



TRAILHEAD
TECHNOLOGY PARTNERS

‘Why Did They Do That?’

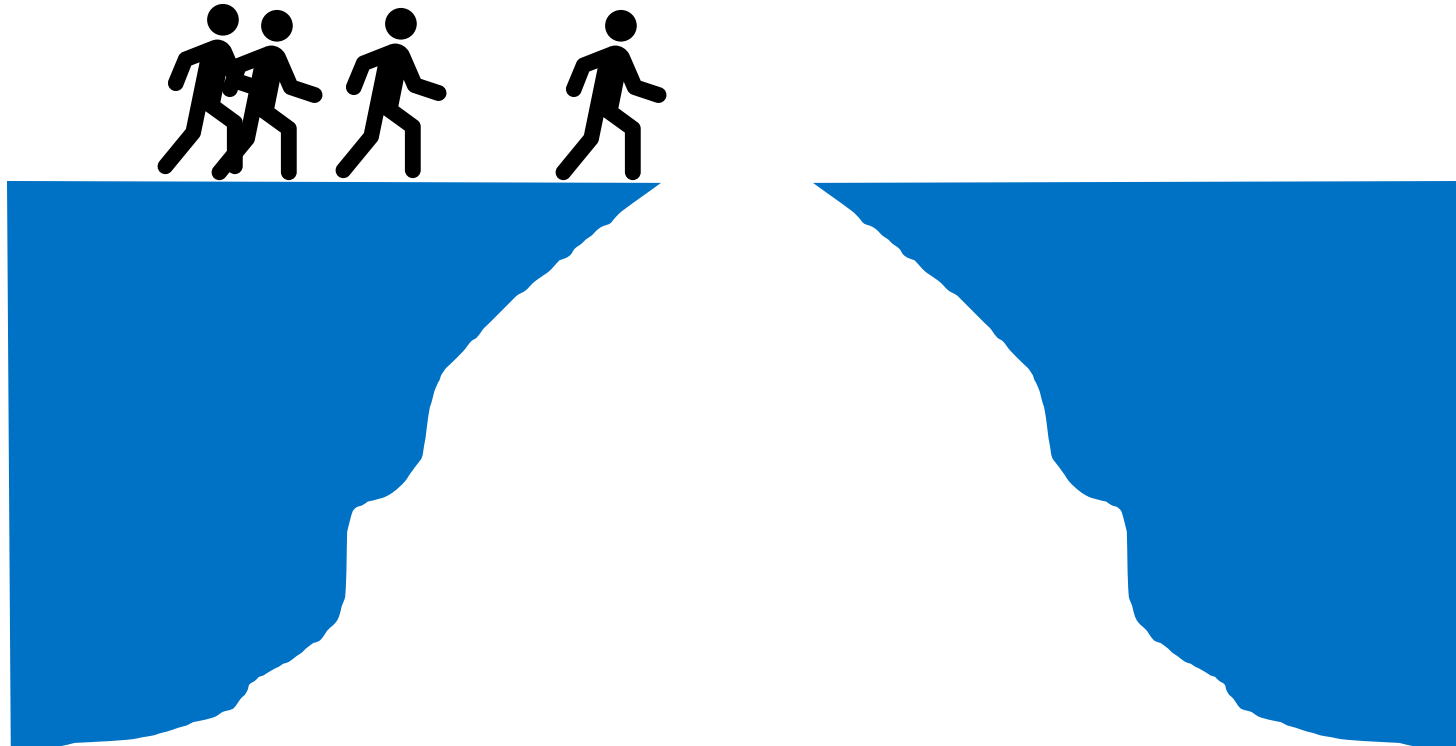
And Other Mysteries

Solved by ADRs

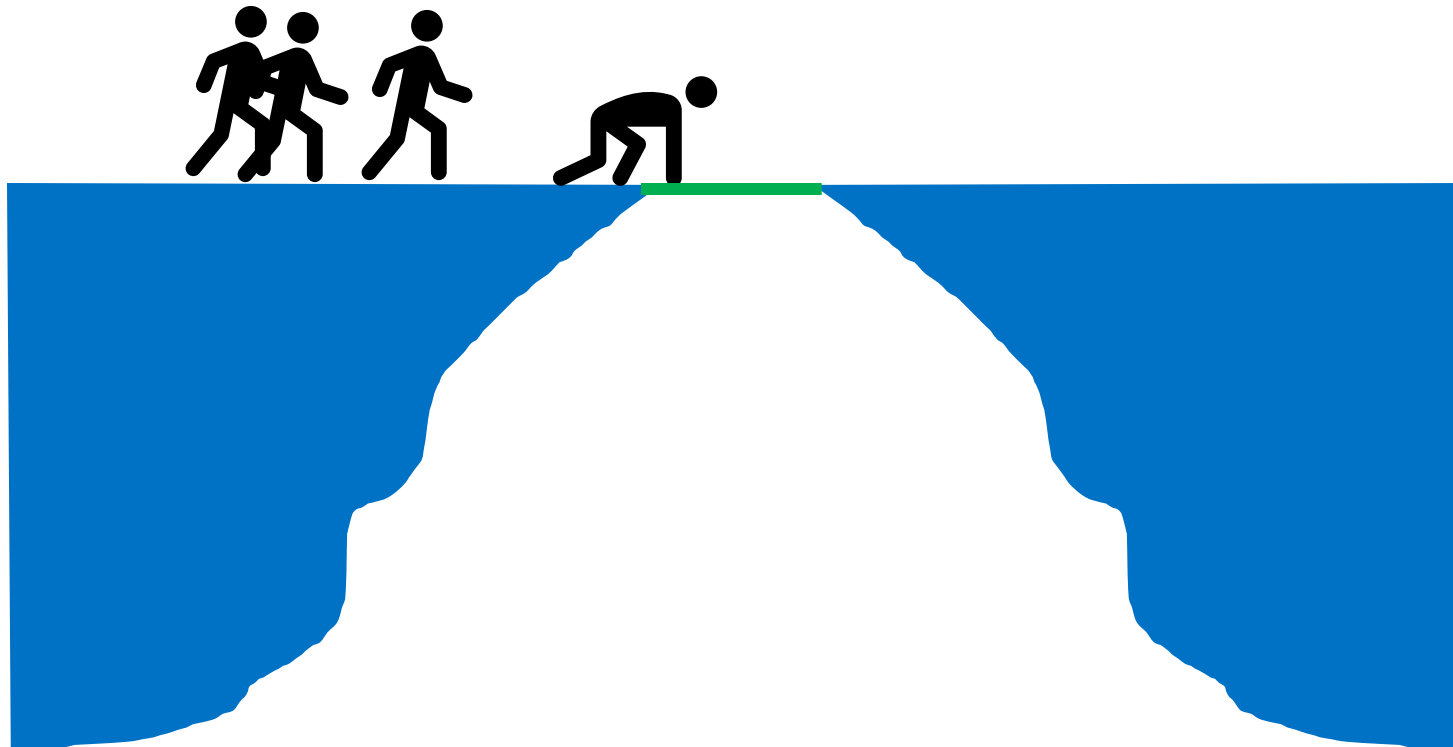


Jonathan “J.” Tower

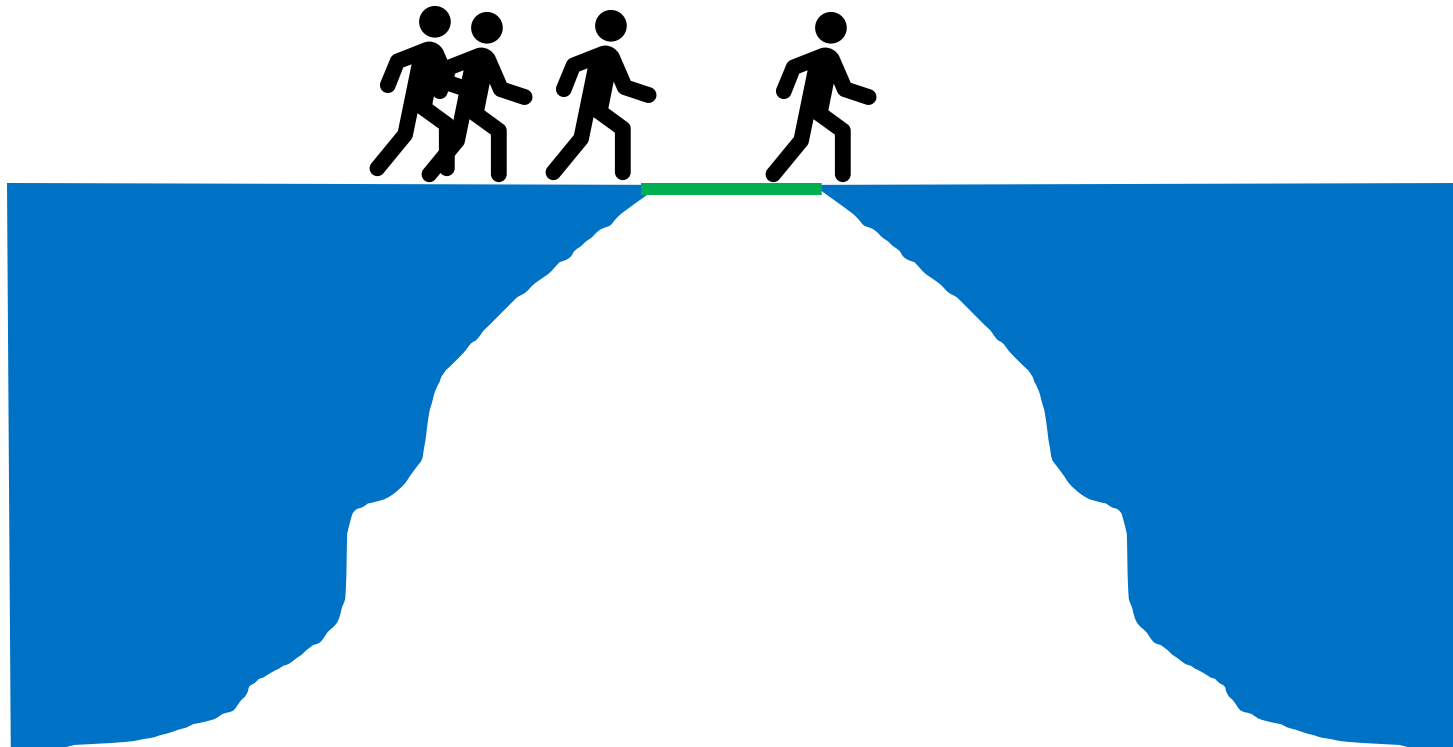
The Chasm of Forgotten Decisions



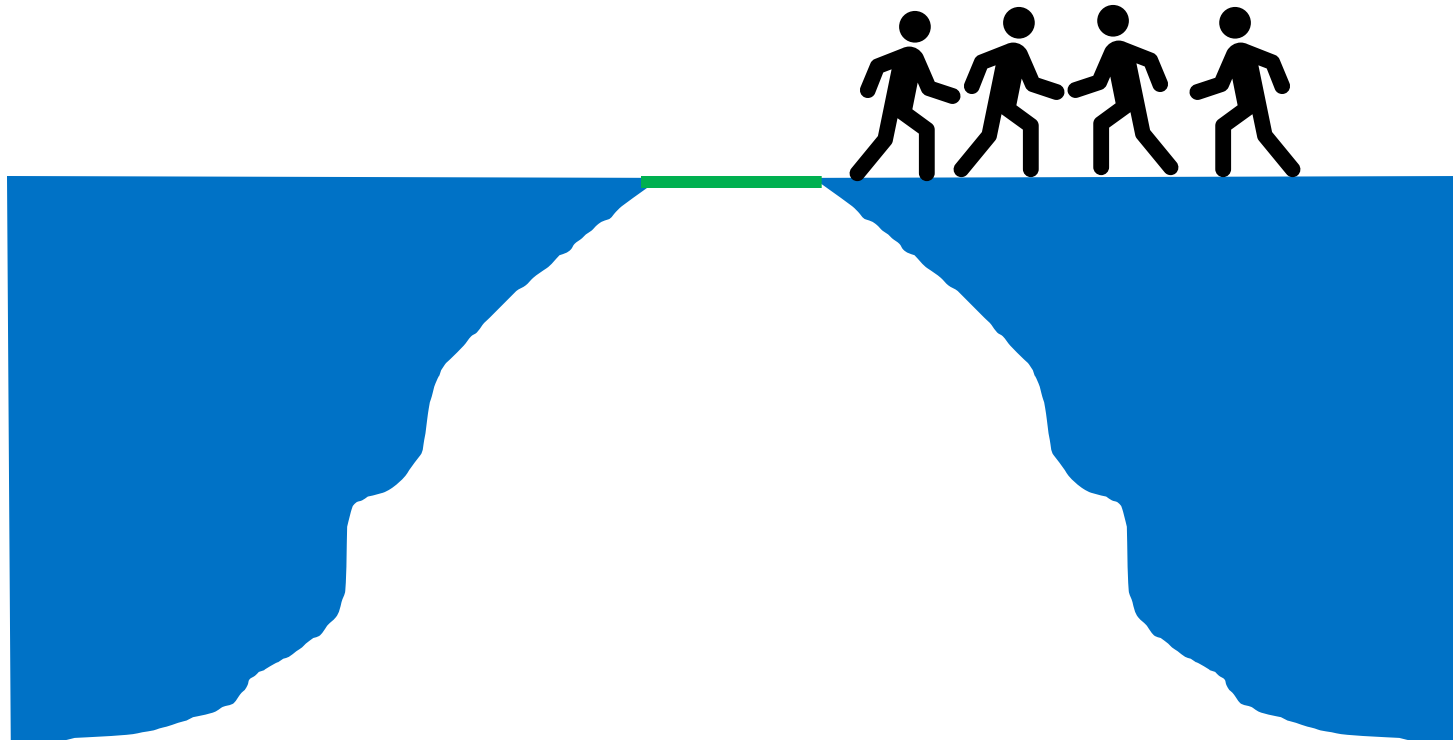
The Chasm of Forgotten Decisions



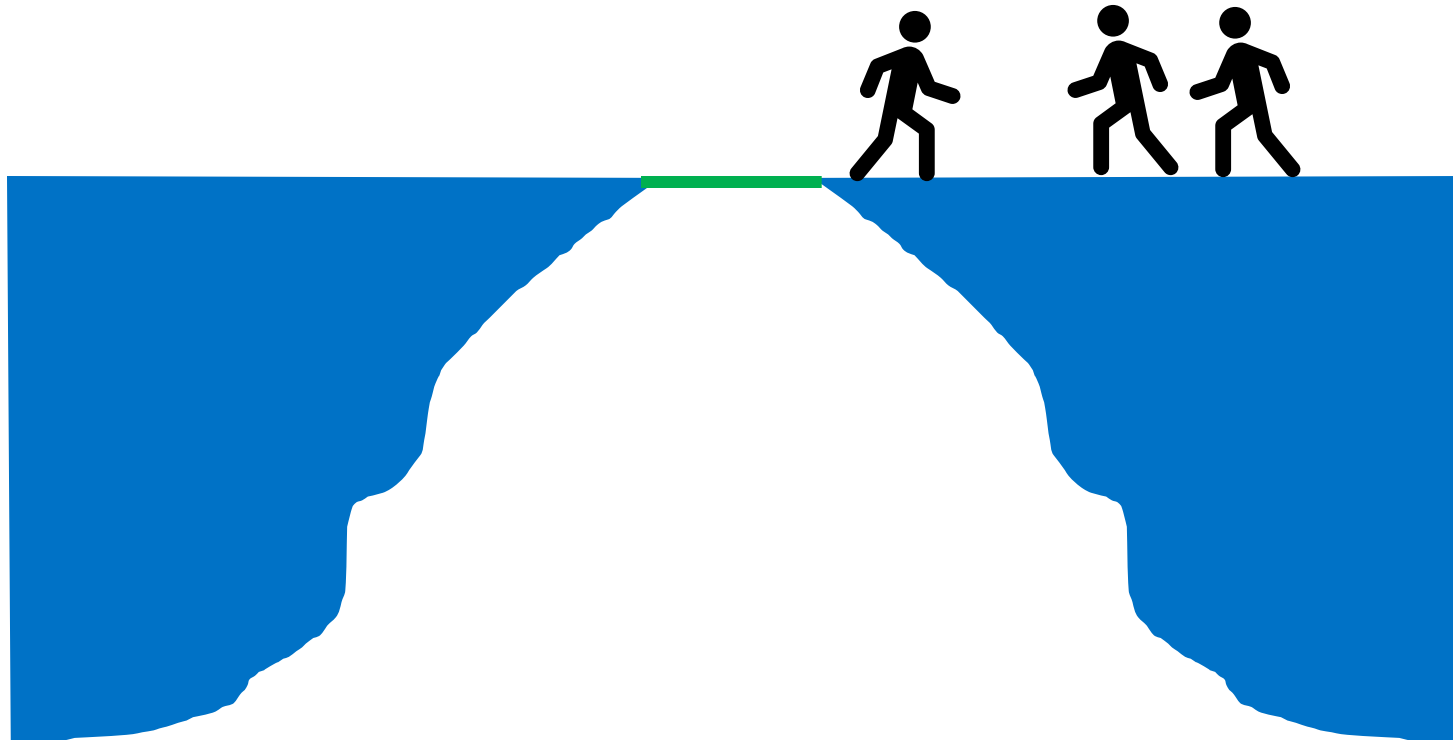
The Chasm of Forgotten Decisions



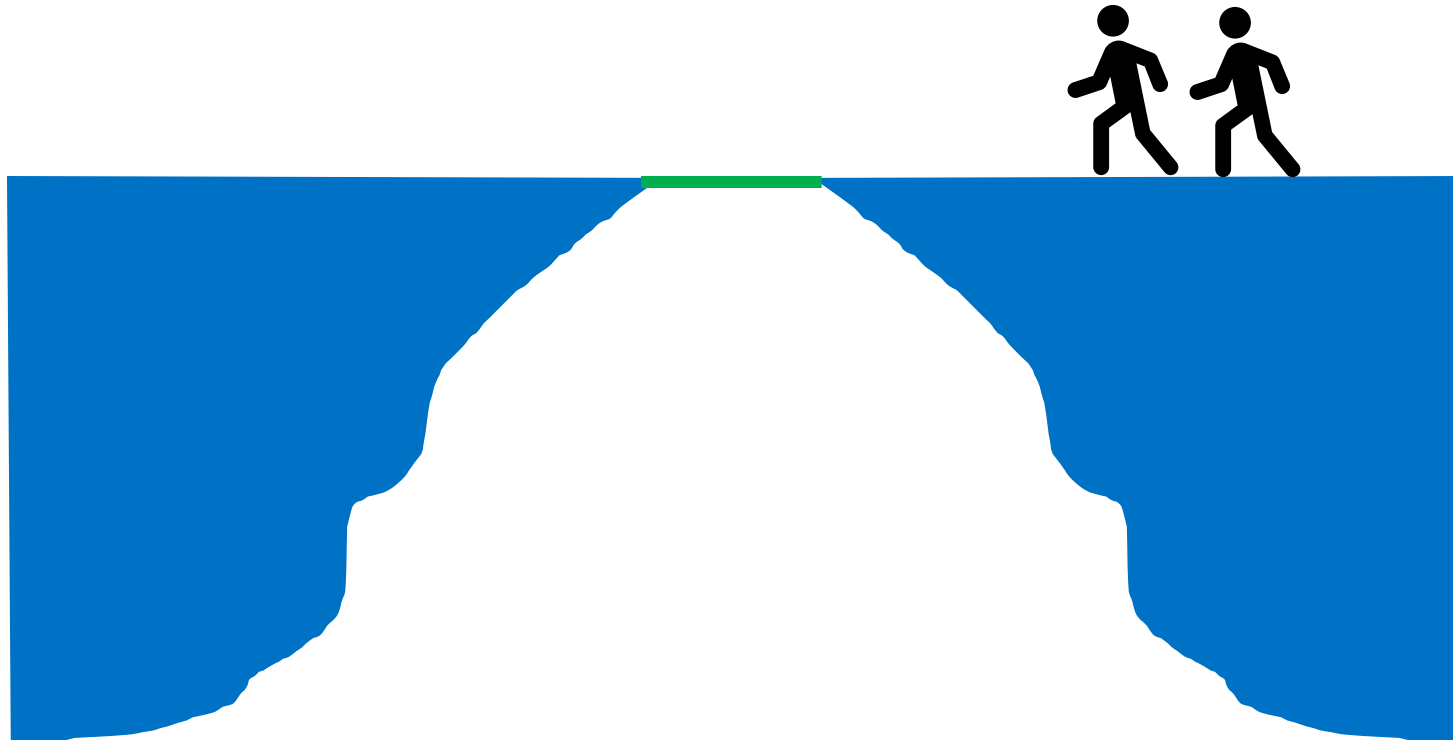
The Chasm of Forgotten Decisions



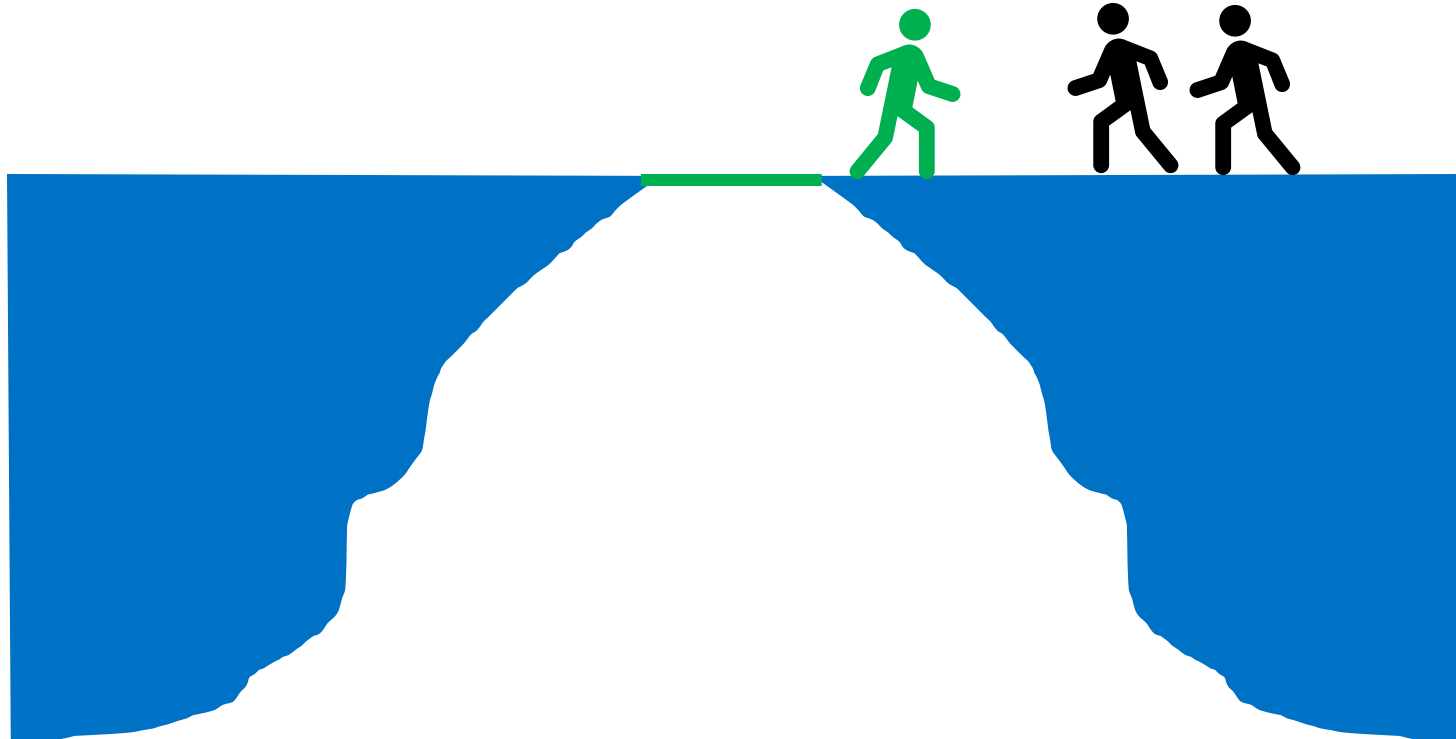
The Chasm of Forgotten Decisions



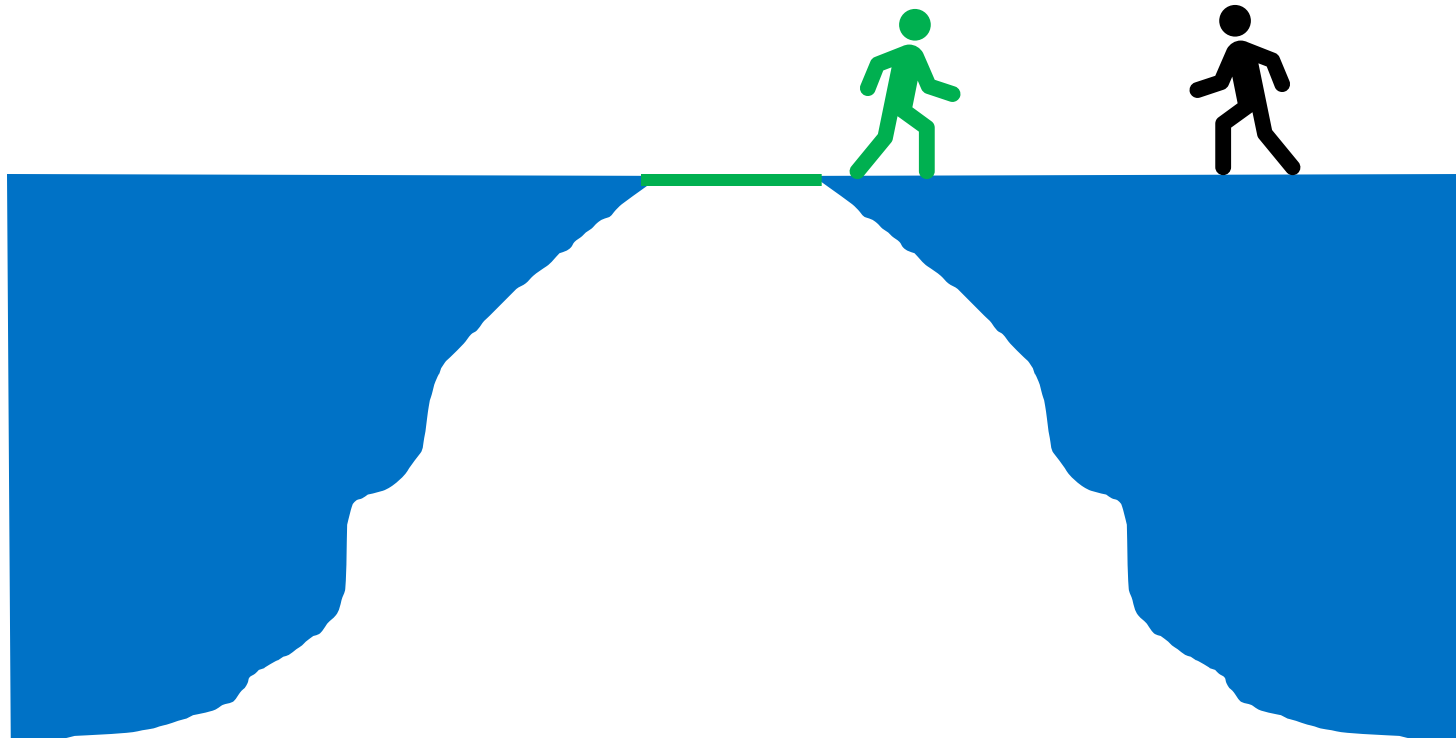
The Chasm of Forgotten Decisions



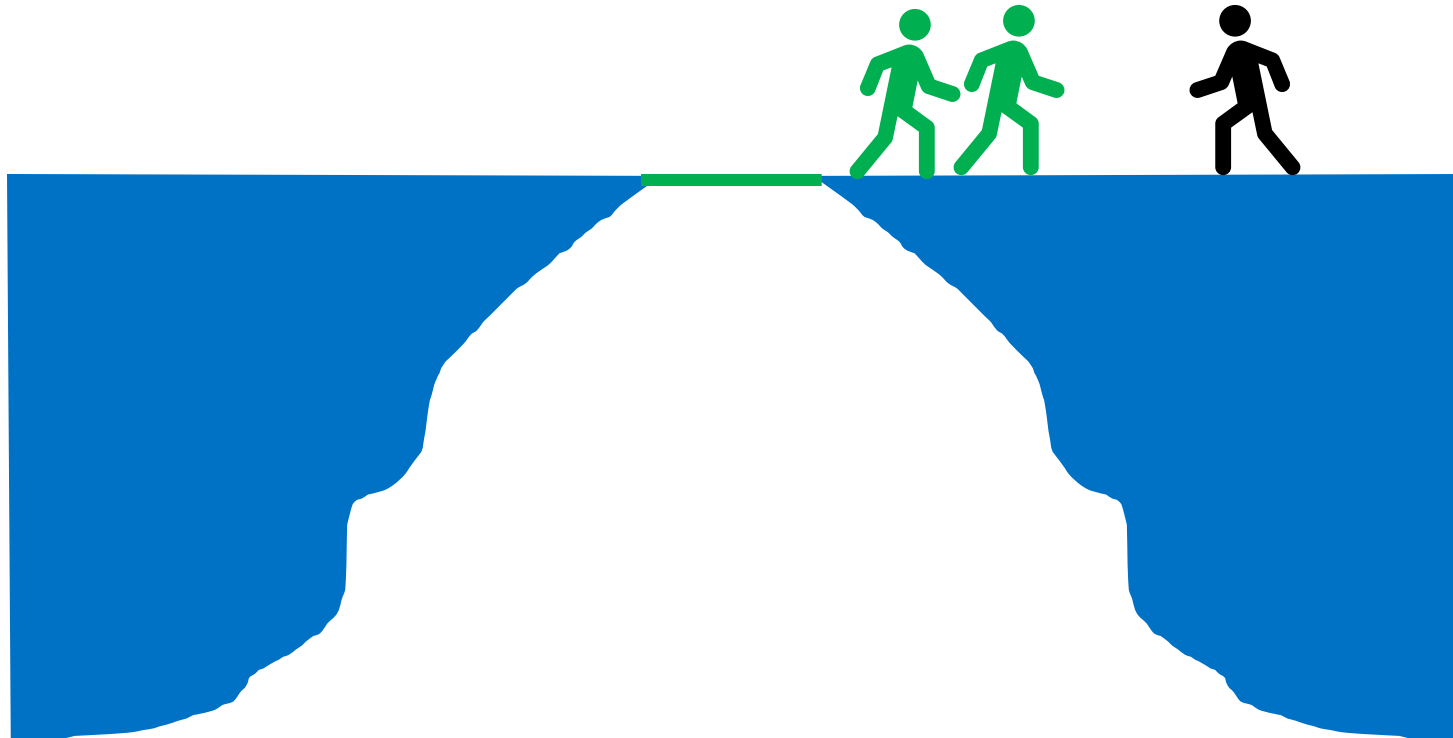
The Chasm of Forgotten Decisions



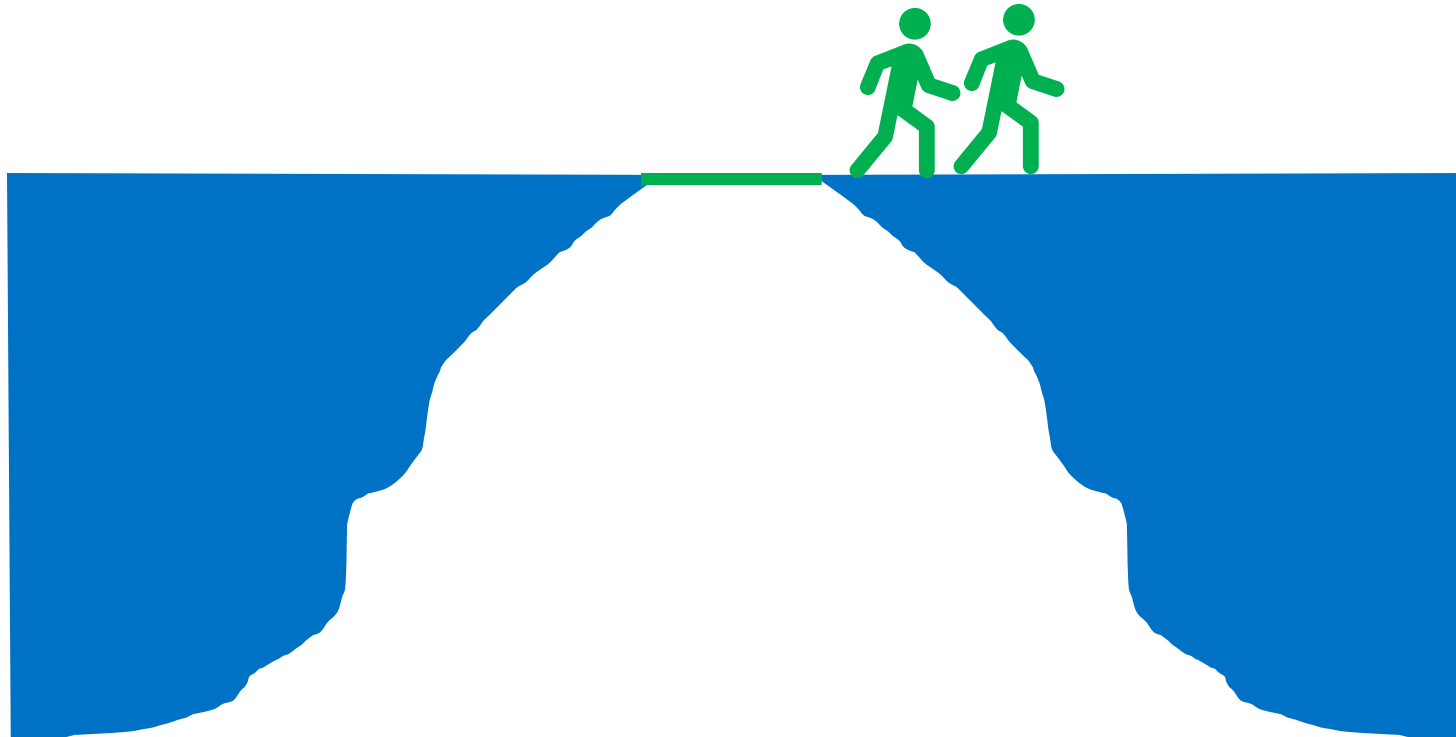
The Chasm of Forgotten Decisions



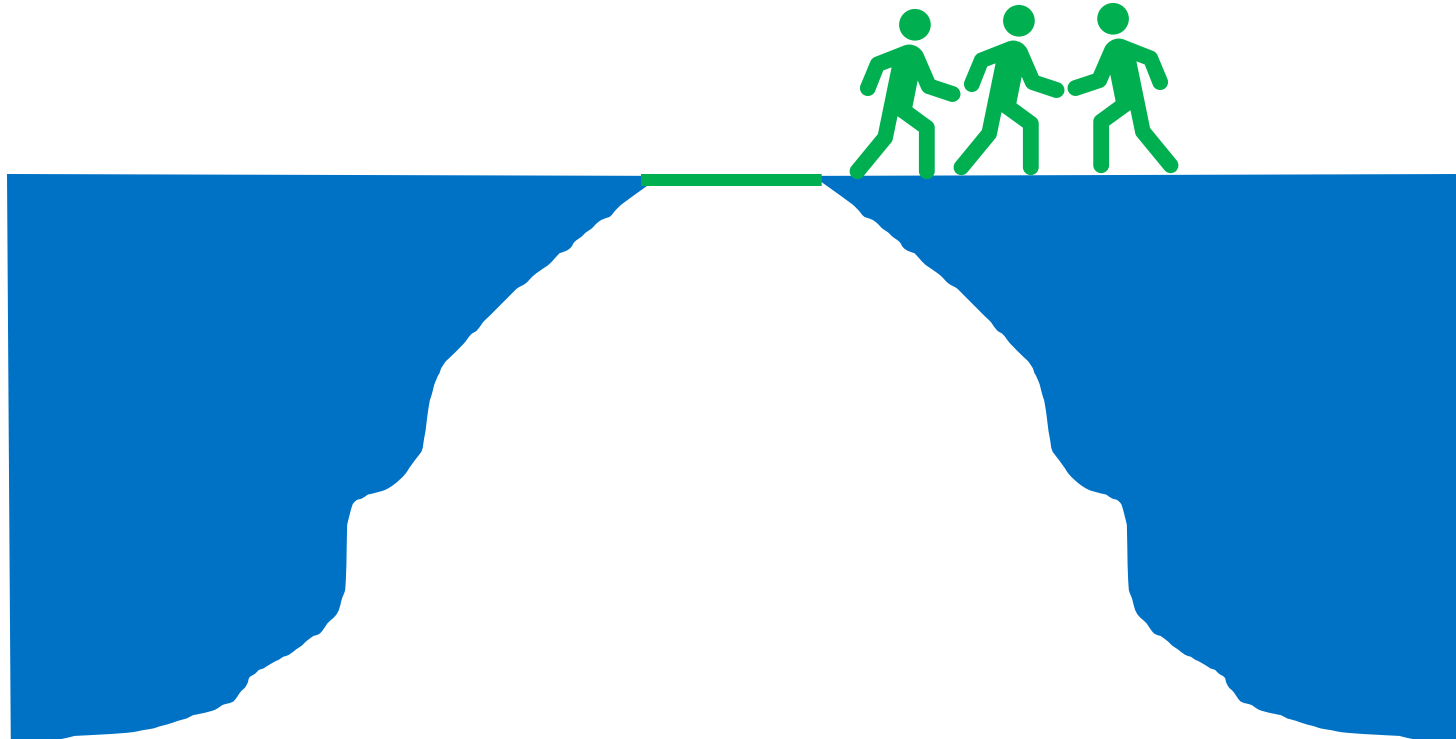
The Chasm of Forgotten Decisions



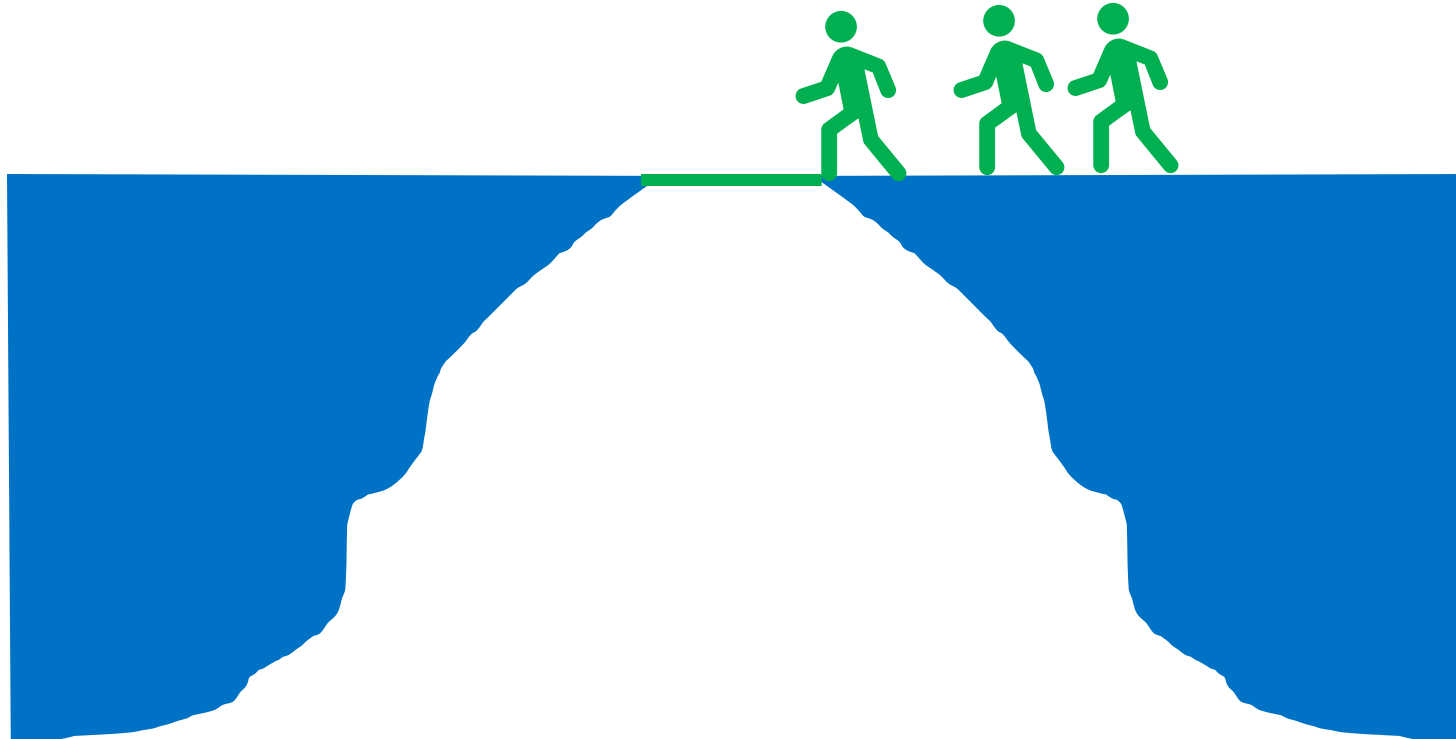
The Chasm of Forgotten Decisions



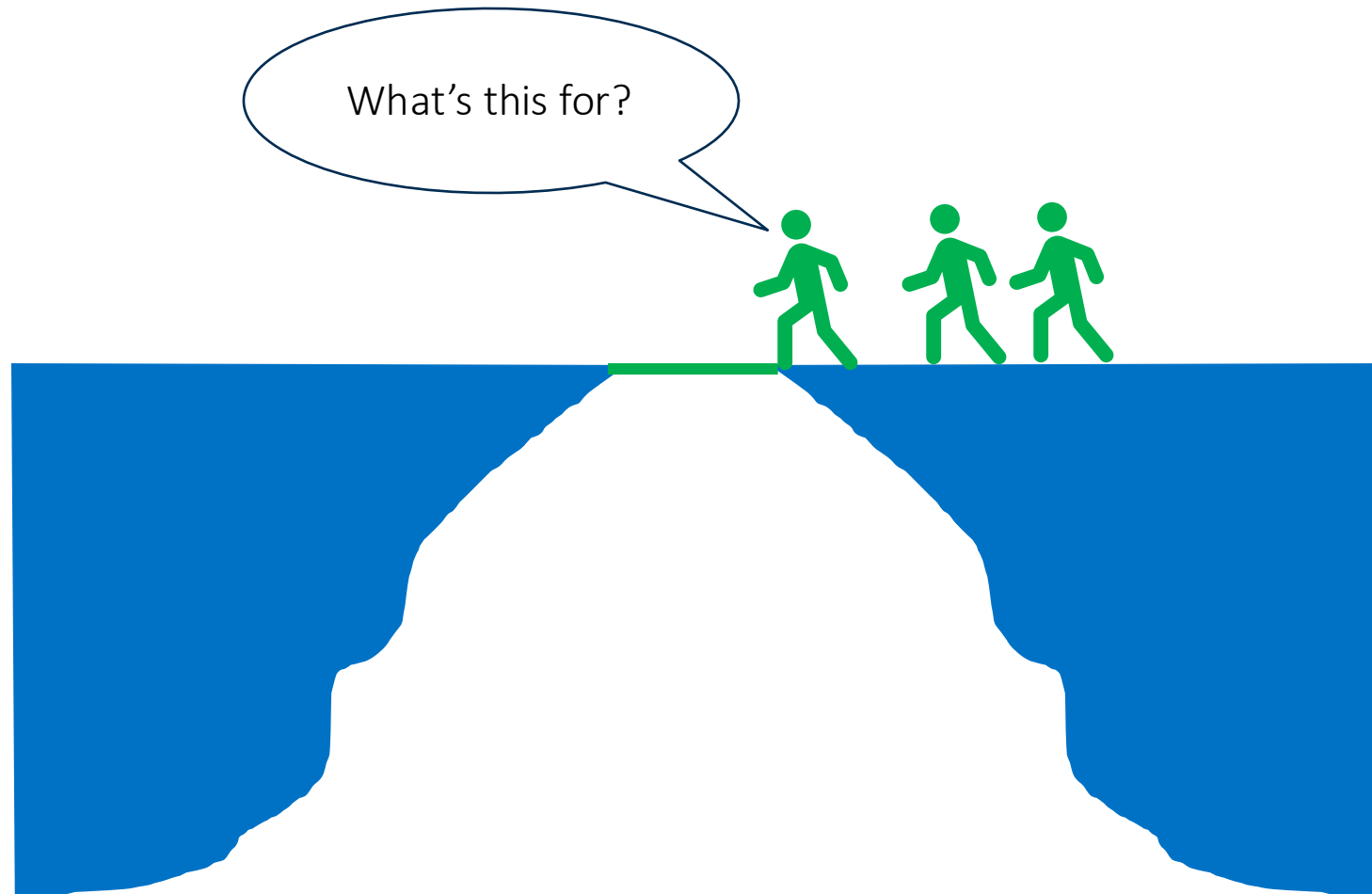
The Chasm of Forgotten Decisions



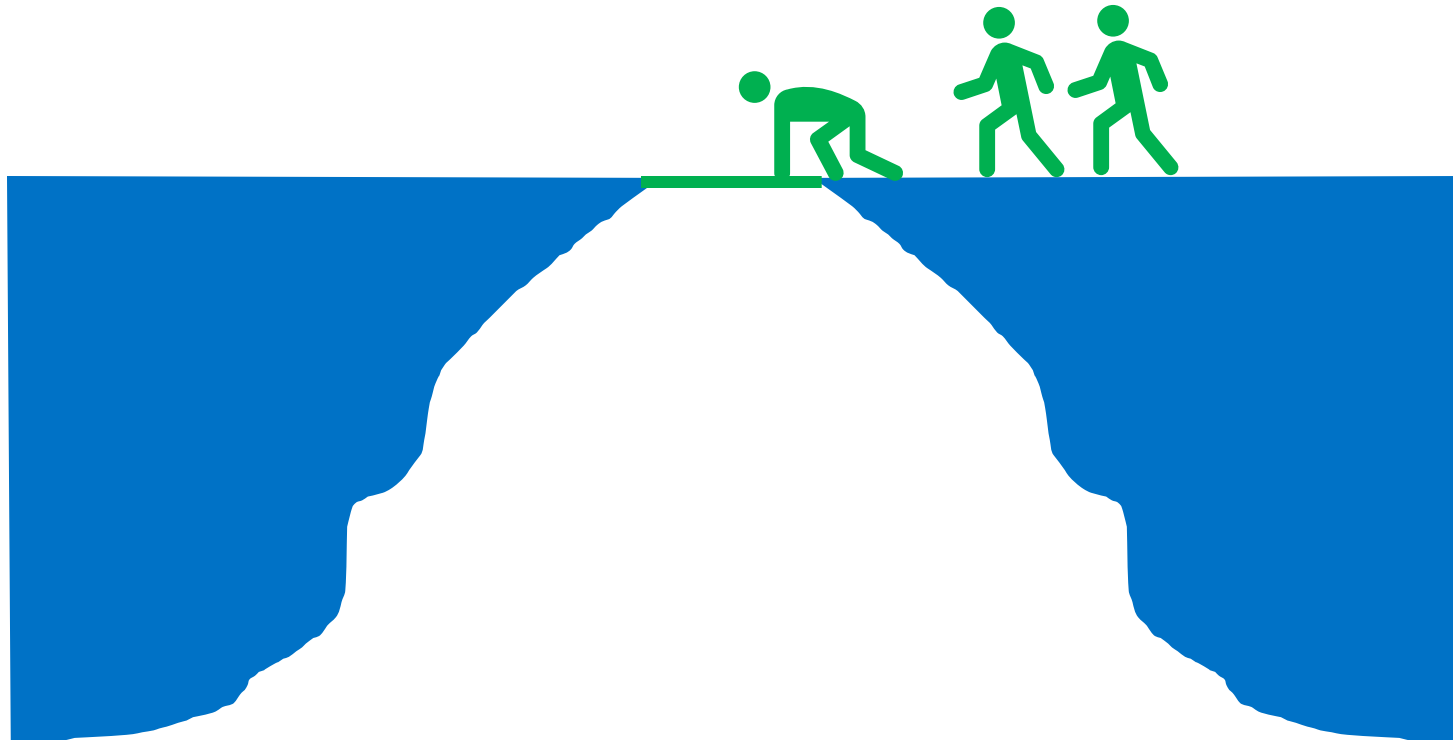
The Chasm of Forgotten Decisions



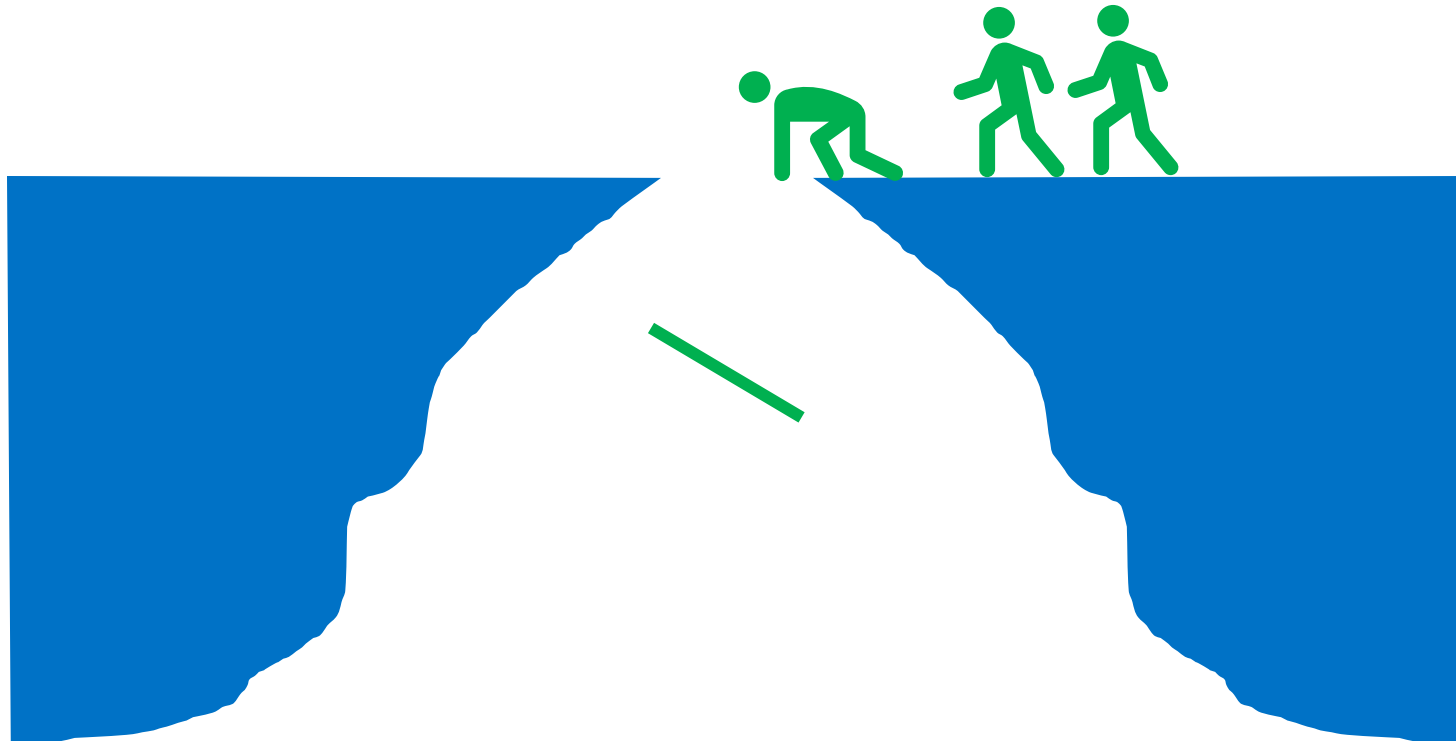
The Chasm of Forgotten Decisions



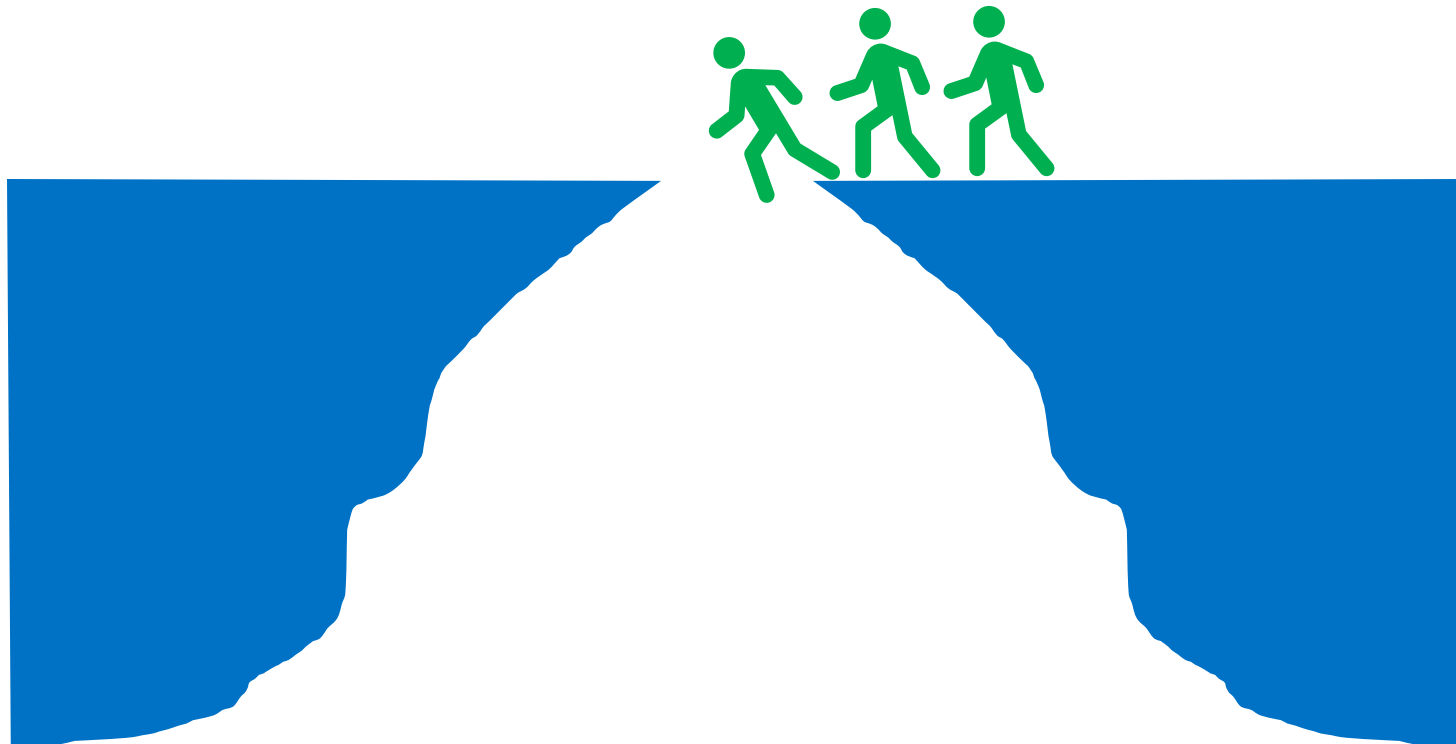
The Chasm of Forgotten Decisions



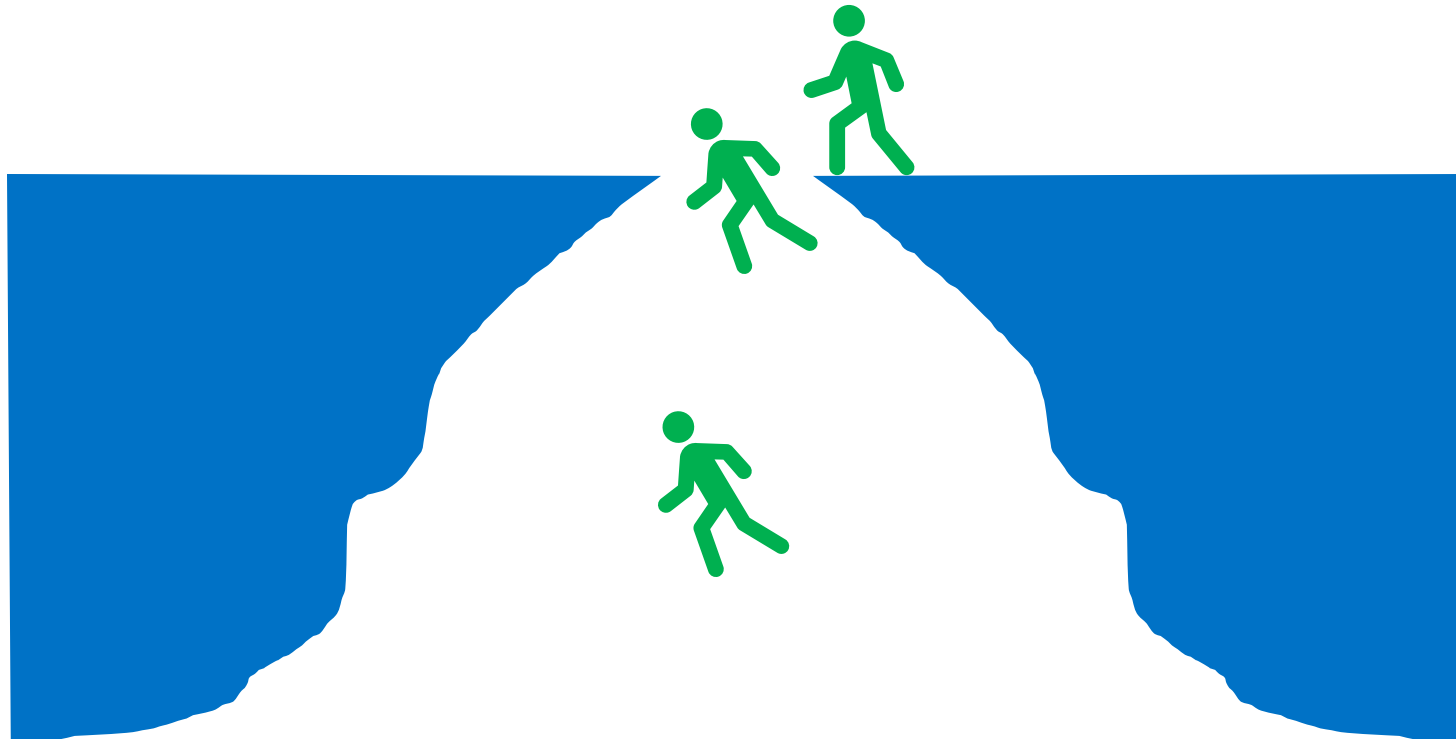
The Chasm of Forgotten Decisions



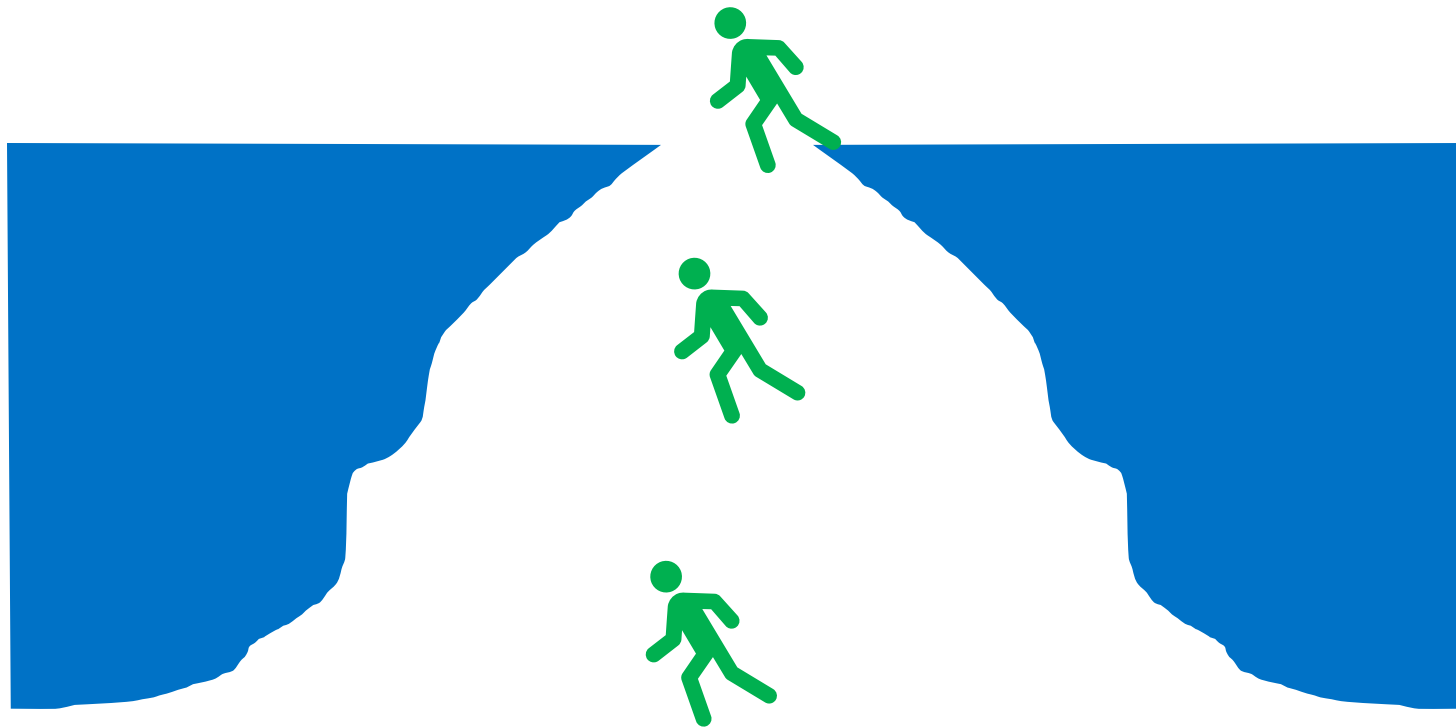
The Chasm of Forgotten Decisions



The Chasm of Forgotten Decisions



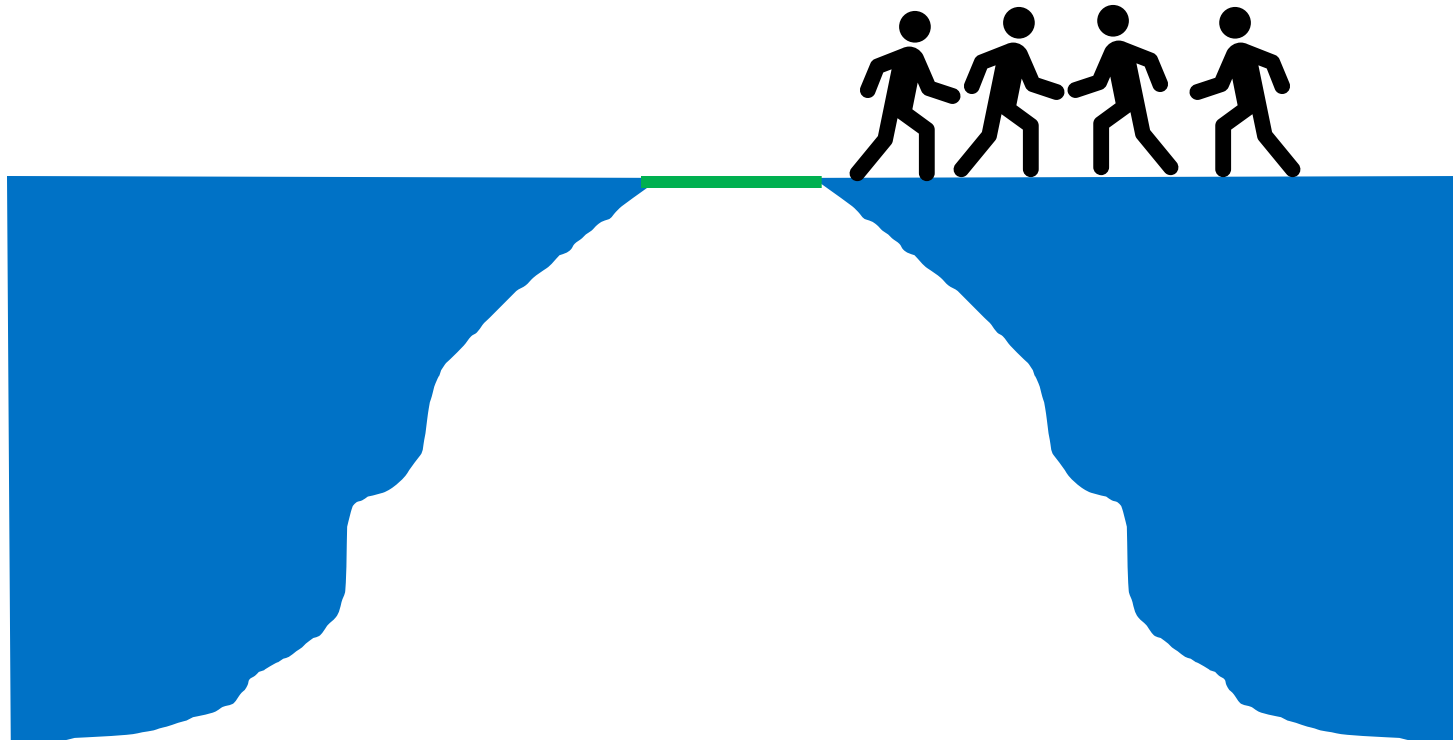
The Chasm of Forgotten Decisions



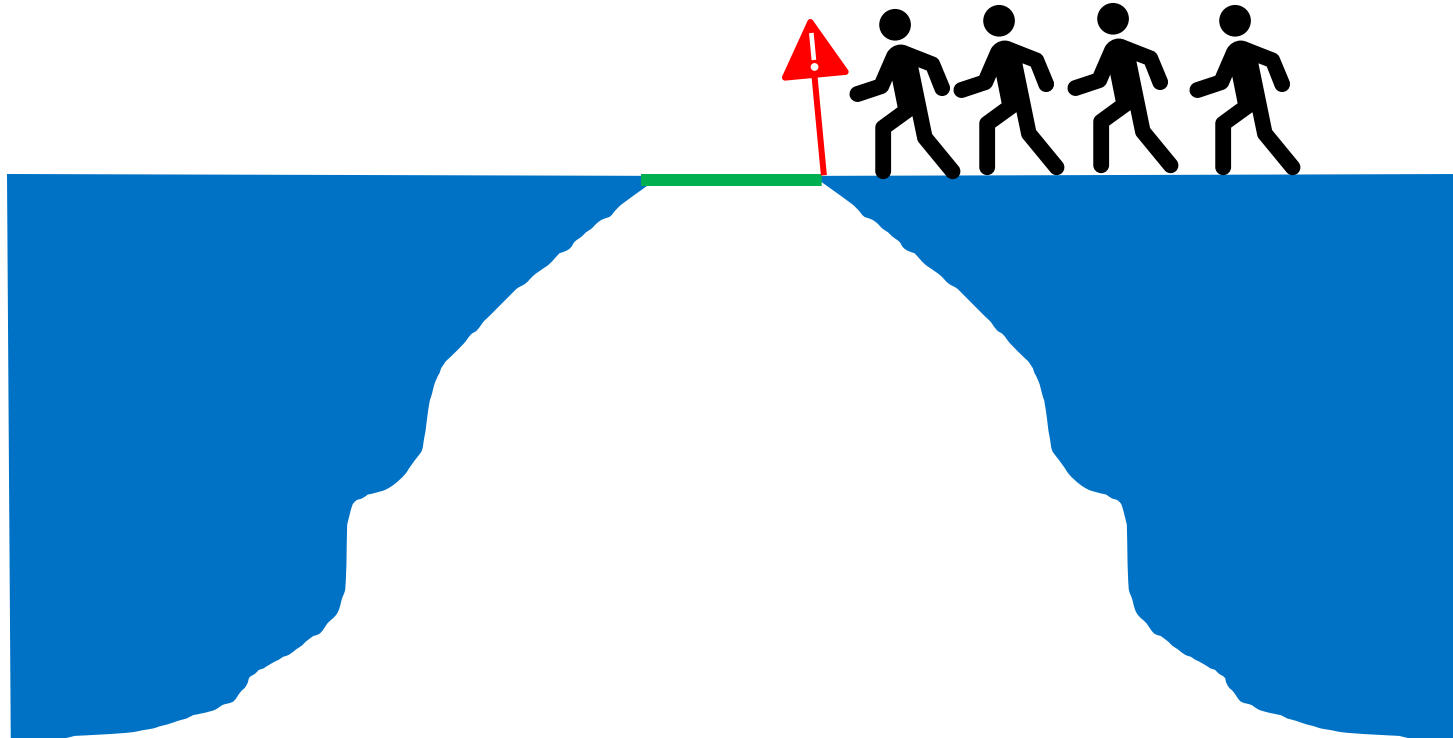
The Chasm of Forgotten Decisions



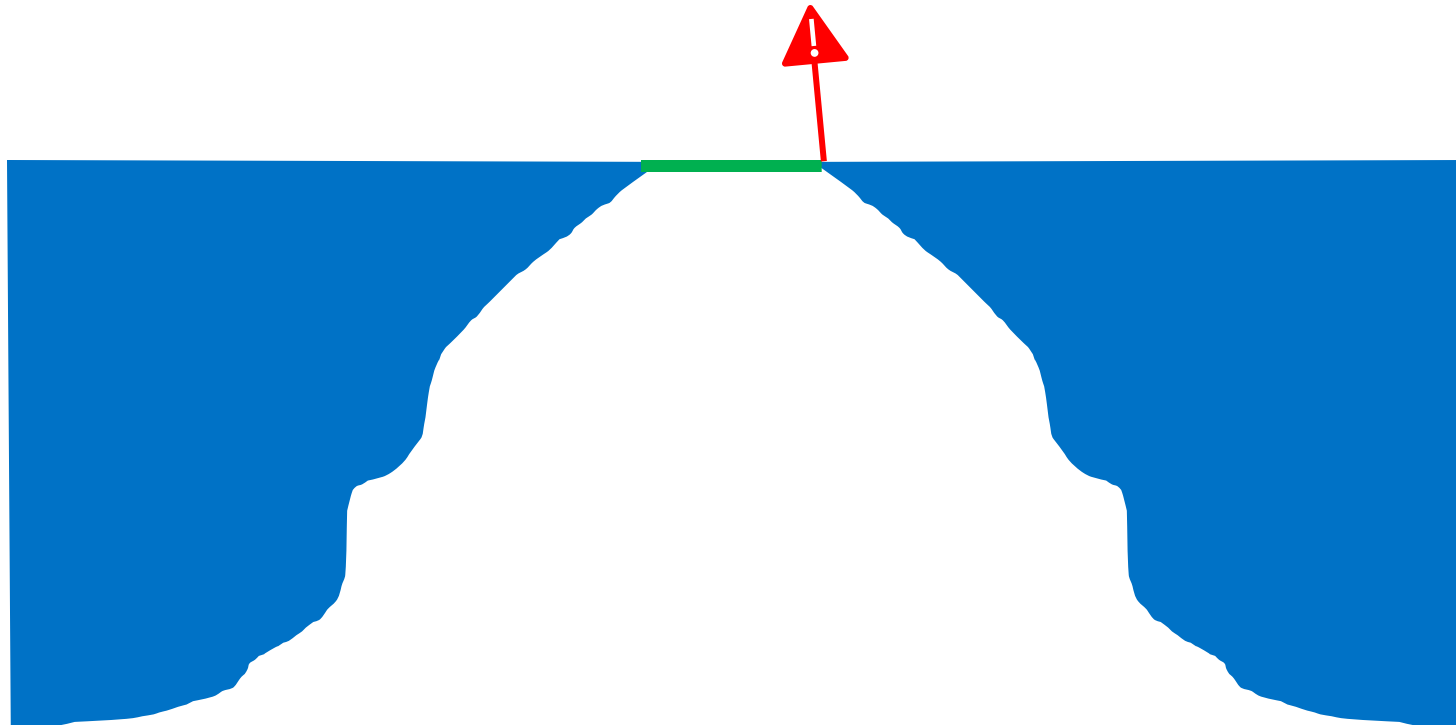
The Chasm of Forgotten Decisions



