identified IT components (functional dependency in terms of request-response and/ or information flow)

3. Add users and roles currently interacting with the IT components. It is likely, that some users and roles are identical to the ones identified for the Solution Context Diagram and are also part of the Stakeholder Matrix.

### Tips & Tricks

The Baseline Solution Architecture is important, because it shows what IT components, or Building Blocks, are already in place and can be considered when creating the Solution Concept and Solution Realization Diagram in the next steps.

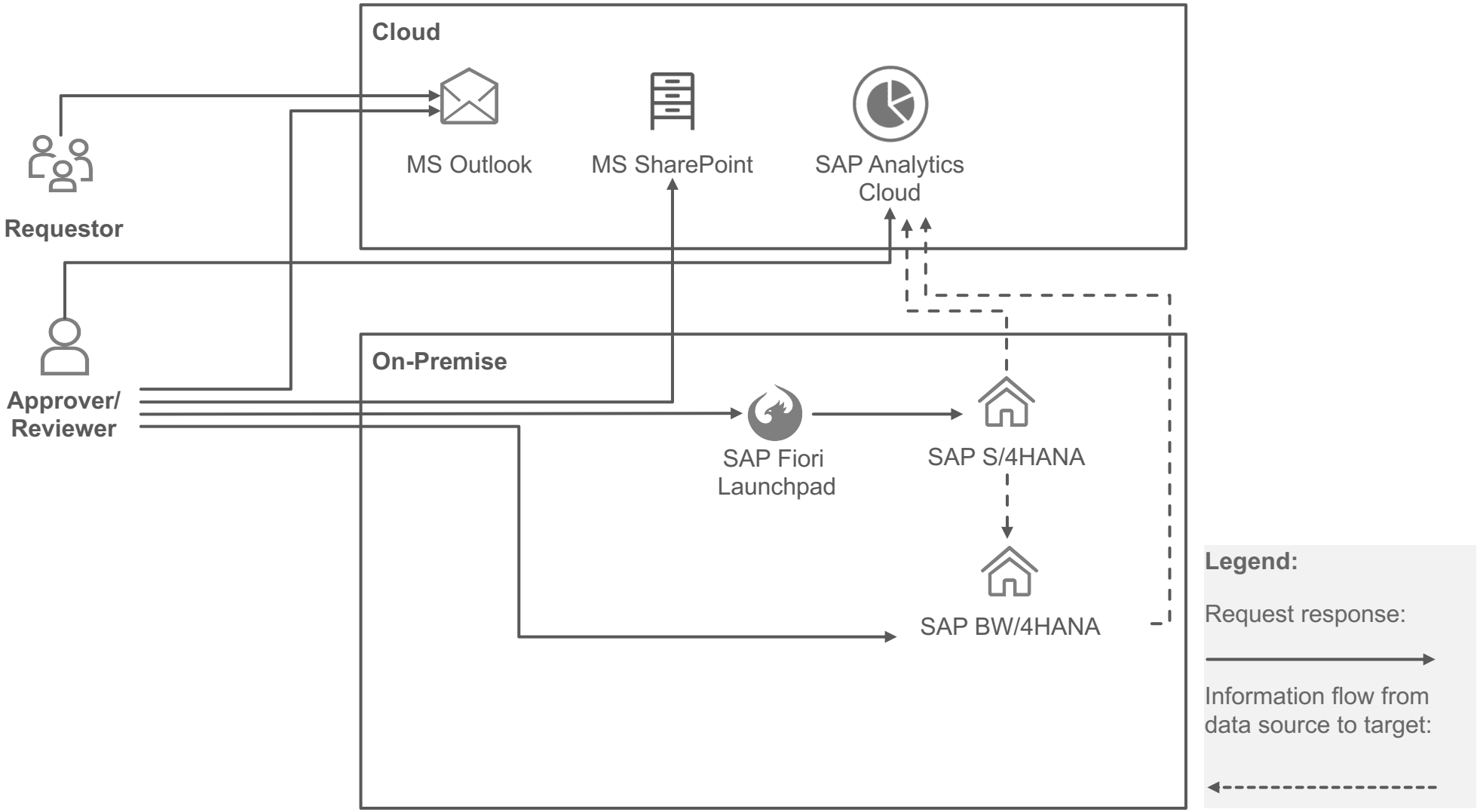
Considering the Statement of Architecture Work, it is worth emphasizing that not the complete corporate IT landscape needs to be represented in the diagram. Instead, focus on the areas of interest, the areas that are in scope of your architecture engagement/ request of architecture work.

The Use-Case Blueprint Diagram can be used to identify existing IT systems relevant for the architecture.

Consult IT and business stakeholders knowing the context of your use-case (the request of architecture work) to understand which existing IT components should be considered.

# Baseline Solution Architecture

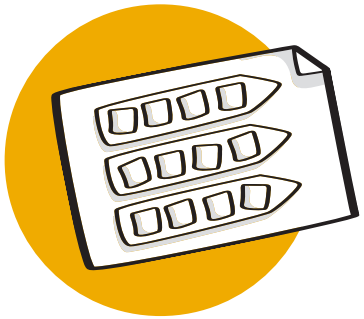
## Example



# Solution

## Concept Diagram

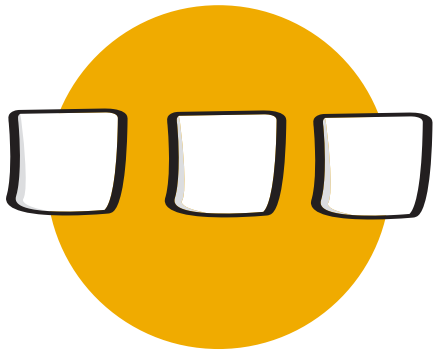
Instructions | Template | Example



A high-level representation of the aspired solution via Architecture Building Blocks

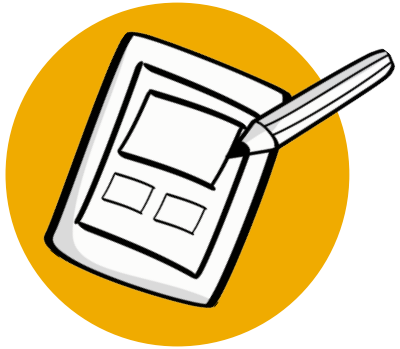
# Solution Concept Diagram

A high-level representation of the aspired solution via Architecture Building Blocks (ABBs)



## Identify architecture building blocks

Identify high-level building blocks to satisfy the key objectives of the aspired solution. Map all business capabilities from the Solution Context Diagram to one or more ABB(s).



