MES
MODEL ENGINEERING SOLUTIONS

Challenges in Refactoring Software Models

Ferry Bachmann

Organisational Info

- **Questions?**
  - Ask at any time – this is an open discussion.
  - Please check whether or not you are on mute.
  - Use the Q&A if sound is not working.
- **You will receive the presentation afterwards by email.**
- **Moderator:**
  - Ferry Bachmann, Product Owner of *MES Model & Refactor*

Agenda

- Refactoring what/why/when/how & survey results
- Open discussion

Definition - What

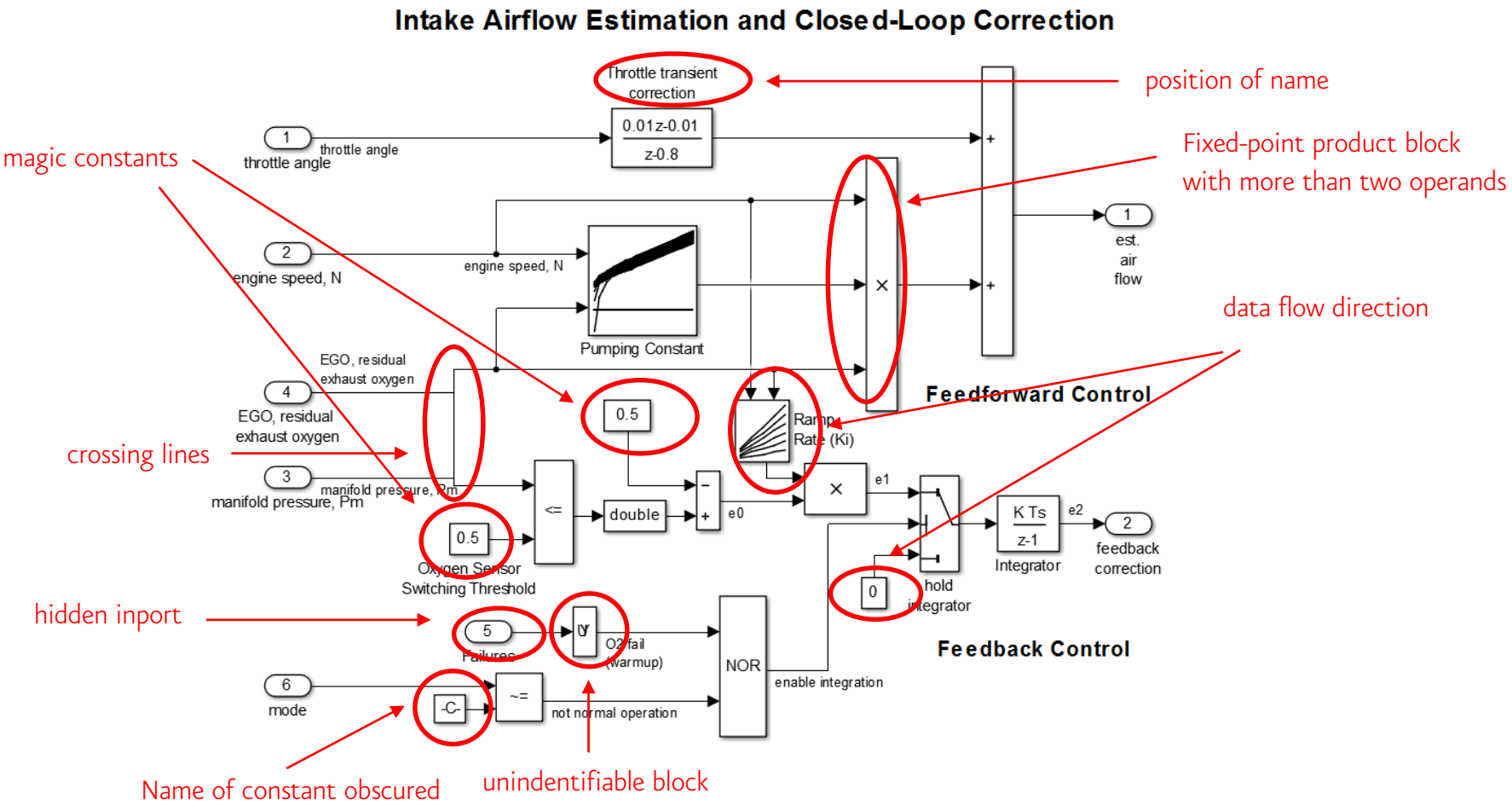
- ”A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior.“
  Martin Fowler, „Refactoring: Improving the Design of Existing Code”

- What does that mean? Big steps, small steps? Is basic guideline fixing also refactoring?