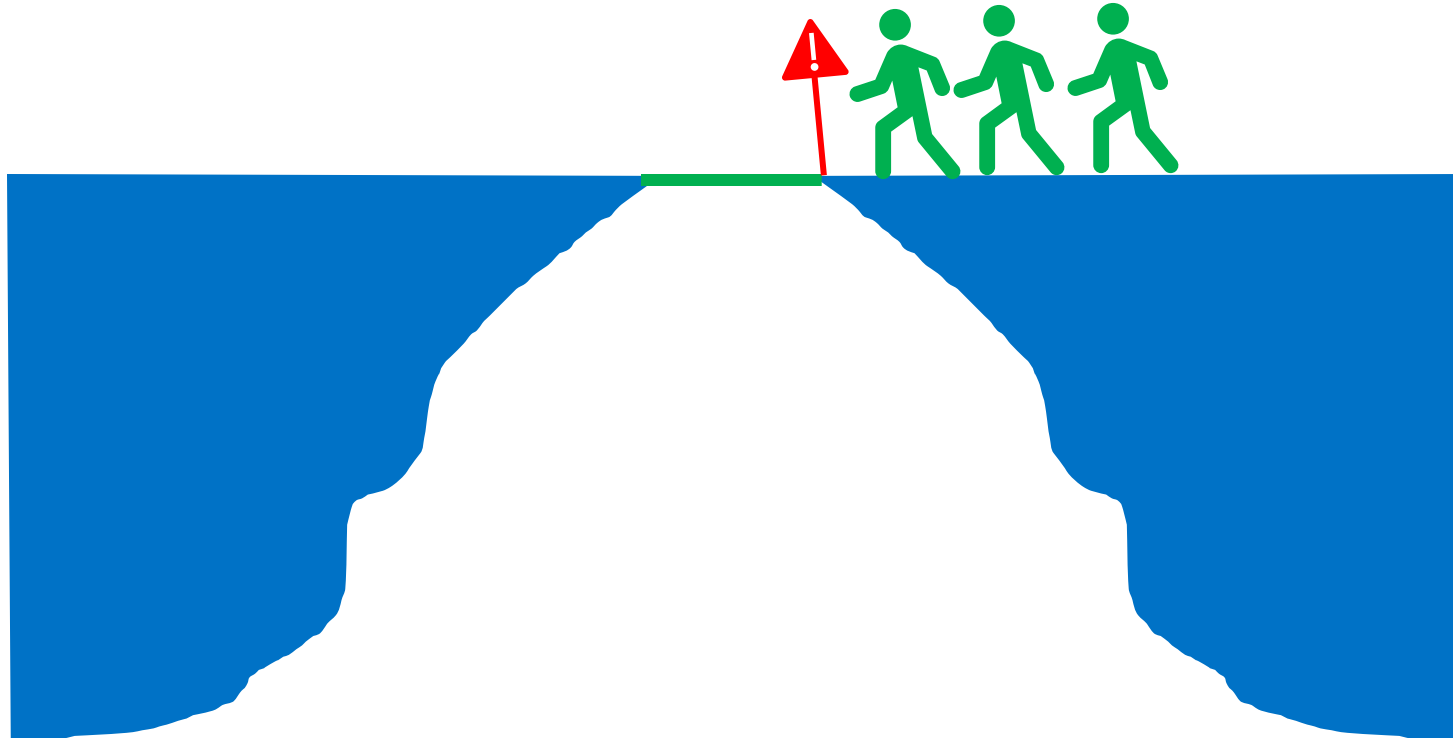
The Chasm of Forgotten Decisions



The Chasm of Forgotten Decisions



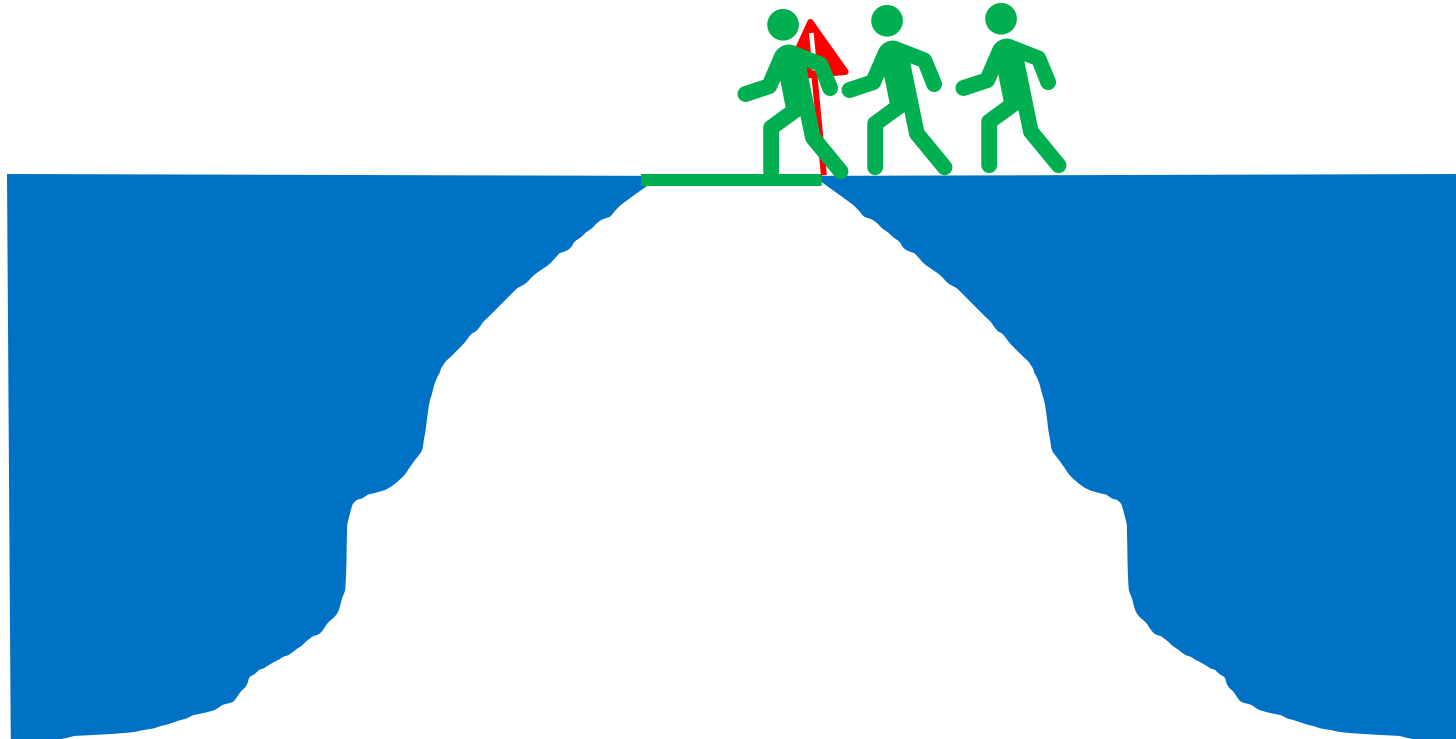
The Chasm of Forgotten Decisions



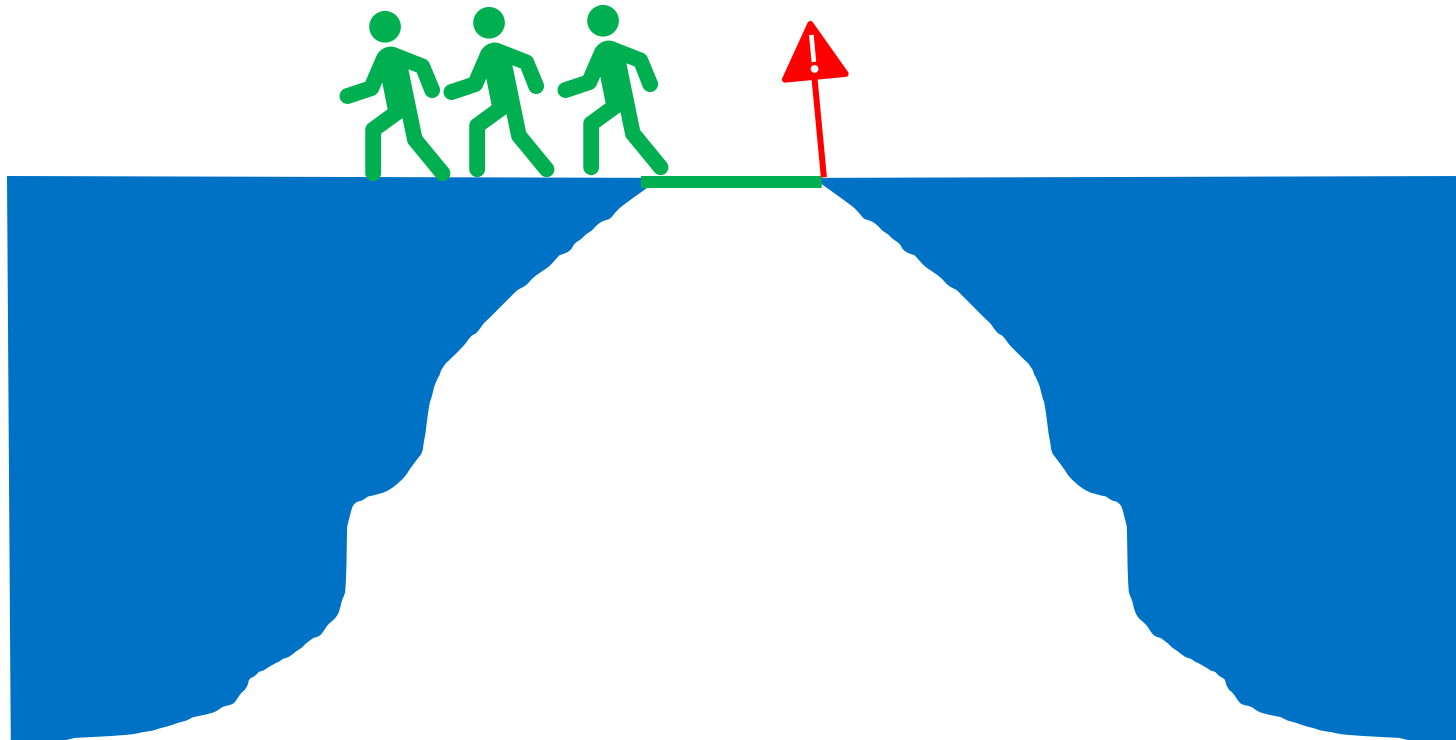
The Chasm of Forgotten Decisions



The Chasm of Forgotten Decisions



The Chasm of Forgotten Decisions



Learn what ADRs are,
How to create and maintain them, &
How ADRs help teams document the
'why'

Learn what **ADRs** are,
How to create and maintain them, &
How ADRs help teams document the
'why'

Learn what **ADRs** are,
How to **create and maintain** them, &
How ADRs help teams document the
'why'

Learn what **ADRs** are,
How to **create and maintain** them, &
How ADRs help teams **document the**
'why'

Jonathan "J." Tower

Principal Consultant & Partner



🏆 Microsoft MVP in .NET

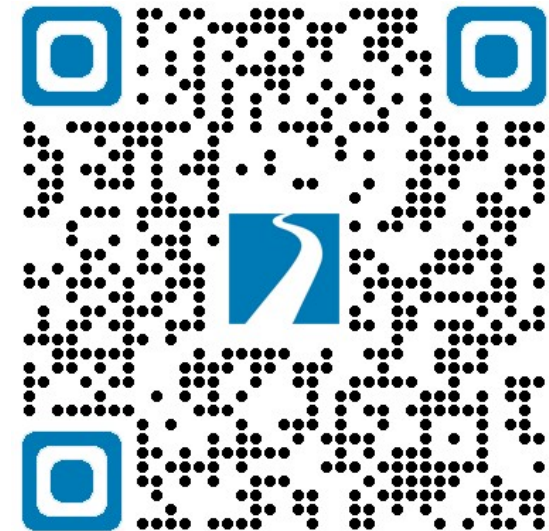
✉ jtower@trailheadtechnology.com

🌐 trailheadtechnology.com/blog

🐦 jtowermi

in jtower

EXPERT CONSULTATION



bit.ly/th-offer

github.com/trailheadtechnology/adrs

What is an **Architectural Decision Record?**

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn’t **forget, repeat mistakes, or guess** later.

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn't **forget, repeat mistakes, or guess** later.

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn’t **forget, repeat mistakes, or guess** later.

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn’t **forget**, **repeat mistakes**, or **guess** later.

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn’t **forget, repeat mistakes**, or **guess** later.

Architecture Decision Records (ADR)

Document the “**why**” behind key **technical decisions** so your team doesn’t **forget, repeat mistakes, or guess** later.