## Draw Diagram

Draw the architecture building blocks including their relationships and dependencies.

## Add Business Units

Add organizational units, functions and users or roles including relationships to the architecture building blocks.

# Solution Concept Diagram

## Instructions

Duration

**approx. 30-60 minutes**

Input

**- Solution Context Diagram**
**- Baseline Solution Architecture**

### Why & What

Provides a high-level representation of your solution that is envisioned in order to meet the requirements of your architecture engagement. You can consider the solution concept as a “pencil sketch” of your expected solution based in architecture building blocks (ABBs).

The purpose of the Solution Concept is to provide a high-level representation of your solution. You can think of the Solution Concept as a first pencil-sketch showing Architecture Building Blocks of your solution.

### How to use it

1. Identify high-level architectural building blocks that are needed to satisfy the key objectives and business capabilities as described in the Solution Context Diagram. Translate the business-oriented capabilities describing the aspired solution in the Solution Context, to respective Architecture Building Blocks.

2. Outline the relationships between the identified building blocks. Relationships can be of type request-response or information flow.

3. Add users, roles, and organizational units including their relationship to the architecture building blocks.

### Tips & Tricks

The Solution Concept is a transition step between the Solution Context and the Solution Realization Diagram. It is a macro version of your architecture, mainly comprised of vendor neutral Architecture Building Blocks.

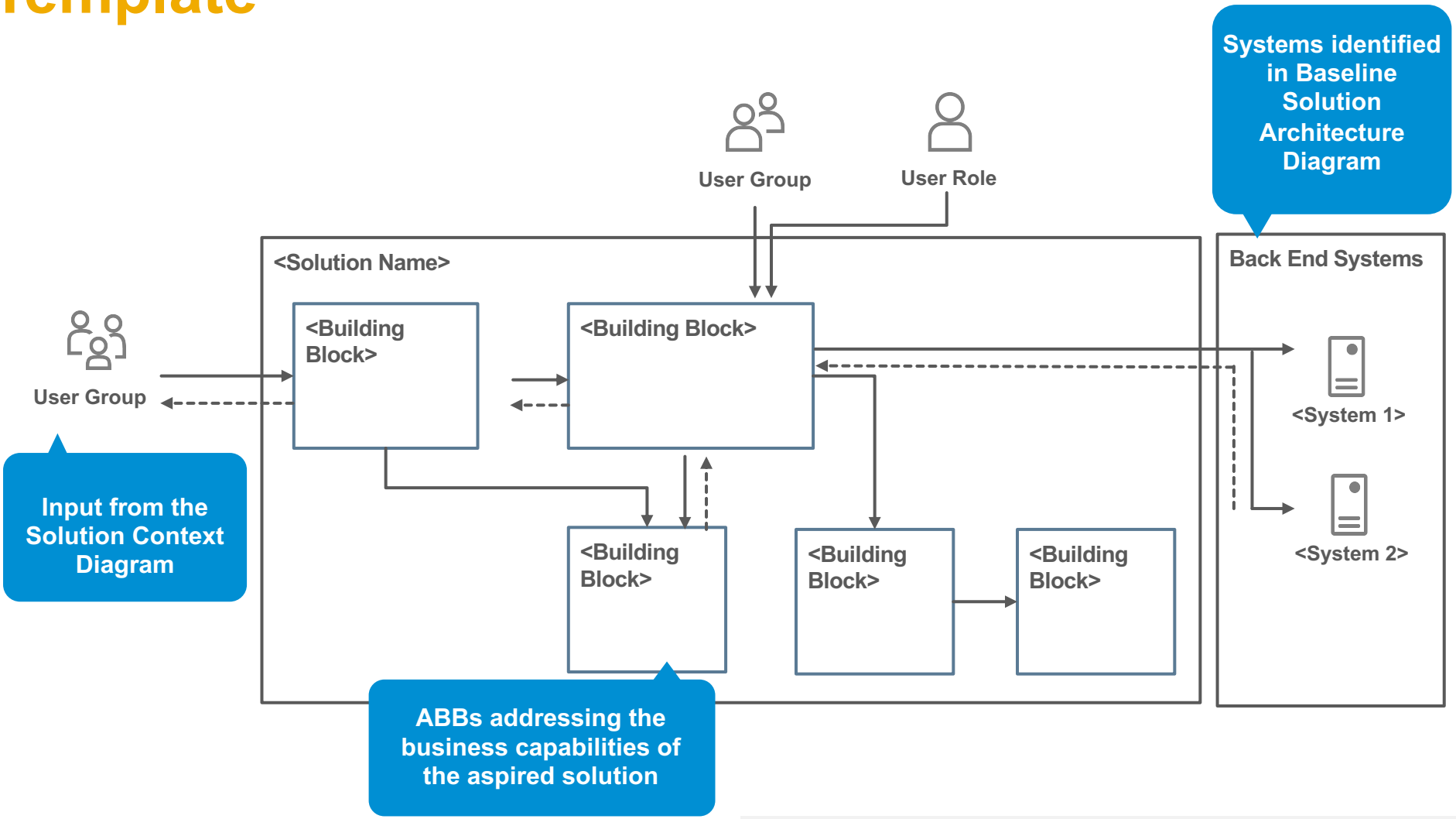
By mapping the capabilities of the aspired solution to architecture building blocks, it might be the case that one business capability directly translates to one ABB. Also, it can be the case that one business functionality translates to many ABBs related to each other.

A good quality check of the Solution Concept Diagram is to check if all required business capabilities are addressed with corresponding ABBs.