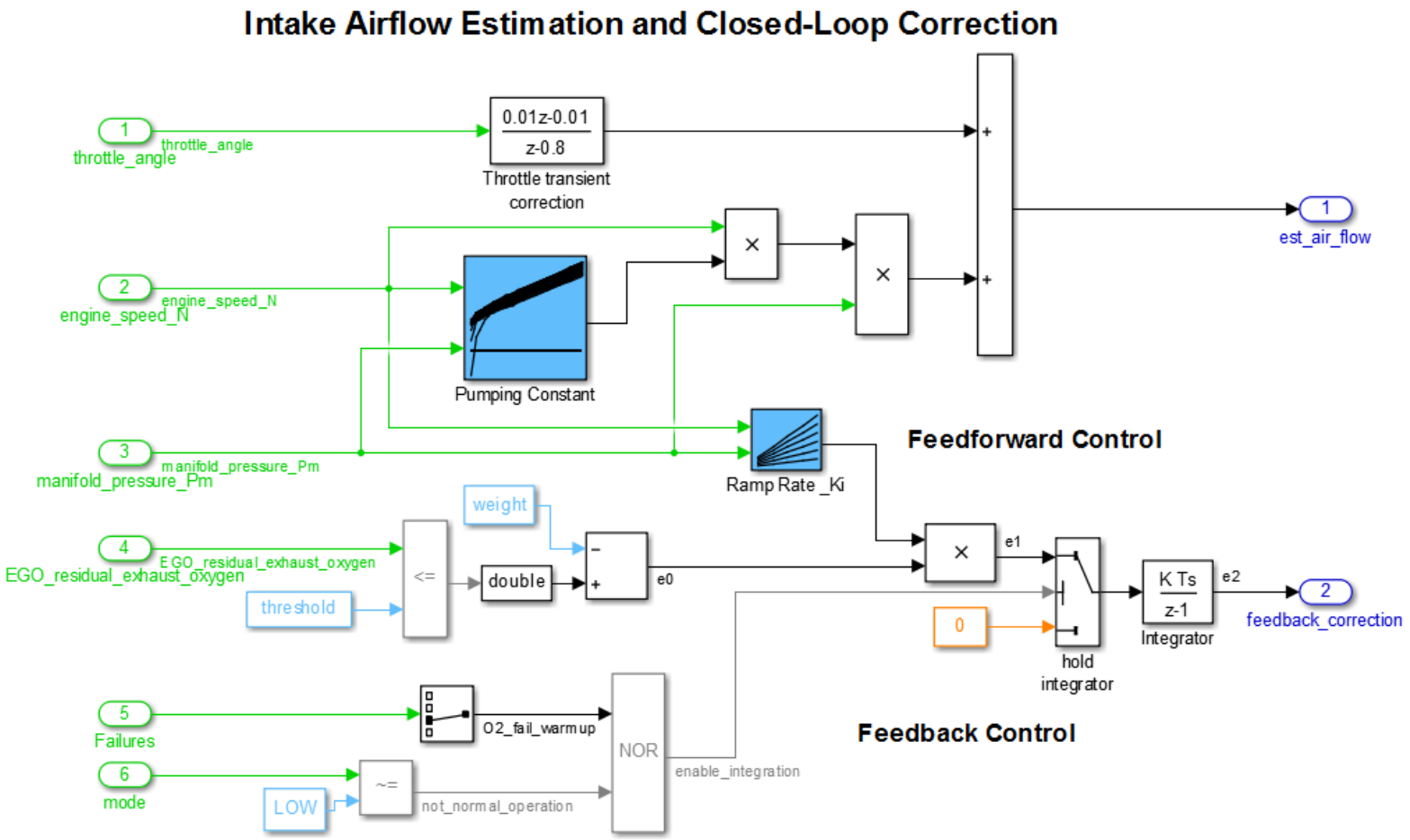
  “Its essence is applying a series of small behavior-preserving transformations, each of which ‘too small to be worth doing’. However the cumulative effect of each of these transformations is quite significant.”
  Martin Fowler, „Refactoring: Improving the Design of Existing Code”