ADR Format

ADR Format

In the context of <use case/user story u>

ADR Format

In the context of <use case/user story u>
facing <concern c>

ADR Format

In the context of <use case/user story u>
facing <concern c>
we decided for <option o>

ADR Format

In the context of <use case/user story u>
facing <concern c>
we decided for <option o>
over <alternatives a>

ADR Format

In the context of <use case/user story u>
facing <concern c>
we decided for <option o>
over <alternatives a>
to achieve <quality q>

ADR Format

In the context of <use case/user story u>
facing <concern c>
we decided for <option o>
over <alternatives a>
to achieve <quality q>
accepting <downside d>

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

Benefits of Using ADRs



Create Accountability
Without Blame



Persist important
decisions long-term



Context for new team
members



Document
unimplemented
decisions



Preserve the “Why”
Behind Technical
Decisions



Reduce Rework and
Duplicate Debates



Lightweight
Documentation



Make Your
Architecture
Discoverable

ADRs By Example

THE LORD OF THE RINGS



Let's Document That Decision...

With an ADR

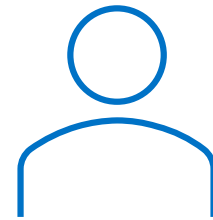
File Format: Markdown



Simple



Not Proprietary



Easy for
Non-Technical Users

File Format: Markdown

Heading 1

Heading 2

Heading 3

****bold**** / **italic**

[Link text](https://example.com)

![Alt text](image.jpg)

`inline code`

- Unordered item 1
- Unordered item 2

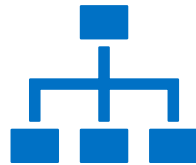
1. Ordered item 1
2. Ordered item 2

> Quoted/indented text

Number



Unique, sequential
number



For organization and
reference, not
importance or priority