AppHaus

# Solution Concept Diagram

## Template



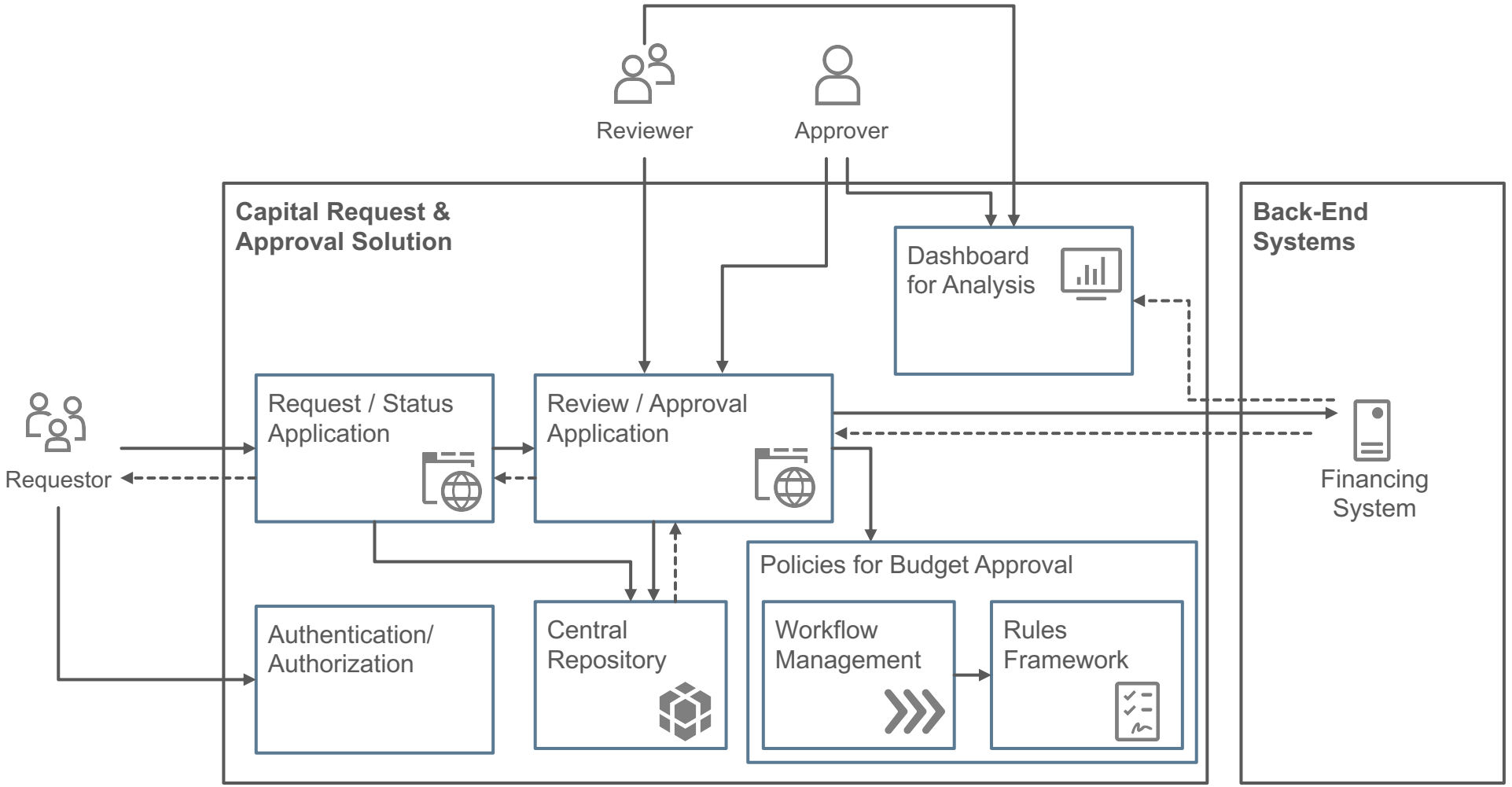

Legend:

Request response:

Information flow from data source to target:

# Solution Concept Diagram

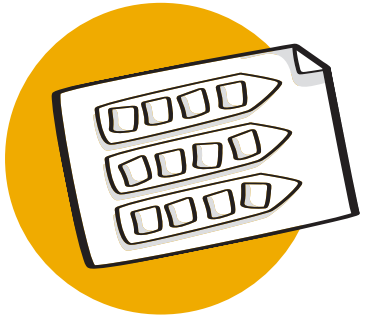
## Example



# Solution

## Realization Diagram

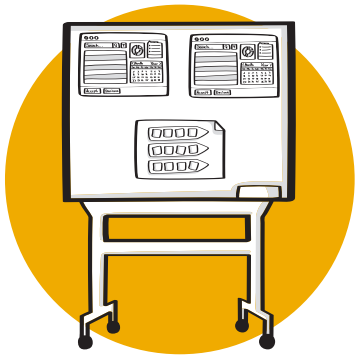
Instructions | Template | Example



A technical description of the aspired solution via Solution Building Blocks

# Solution Realization Diagram

A technical description of the aspired solution via Solution Building Blocks

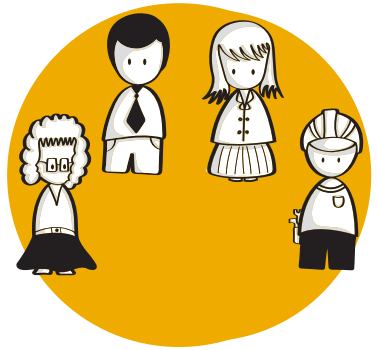


## Identify Solution Building Blocks

Perform a Fit-Gap Analysis of the Baseline Solution Architecture. Map vendor specific solution building blocks* to architecture building blocks.

## Identify functional dependencies

Identify dependencies and the data flow between the solution building blocks.



## Add users

Add users & roles including their relationship to the solution building blocks.

* based on experience, reference architectures, solution descriptions.