  => Yes, basic guideline fixing is also refactoring

Example: Basic guideline fixing - before

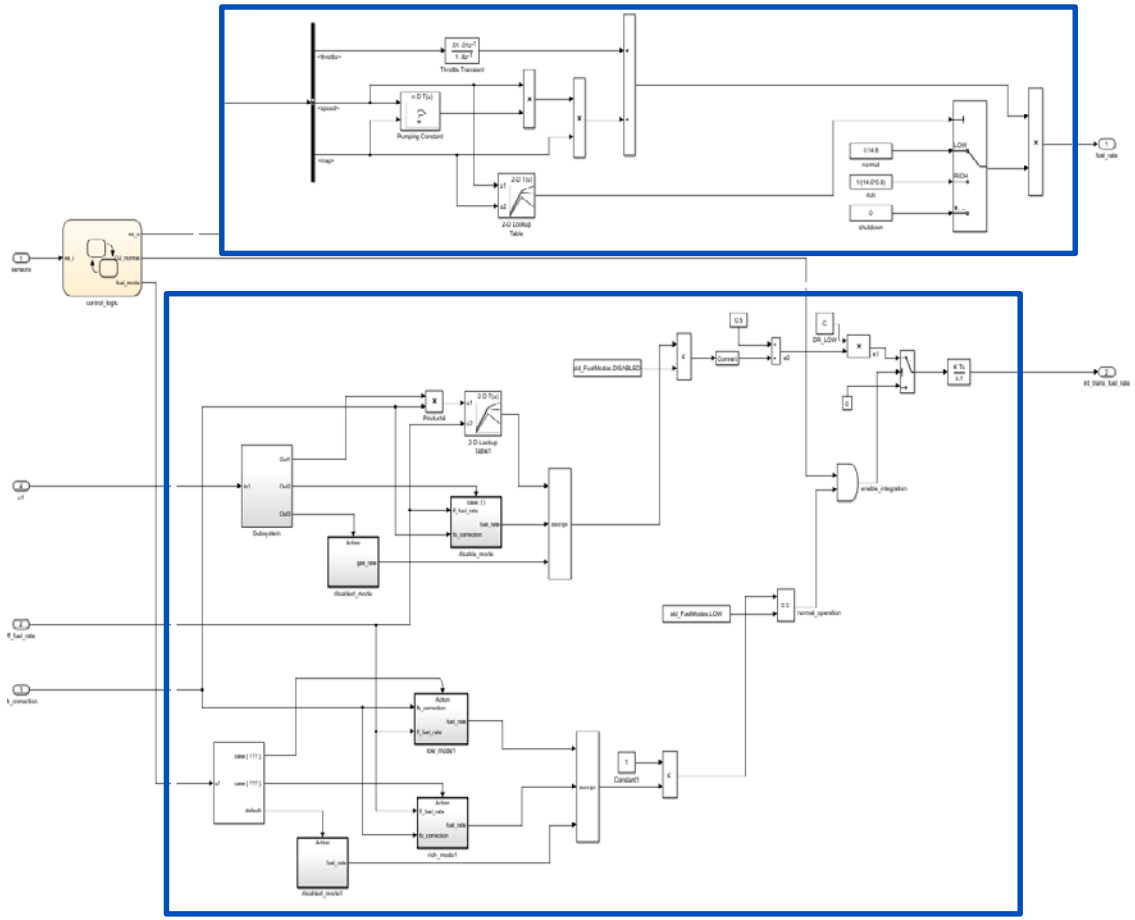


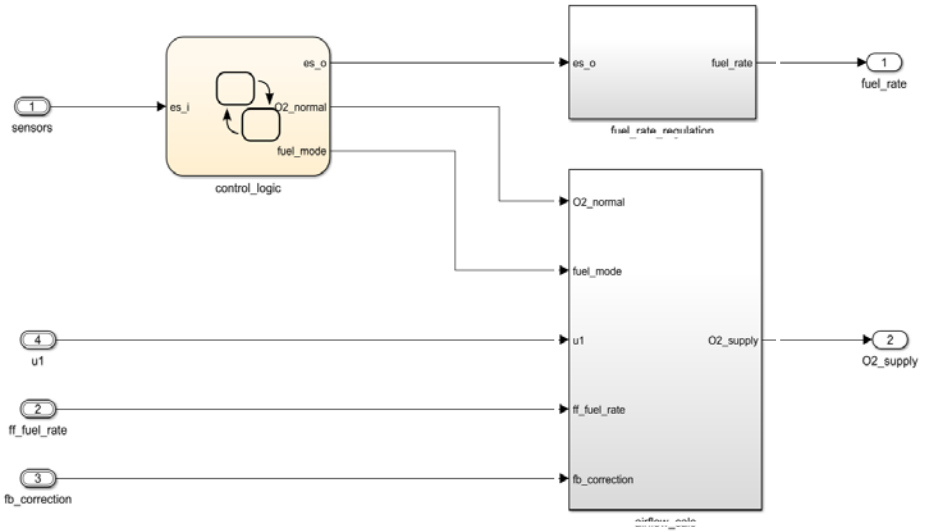
Example: Basic guideline fixing - after



Other basic refactoring examples – Create Subsystem

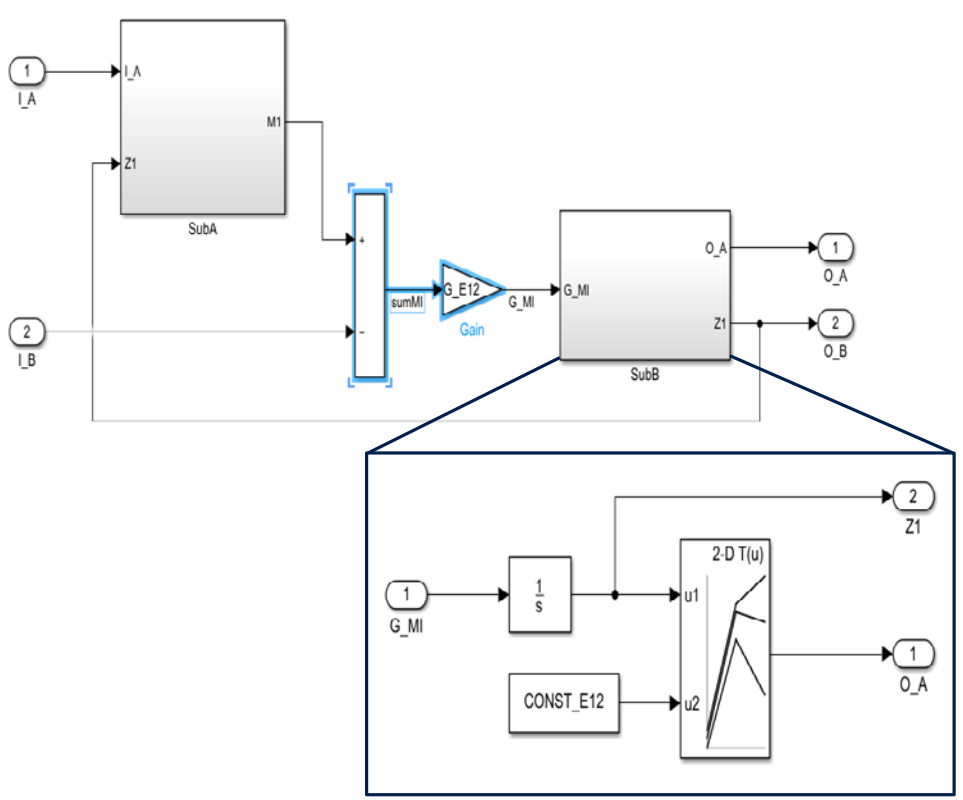
- Guideline: size/complexity on subsystem scope should be low