Don't overthink the
numbering



No need to reserve or
preplan numbers

Number

****ADR #:** 0001**

Title



Use a short, present-tense phrase (like a commit message)



Stick to what was decided, not the rationale or outcome



Think of it like naming a file or PR title: concise, descriptive, no fluff

Title

****ADR #:** 0001**

****Title:** Abandon the Pass of Caradhras and Enter the Mines of Moria**

Filename



Summarize the title,
but even shorter



NNNN-short-title-with-dashes.md

Filename

0001-enter-moria.md

****ADR #:**** 0001

****Title:**** Abandon the Pass of Caradhras and Enter the Mines of Moria

Status

Status	Meaning
Proposed	Still being discussed—not final
Accepted	Actively in effect or currently implementing
Rejected	We considered it, but didn't go that route
Deprecated	The decision was made, but we've moved on or plan to replace it
Superseded	Replaced by another ADR (usually includes a reference to that ADR)

Status

0001-enter-moria.md

****ADR #:** 0001**

****Title:** Abandon the Pass of Caradhras and Enter the Mines of Moria**

****Status:** Accepted**

Date



Use either **January, 13, 3019**
or **3019-01-13** format



The date of the most
recent status change

Date

0001-enter-moria.md

****ADR #:** 0001**

****Title:** Abandon the Pass of Caradhras and Enter the Mines of Moria**

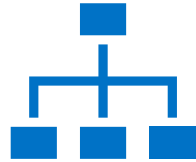
****Status:** Accepted**

****Date:** January 13, 3019 (Third Age)**

Context



The **situation** that prompted the decision



Relevant technical, business, or human constraints



Any known pain points or prior attempts



Context about the environment (time pressure, Saruman's interference)

Context

0001-enter-moria.md