# Solution Realization Diagram

## Instructions

Duration

**approx. 60-120 minutes**

Input

- **Baseline Solution Architecture**
- **Solution Concept Diagram**
- **SAP Discovery Center (vendor specific information sources)**
- **SAP API Business Hub**

### Why & What

Describes your target architecture by breaking the aspired solution into functional components based on solution building blocks (SBB). The purpose of the Solution Realization Diagram is to add technical details so that it can be used for the implementation phase of the aspired solution.

Solution Building Blocks are product or vendor-aware and can be either developed or procured. With the Solution Building Blocks you outline which products, like cloud services, are used, or need to be developed, in order to realize and implement the aspired solution/ its architecture building blocks.

### How to use it

1. Identify Solution Building Blocks that realize the previously identified Architecture Building Blocks. Map the Architecture Building Blocks of the Solution Concept to one ore more Solution Building Blocks. Sources for Solution Building Blocks are the Baseline Solution Architecture (perform a fit-gap-analysis) and vendor specific information sources like the SAP Discovery Center for SAP BTP Solution Building Blocks.

2. Outline the dependencies between the identified SBBs (functional dependency in terms of request-response and/ or information flow)

3. Add users and roles currently interacting with the SBBs.

4. Check the Solution Realization Diagram for completeness, by verifying that each ABB from the Solution Concept is addressed with SBBs.

### Tips & Tricks

There are basically two sources of Solution Building Blocks:

(1) Baseline Solution Architecture. Based on the Baseline Solution Architecture you perform a Fit-Gap-Analysis, identifying Solution Building Blocks that are already in place and can either be completely or partly re-used in your Solution Realization.

(1) SAP Business Technology Platform services (e.g., via SAP Discovery Center).

The mapping of Solution Building Blocks to Architecture Building Blocks depends on the experience and knowledge in the team (experience-based mapping).

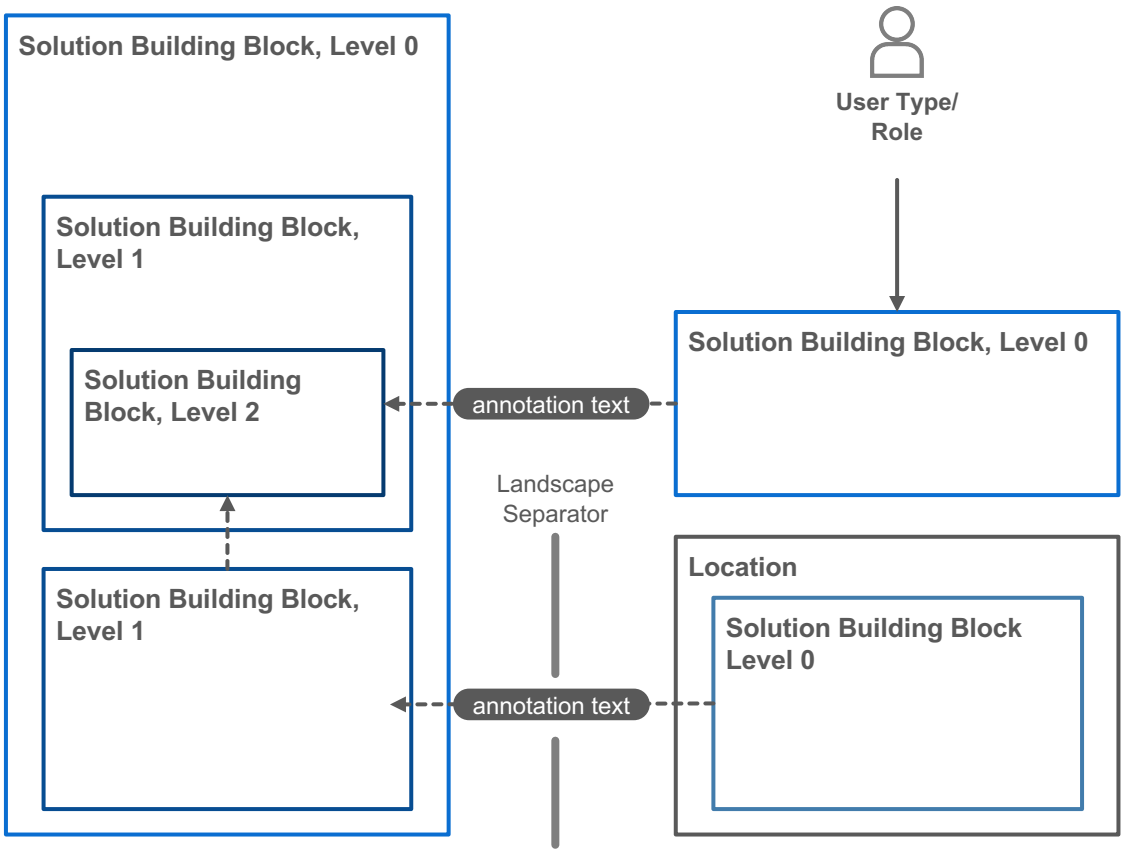
Publicly available information/ documentation can be consulted about the Solution Building Blocks for doing the mapping. Reference architectures solving a similar problem or addressing parts of your request of architecture work are also good sources for doing the mapping.

Use BTP service icons to draw the diagram.