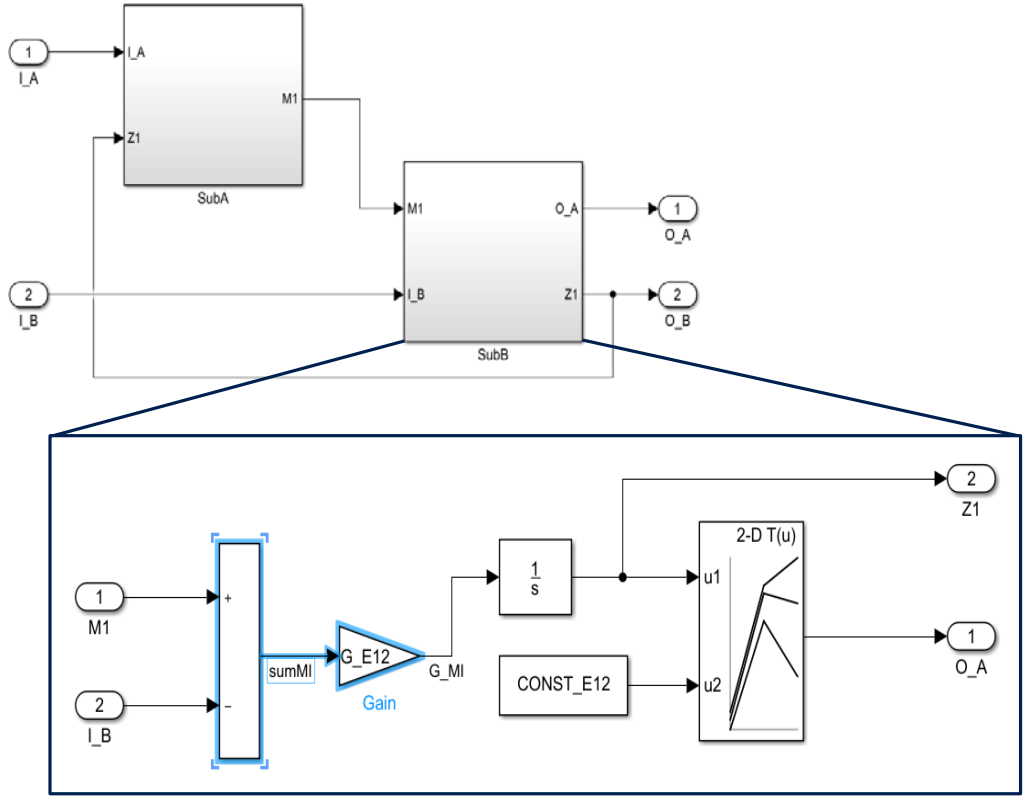




Other basic refactoring examples – Move Blocks Down

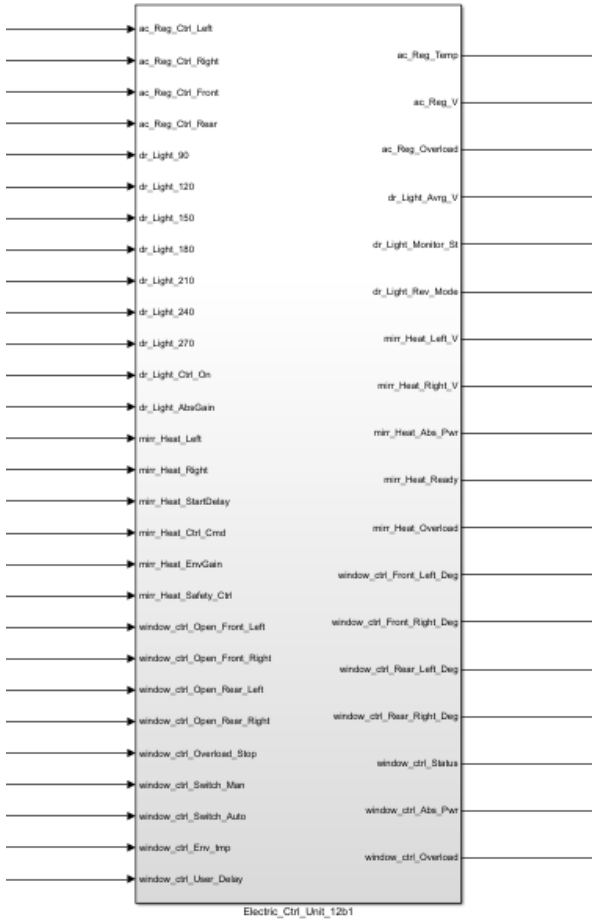
- Guideline: blocks of same type on single layer (subsystems or basic blocks)

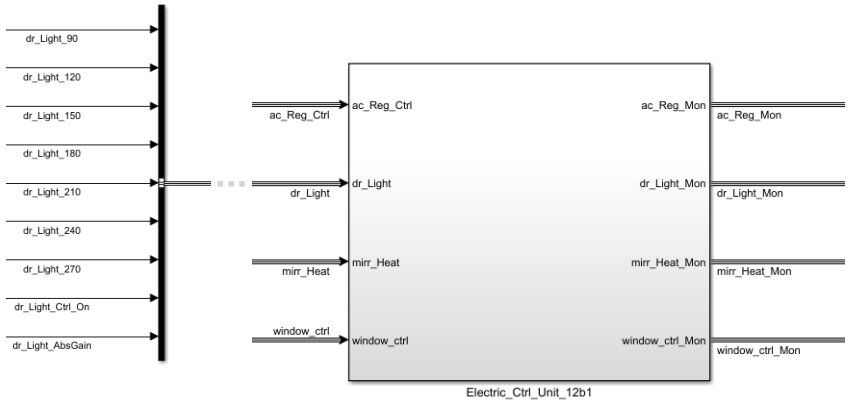




More complex refactoring examples – Convert Signals to Buses

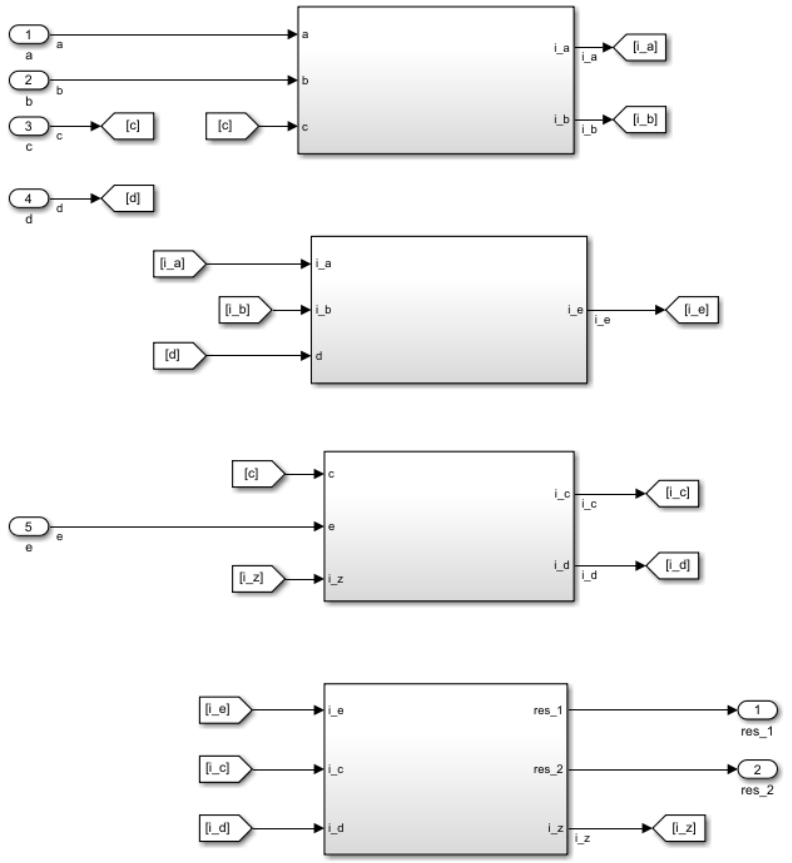
- Guideline: number of inports/outports should be restricted