****ADR #:**** 0001

****Title:**** Abandon the Pass of Caradhras and Enter the Mines of Moria

****Status:**** Accepted

****Date:**** January 13, 3019 (Third Age)

****Context:**** The Fellowship set out from Rivendell with the goal of reaching Mordor while avoiding detection. We originally planned to cross the Misty Mountains via the Pass of Caradhras, a straightforward route with minimal entanglements. However, the pass has proven impassable due to heavy snow, magical interference from Saruman, and a general lack of adequate cold-weather gear among team members (esp. Hobbits).

Decision

- Be **concise and direct**—avoid rambling or explanation.
- Stick to the **primary decision**. Put secondary details in "Consequences" or "Assumptions")
- Write it like a command or declaration—use **present tense**.
 - *No "we might" or "probably should."*
 - *Say "We will use Azure Functions," not "Azure Functions will be used."*

Decision

0001-enter-moria.md

****Decision:**** We will abandon the Caradhras route and proceed instead through the ancient Dwarven mines of Moria. While this path poses significant risks—including darkness, potential orc activity, and unknown system integrity—it offers shelter from surveillance and harsh environmental conditions.

Alternatives Considered [Optional]

- List the main alternatives you seriously considered.
- Briefly explain **why each was rejected or deferred**
- Keep it concise—just enough detail to understand the trade-offs
- Avoids “Why didn’t you just do X?” questions later
- Shows your team did its homework—important when decisions are questioned
- Helps new team members understand the landscape of choices

Alternatives Considered [Optional]

0001-enter-moria.md