AppHaus

# Solution Realization Diagram

## Template



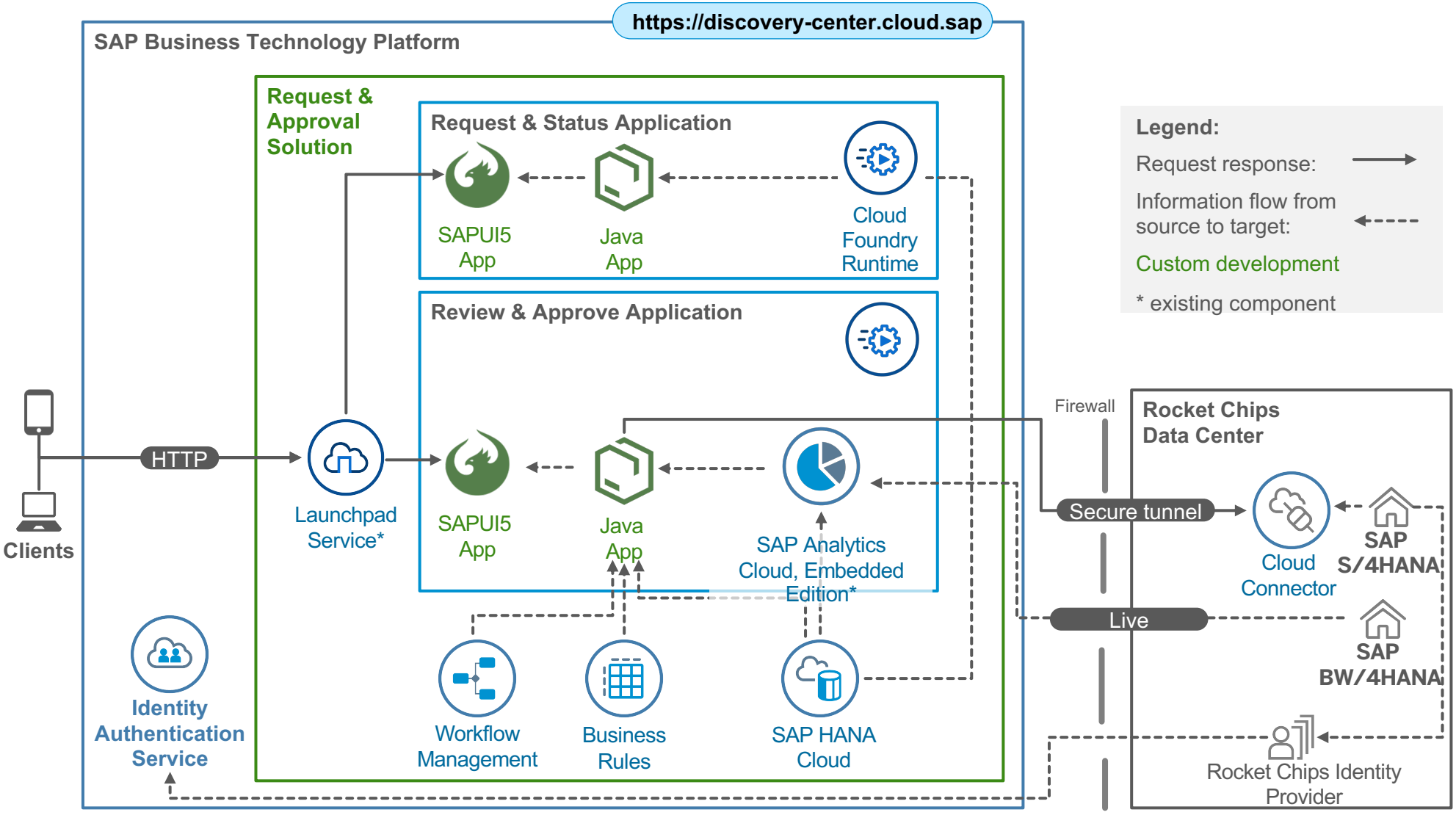

**Legend:**

Request response: ⟶

Information flow from source to target: ⇠ - - - -

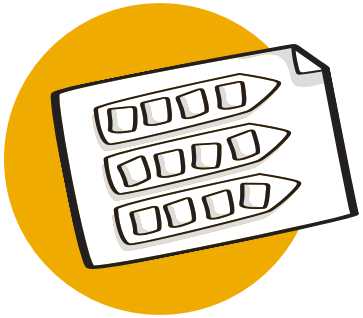
# Solution Realization Diagram

## Example



# Architectural Decisions

Instructions | Template | Example



Document your architectural decisions and reasoning

# Architectural Decisions

Document your architectural decisions and reasoning

## Keep track of alternatives

Identify discussions for different architectural alternatives. What alternatives do you consider?

## Document your thoughts

Document the reasoning for the decision taken. Why did you make this decision?

## Keep updating

Continuously update your architectural decisions throughout the architecture development process.

# Architectural Decisions

## Instructions

Duration

**ongoing**

Input

- **Solution Concept Diagram**
- **Solution Realization Diagram**
- **Software Distribution Diagram**
- **Environments & Location Diagram**

### Why & What

Document the architectural decisions you have chosen for realizing the target architecture.

The purpose is to document your architectural discussions and outline the reason for the decisions taken.

### How to use it

1. Every time you consider two or more alternatives in the architecture, you should remind yourself to write down the reasoning behind your decision for one of the alternatives.

2. Note the reason why you have chosen the alternative. Was it due to corporate strategy? Maybe there is a corresponding architecture principle that you considered during the decision? You also can include a SWOT analysis to explain the reasoning of your decision.

### Tips & Tricks

Throughout the creation of your architecture, you make different decisions. For example, you decide to use one solution building block of vendor A instead of another alternative from vendor B.

Or you decide to implement a required solution building block by your own, instead of purchasing a building block.

All these decisions are documented for later reference and show the reasoning behind your decision.

Documenting your architectural decisions is a recurring task and it is likely that you add decisions at later stages of your architecture development. For example, when you think about the deployment of your architecture in the context of the environments and location diagram.

# Architectural Decisions

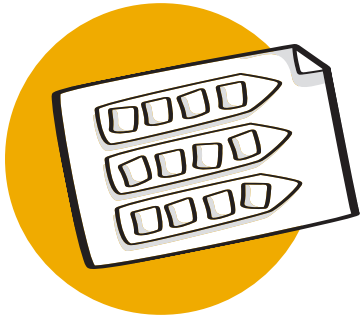
## Template & Example

| Ref | Decision | Reasoning |
|---|---|---|
| *<unique identifier>* | *<short description of the decision>* | *<details about the reason for the decision and possible alternatives that were considered>* |
| An_100 September-2020 | Provide embedded analytics to support profitability analysis of budget requests. | To offer an integrated user experience and integrated end-to-end experience, SAP Analytics Cloud, embedded edition is used and integrated into the “Budget Request & Review application”. |
| … | … | … |

# Conceptual

## Data Diagram

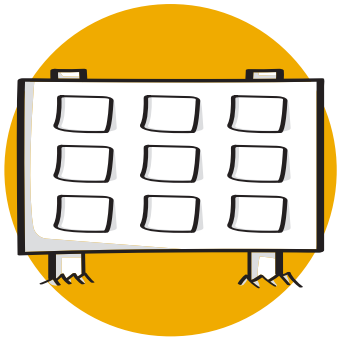
Instructions | Template | Example



Describes information objects processed by the architecture

# Conceptual Data Diagram

Describes information objects processed by the architecture.