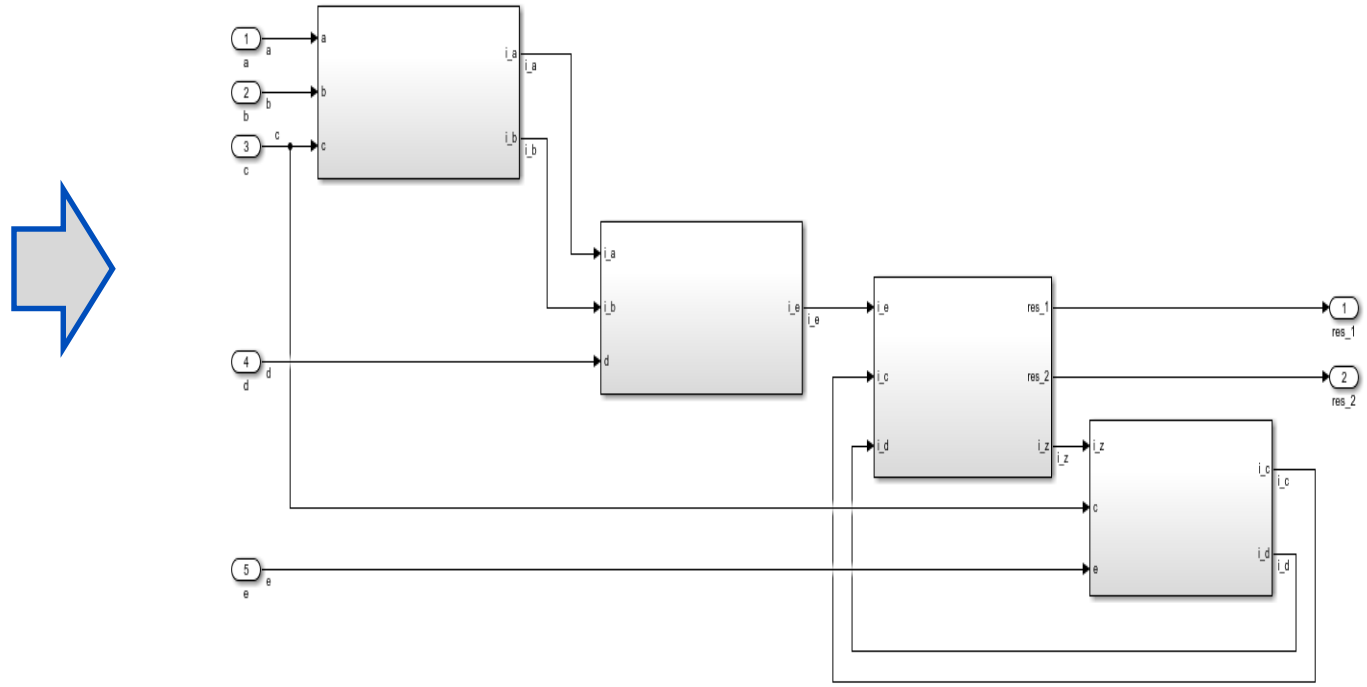




More complex refactoring examples – Resolve Goto/Froms + Rearrange

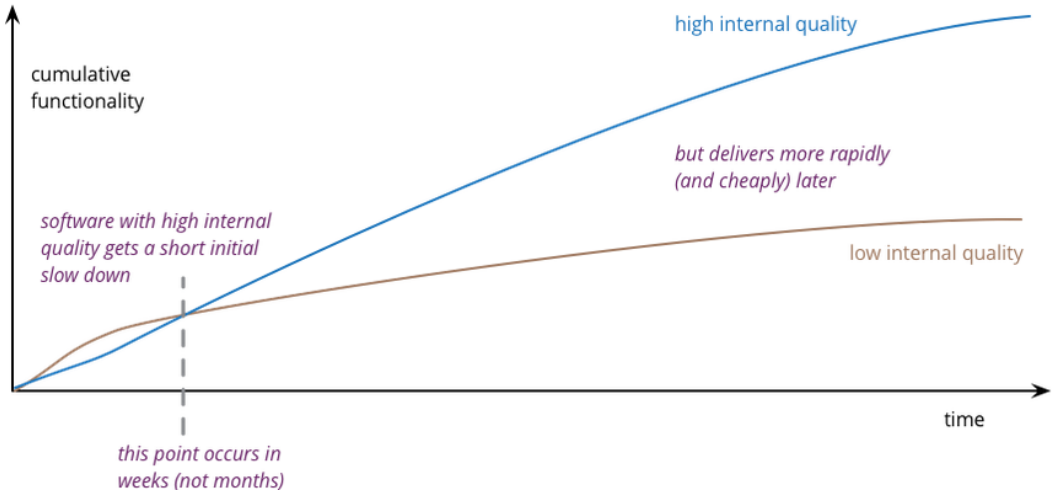
- Guideline: data flow should be obvious





Refactoring - Why

- "As an evolving program is continually changed, its complexity … increases unless work is done to maintain or reduce it."
  Meir M. Lehman, "Programs, Life Cylces, and Laws of Software Evolution"
- "The problem with quick and dirty is that the dirty remains long after the quick has been forgotten"
  Steve C. McConnell, "Software Project Survival Guide"
- "I stress that we should only approach it (refactoring) as an economic argument. High internal quality reduces the cost of future features, meaning that putting the time into writing good code actually reduces cost."
  Martin Fowler,
  "Is High Quality Software Worth the Cost?"