****Decision:**** We will abandon the Caradhras route and proceed instead through the ancient Dwarven mines of Moria. While this path poses significant risks—including darkness, potential orc activity, and unknown system integrity—it offers shelter from surveillance and harsh environmental conditions.

****Alternatives Considered:****

- **Push forward over Caradhras** - *Rejected due to magical interference, risk of avalanche, and frozen hobbit toes.*
- **Detour south toward the Gap of Rohan** - *Dismissed due to increased exposure to enemy surveillance and geopolitical instability.*
- **Fly over the mountains on eagles** - *Shot down for being logistically implausible, tactically unsound, and way too obvious.*

Deciders

- Who agreed to this?
- Who should you ask if you have questions?

Deciders

```
0001-enter-moria.md
```

```
**Deciders:** Gandalf the Grey
```

Contributors [Optional]

- Anyone who provided input, feedback, or research that influenced the decision
- Don't include Deciders here

Contributors [Optional]

0001-enter-moria.md

****Deciders:**** Gandalf the Grey

****Contributors:**** Aragorn, Legolas, Gimli, Boromir, Frodo, Sam, Merry, Pippin, Bill the Pony

Assumptions [Optional]

- Technical assumptions (e.g., system capabilities, resource availability)
- Business or organizational assumptions (e.g., team skills, budget constraints)
- Environmental or external factors (e.g., vendor stability, market conditions)

Assumptions [Optional]

0001-enter-moria.md

****Deciders:**** Gandalf the Grey

****Contributors:**** Aragorn, Legolas, Gimli, Boromir, Frodo, Sam, Merry, Pippin, Bill the Pony

****Assumptions:****

- The mines are still navigable (no cave-ins, blocked paths, or magical seals).