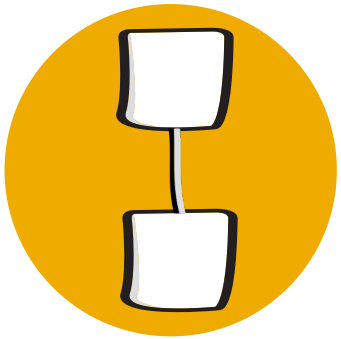


## Identify entities

Identify (business) entities the architecture needs to process.

## Define Attributes

Add attributes (properties) and data types to the entities.



## Define Relationships

Identify and draw relationships between the entities.

# Conceptual Data Diagram

## Instructions

Duration

**approx. 60-120 minutes**

Input

- **Use-Case Blueprint Diagram**
- **Solution Realization Diagram**

### Why & What

Describes entities (information objects) processed by the architecture.

The purpose of the conceptual data diagram is to outline the relationship between data and business entities of the aspired solution.

### How to use it

1. Identify information objects (entities) being processed by the Solution Building Blocks in the Solution Realization Diagram and understand the information flow between the SBBs.

2. Name the entities you have identified with an unambiguous name and add attributes or properties with respective data types to further describe the entities. For choosing data types, keep it simple and chose types like “number”, “string”, “date”, for example.

3. Think about the relationship between the different entities. Mostly, you will design an association between two entities and define a multiplicity. The association is represented by a solid line between two entities and is described with a verb.

### Tips & Tricks

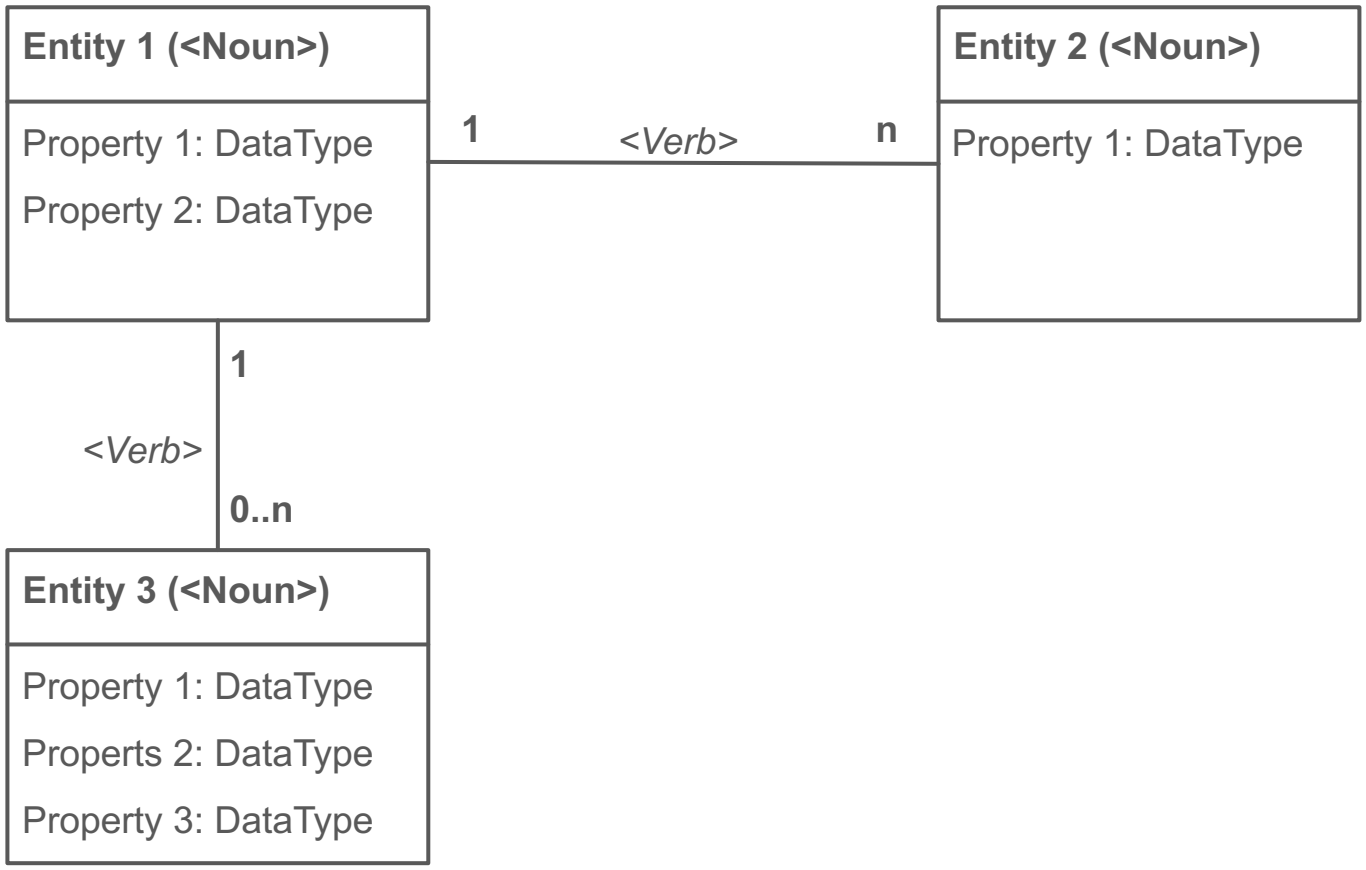
The Conceptional Data Diagram can be treated like an Entity-Relationship Diagram, outlining, and describing the information being processed by the aspired solution.