Refactoring - Why

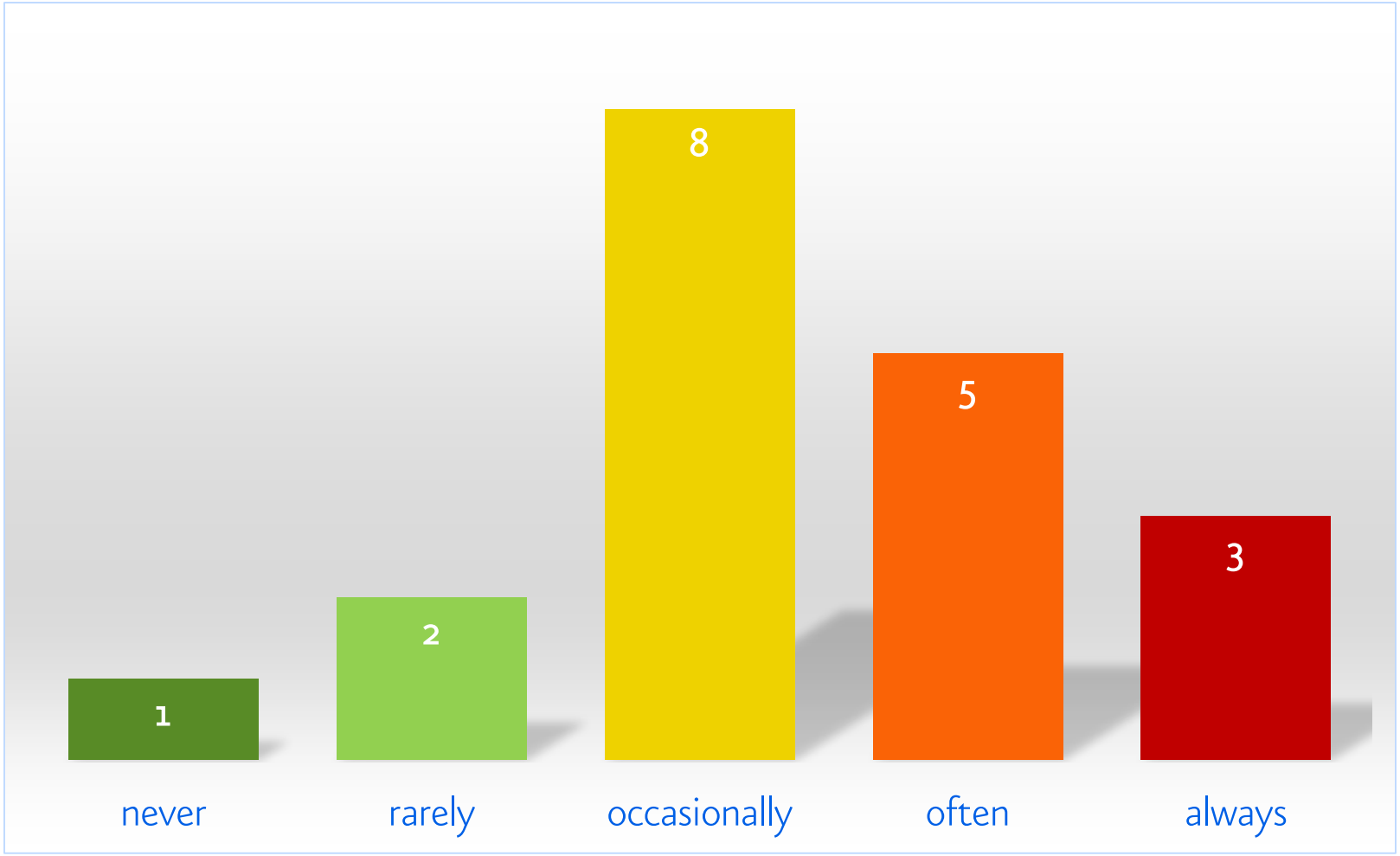
- Direct objectives:
  Model should be easier to …
  understand/read, modify, test

- Results from direct objectives:
  Models are easier / it is easier to …
  maintain, review, navigate, extend, reduce, locate bugs, avoid errors, …

- Other objective:
  Performance (memory/runtime)