- Hostile forces within Moria are either dormant or avoidable.
- Gandalf remembers the way.
- The Ring will not attract undue attention underground (unlikely, but everyone's pretending otherwise).

Consequences

- Effects, risks, and trade-offs that result from the decision
- Positive outcomes and improvements
- Potential downsides, risks, or technical debt
- Impacts on team workflow, architecture, or performance
- Any follow-up actions or monitoring needed

Consequences

0001-enter-moria.md

****Consequences:****

- The team will move underground, losing visibility to the outside world and internet connectivity.
- Gandalf's access to production is elevated, as he is the only team member with knowledge of the Moria subsystem.
- Risk of encountering legacy entities (e.g., the Balrog) that have not been maintained or documented in millennia.
- Morale impact: the Hobbits are nervous, Gimli is thrilled, and Boromir is grumbling again.
- If successful, we'll emerge nearer to Lothlórien, ahead of schedule—but survivability metrics are uncertain.

Reconsideration Conditions [Optional]

- Clear triggers for reviewing the decision
- Specific conditions that would invalidate the decision
- New technology, architecture shifts, team changes, or external factors to watch
- Performance issues, security incidents, or failures related to the decision

Reconsideration Conditions [Optional]

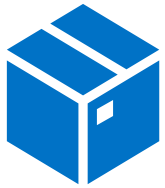
0001-enter-moria.md

****Reconsideration Conditions:****

- A Balrog is encountered.
- Gandalf is lost or incapacitated.
- The mines become impassable.
- A safer route becomes available unexpectedly.

Where Do ADRs Go?

Where Do ADRs Go?