However, the conceptual data diagram is not a technically detailed data model description which you can directly map to a relational data model, for example. In fact, the information objects, or data entities, you are describing via the Conceptional Data Diagram can be stored and handled by different solution building blocks of the architecture. They might not necessarily end up in a relational database, maybe a graph store is suited better, or you get the information objects via an API or message queue, from a building block outside the scope of your architecture.

One source of input for creating the Conceptual Data Diagram is the Use-Case Blueprint Diagram. Another source of input for creating the diagram is the Solution Realization Diagram.

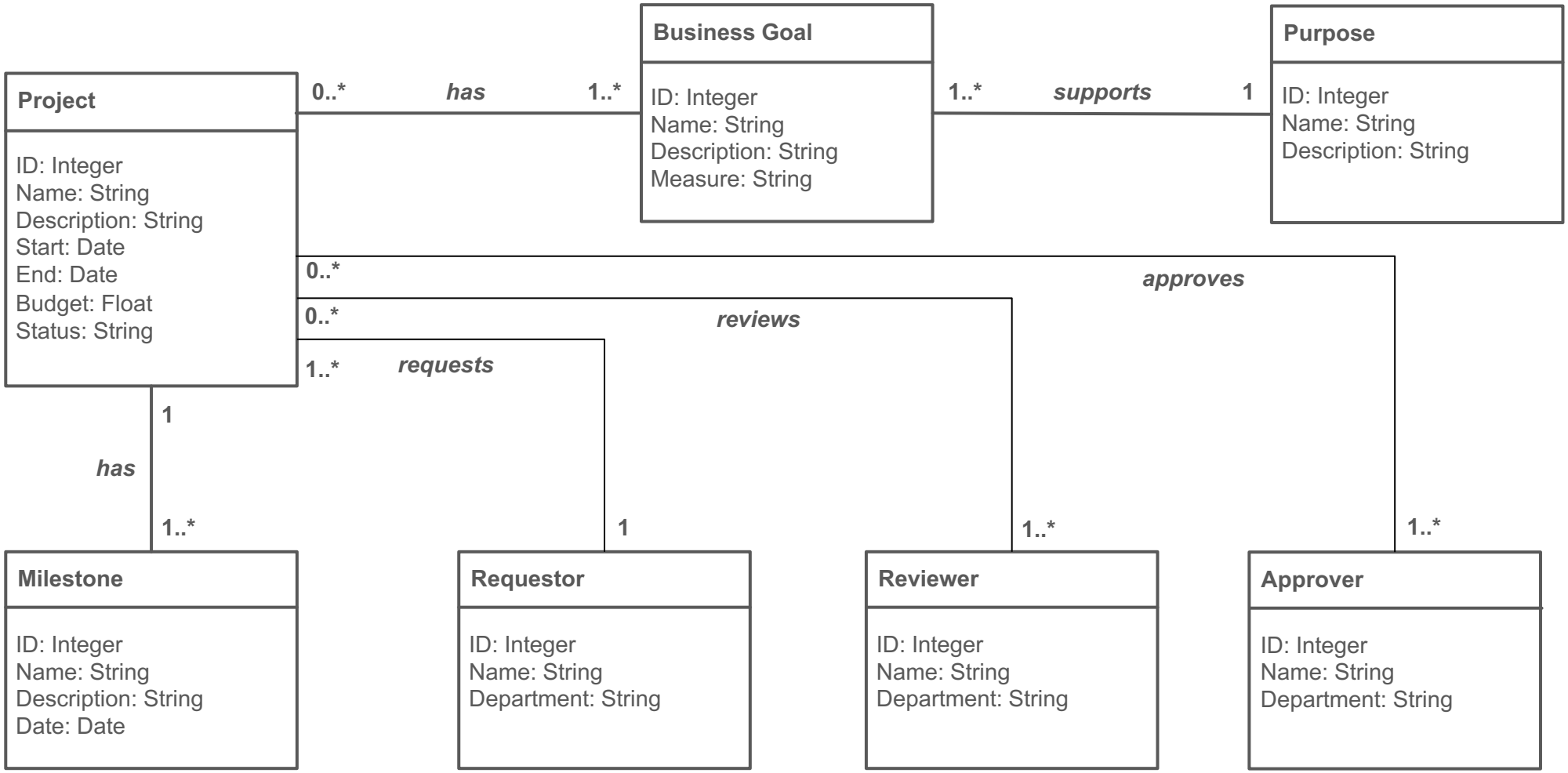
# Conceptual Data Diagram

## Template



# Conceptual Data Diagram

## Example



# Software

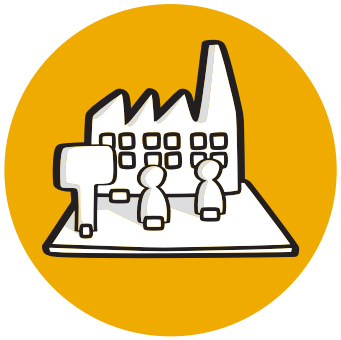
# Distribution Diagram

Instructions | Template | Example

Shows how architecture- and solution building blocks are distributed across the IT infrastructure

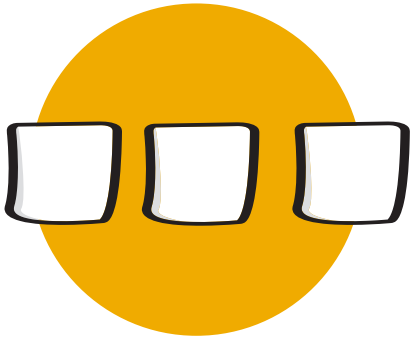
# Software Distribution Diagram

Shows how architecture- and solution building blocks are distributed across the IT infrastructure.