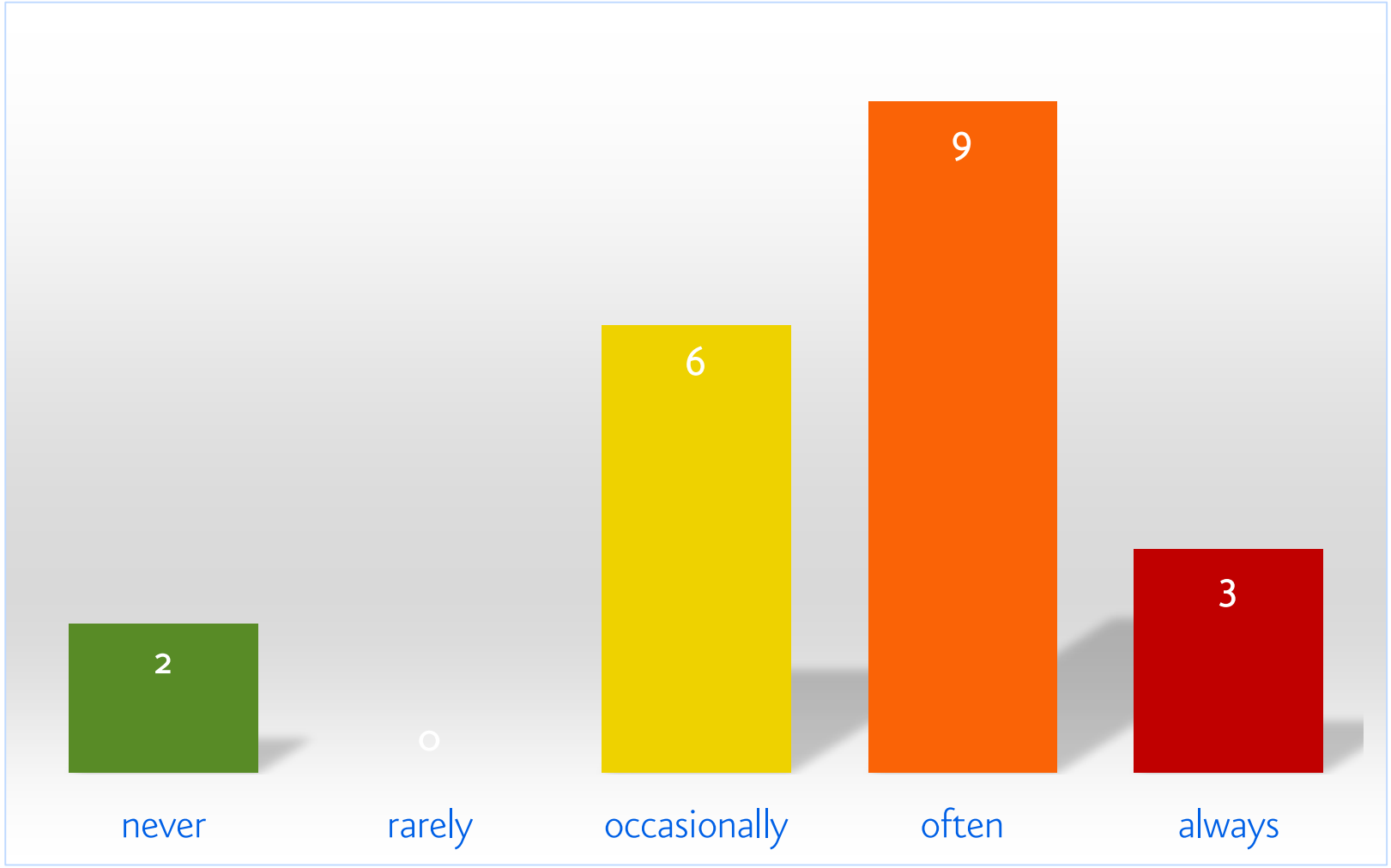
Refactoring – Why – Your feedback

- Do model readability and maintainability cause problems for you?



Refactoring – Why – Your feedback

- Does the structural quality of the models decrease during the development process?



Refactoring - When

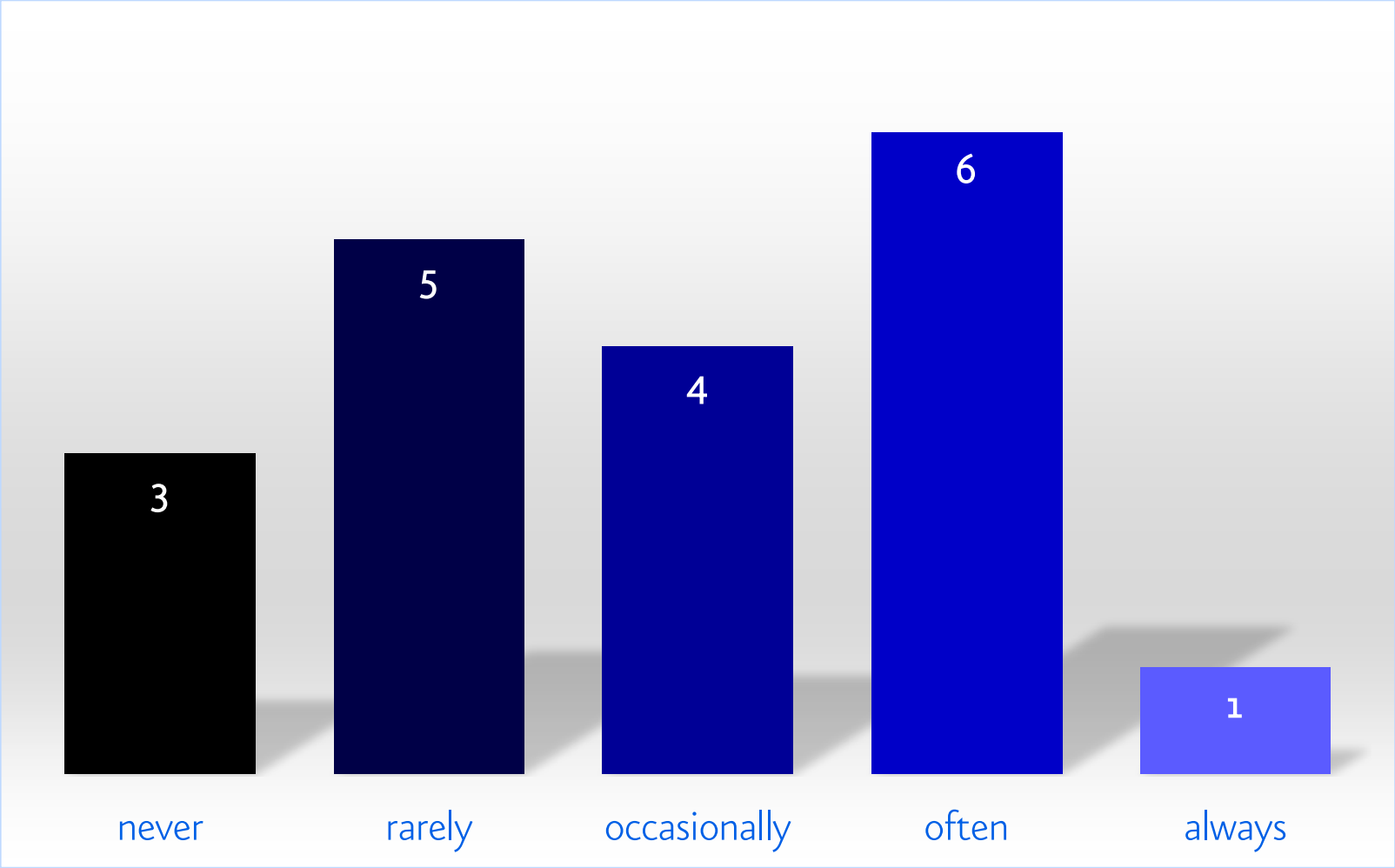
- **"By continuously improving the design of code, we make it easier and easier to work with."**
  Joshua Kerievsky, "Refactoring to Patterns"
- **"In my view refactoring is not an activity you set aside time to do. Refactoring is something you do all the time in little bursts."**
  Martin Fowler, "Refactoring: Improving the Design of Existing Code"