In the repo (with the
code)



In a separate repo



Project wiki or
documentation site



Dedicated ADR Tool

In the Repo with the Code

Pros

- Versioned alongside code
- Easy for devs to find and update
- Supports code review workflows

Cons

- Might clutter repo if not organized
- Requires discipline to write/update
- Messy with branching



In a Separate Repo



Pros

- Easy for devs to use
- Central place for entire team
- Need for branching

Cons

- Further from code and dev workflows
- Risk of ADRs drifting from code
- Harder for non-technical team to access

Project Wiki or Documentation Site

Pros

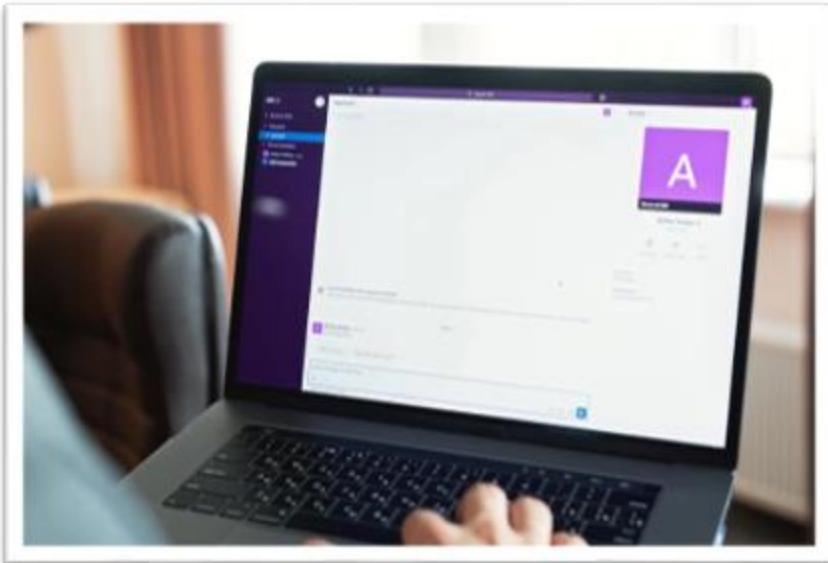
- Easy for non-devs to access
- Central place for broader team
- Can link to ADRs from many places

Cons

- Not version controlled with code
- Can become outdated quickly
- Harder to discover in dev workflows



Dedicated ADR Tools or Platforms



Pros

- Built specifically for ADR management
- Features like status tracking, search, templates
- Can integrate with other tools

Cons

- Extra setup and maintenance overhead
- Learning curve for team
- Risk of silo if not well integrated

ADR Tools

ADR-Manager

A web-based application for the efficient creation and management of architectural decision records (ADRs) in Markdown (MADR)

<https://github.com/adr/adr-manager>

Log4brains

Log4brains is a docs-as-code knowledge base that enables you to log Architecture Decision Records (ADR) right from your IDE and to publish them automatically as a static website.

<https://github.com/thomvaill/log4brains>

Loqbooq

Loqbooq gives you a solid log of all important decisions and considerations that is always accessible to everybody in the project. Includes Slack integration.

<https://loqbooq.app/>

ADR Tools (Continued)

ADR Tools

A command-line tool for working with a log of Architecture Decision Records (ADRs).

<https://github.com/npryce/adr-tools>

dotnet adr

A cross-platform .NET Global Tool for creating and managing Architectural Decision Records (ADR).

<https://github.com/endjin/dotnet-adr>

ADR Manager VS Code Extension

Visual Studio Code (VS Code) extension based on the ADR Manager

<https://marketplace.visualstudio.com/items?itemName=StevenChen.vscode-adr-manager>

Common ADR **Mistakes** to Avoid

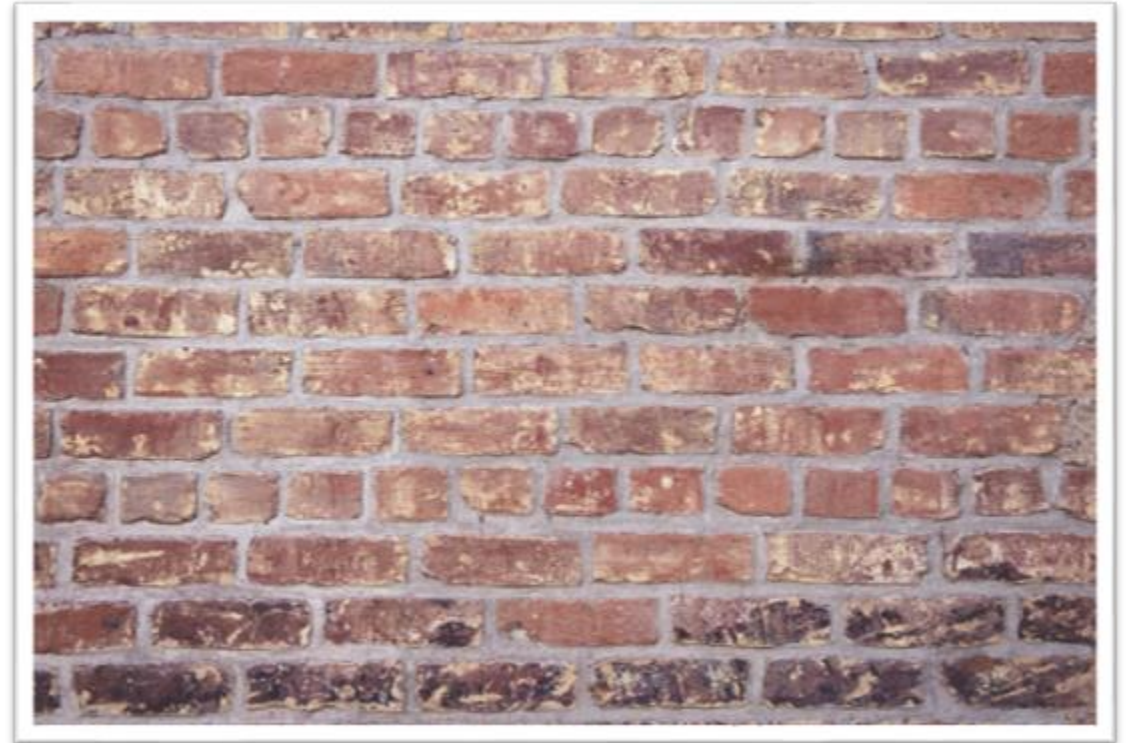
Writing ADRs After the Fact Just to “Check a Box”



- Write ADRs **during** the decision process
- A half-finished, “proposed” ADR is **better than none**
- Treat ADRs like **commit messages** for your architecture

Burying the Decision in a Wall of Context

- Make the Decision section **short, bold, and easy to find**
- Put **just enough** in the context section to explain the problem
- Use **structure** and **headings** well



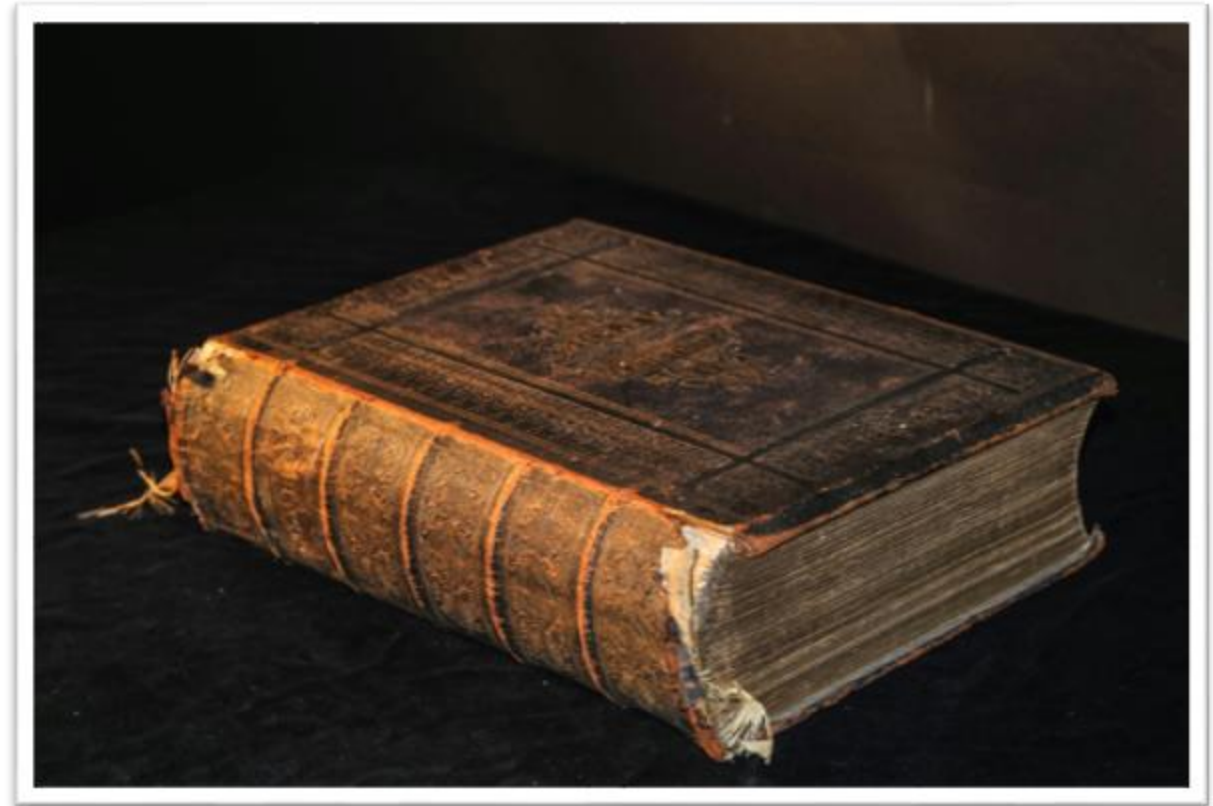
Not Updating the Status When it Changes



- Use status fields and **keep them updated**
- Open **follow-up ADRs**, reference the old ones, update the original

Writing Novels Instead of Decisions

- Think of ADRs like well-structured **release notes** for architecture: What changed, Why, What it affects
- Use headings, bullets, **short** paragraphs
- Be **specific, direct, and easy** to skim
- Capture the **insight**, not the whole existential journey

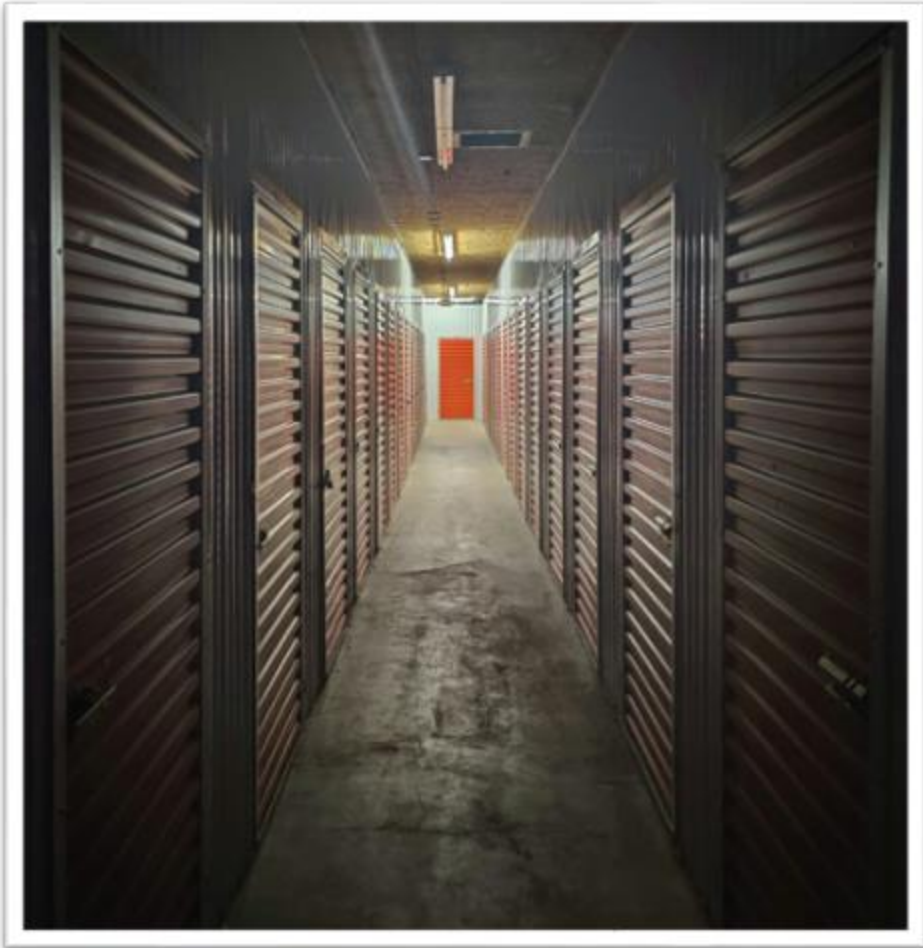


ADRs That Don't Explain *Why*



- Always include the **rationale**: the trade-offs, constraints, and motivations behind the choice
- Think of it as writing for **future you**, 6 months from now, who's forgotten everything
- Ask yourself: "Would this still make sense to someone **who wasn't in the room?**"

Storing Them in a System No One Uses



- Keep ADRs **close to the code** if possible
- Use plain-text formats like **Markdown** so they're easy to diff, review, and grep
- Link to ADRs from READMEs, wikis, and PRs to surface them at the right moment

Never Revisiting or Updating Them



- Schedule **periodic reviews** of major decisions—especially before large refactors, migrations, or team transitions
- Use statuses like **Deprecated, Superseded, or Amended** to track evolution
- Don't be afraid to **create follow-up ADRs** that build on or replace earlier ones
- Add a "**Reconsideration Conditions**" section from the start to help future-you know when to revisit

Summing Up

1. ADRs capture **otherwise hidden decisions** and trade-offs
2. They are made up of **several simple components** that capture the decision
3. Good to store them **near the code**, but consider pros and cons
4. Avoid **common mistakes** made with ADRs



Thanks! Questions?



Jonathan "J." Tower

🏆 Microsoft MVP in .NET

✉️ jtower@trailheadtechnology.com

🌐 trailheadtechnology.com/blog

🐦 [jtowermi](https://twitter.com/jtowermi)

🌐 [jtower](https://www.linkedin.com/in/jtower)

EXPERT CONSULTATION



bit.ly/th-offer

github.com/trailheadtechnology/adrs