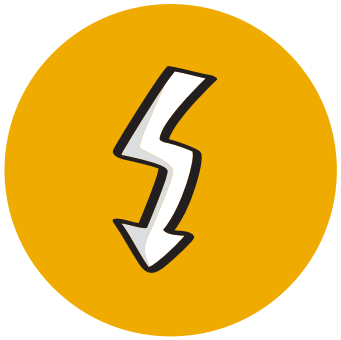
### Deployment Environments

Identify relevant hosting and deployment environments of your solution (e.g., cloud and on-premise).



### Map Building Blocks

Map the building blocks of your architecture to the hosting and deployment environments.



### Data Flow

Visualize the data flow/ response-request relationship between building blocks indicating network communication between different environments.

# Software Distribution Diagram

## Instructions

Duration

**approx. 30-60 minutes**

Input

- **Baseline Solution Architecture**
- **Solution Concept Diagram**
- **Solution Realization Diagram**

### Why & What

Shows how the aspired solution is structured and how the solution building blocks / architecture building blocks are distributed across the IT infrastructure.

The purpose of the Software Distribution Diagram is to enable a view of how the aspired solution is deployed and hosted.

### How to use it

1. Identify relevant hosting and deployment environments for the building blocks of the architecture. A classification between cloud and on-premise might be sufficient.

2. Map the building blocks of the architecture to the identified hosting and deployment environments.

3. Visualize the relationships of type request-response or information flow between the building blocks. This helps to understand potential latency considerations or network requirements .

### Tips & Tricks

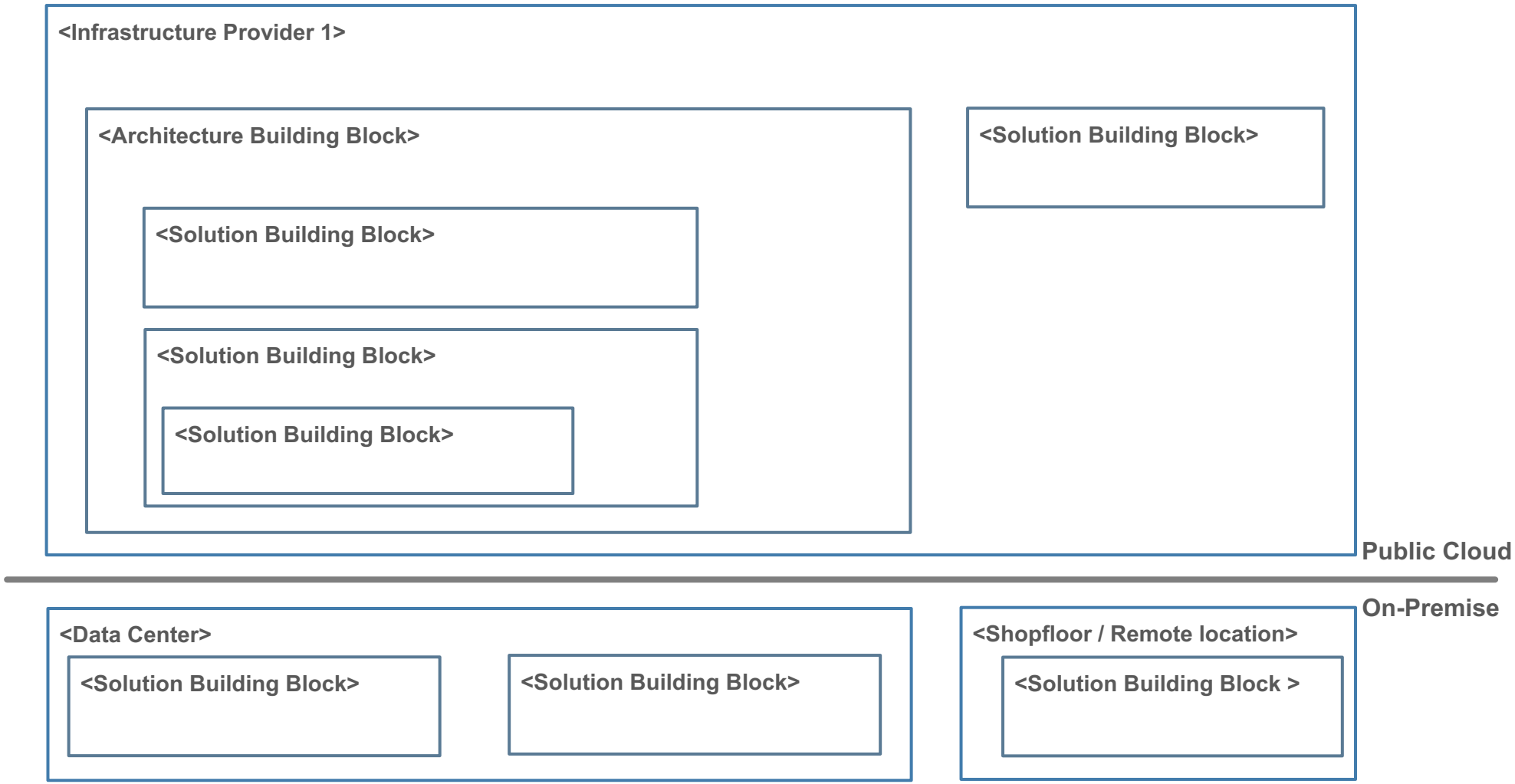
As a corporate IT infrastructure has typically grown beyond the four walls of a corporate data center, the purpose of the software distribution diagram is to show in which “landing zones” the different building blocks of the architecture are running. As today’s IT infrastructures are typically hybrid, you can use a general classification in On-Premise- and Cloud- landing zones. The On-Premise landing zone can be further categorized into specific data center, shopfloors or stores, for example. The cloud landing zone can be further categorized into IaaS provider or SaaS provider, for example.

For creating the Software Distribution Diagram, you need the list of building blocks making up your solution. Consult the Solution Concept Diagram to get the list of architecture building blocks and the Solution Realization Diagram, to get the list of solution building blocks.

AppHaus

# Software Distribution Diagram

## Template



AppHaus

# Software Distribution Diagram

## Example