- **Recommendation: Refactoring is part of the modeling**
- **Why?**
  - Efficiency: Avoid technical debt instead of reducing it later
  - Efficiency: Avoid having to familiarize with the model again for refactoring
  - Justification: Small steps do not need extra justification

- **Refactoring can also be useful/necessary when start working on legacy code**

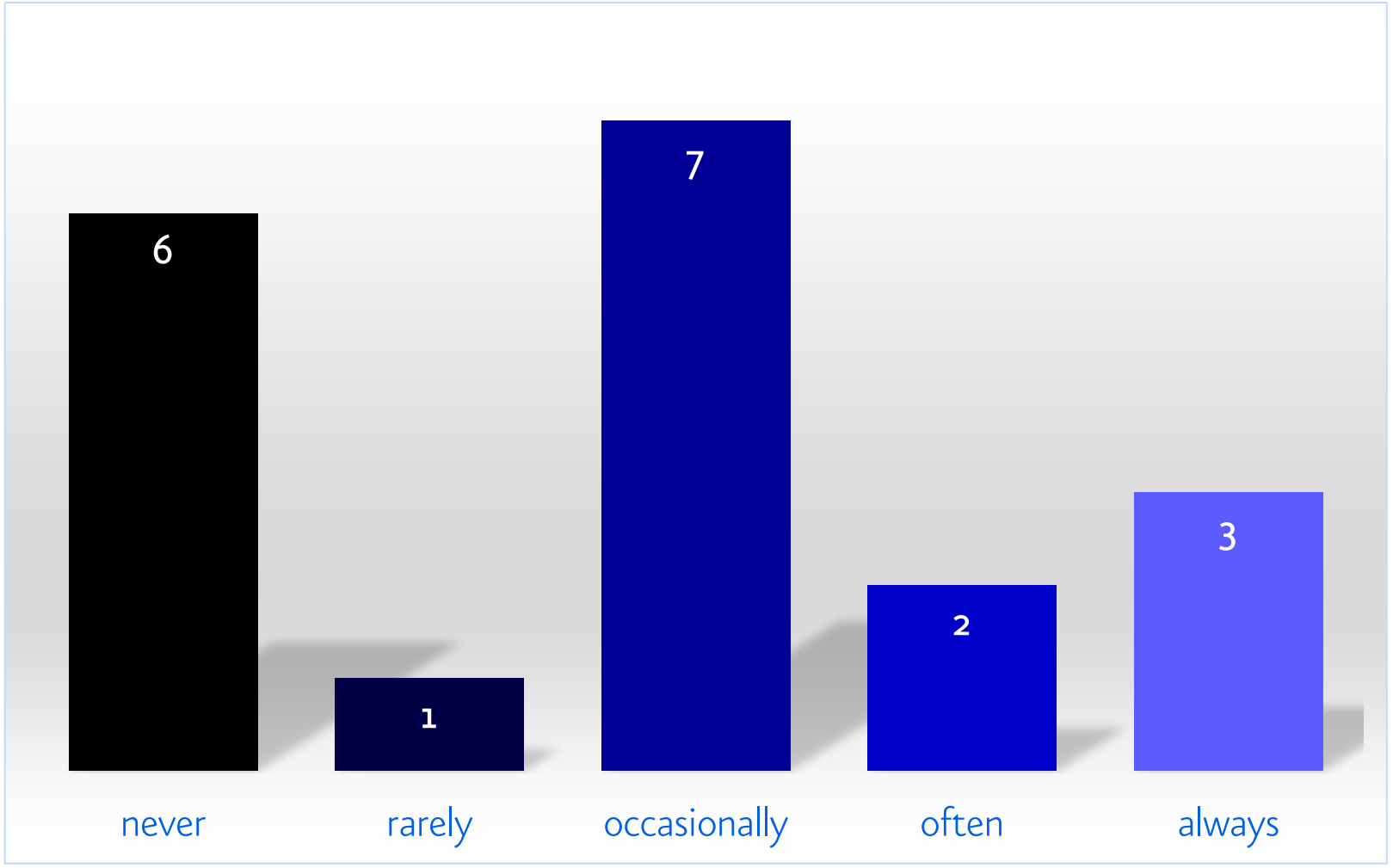
Refactoring – When – Your Feedback

- "Is refactoring part of your development process?"



Refactoring – When – Your Feedback

- "Is refactoring a dedicated development step for you at work?"



Refactoring - How

- "The key to effective refactoring is recognizing that you go faster when you take tiny steps."
  Martin Fowler, "Refactoring: Improving the Design of Existing Code"
- "Focus on progress, not perfection. Accept the fact that you will never be 100 percent satisfied."
  Sydney Stone, "Code Refactoring Best Practices", https://www.altexsoft.com
- "Good architecture and design are important; but the effect of a robust suite of tests is an order of magnitude greater. It's so much greater because those tests enable you to improve the design."
  Robert C. Martin, "So… You want your code to be maintainable."

Refactoring - How

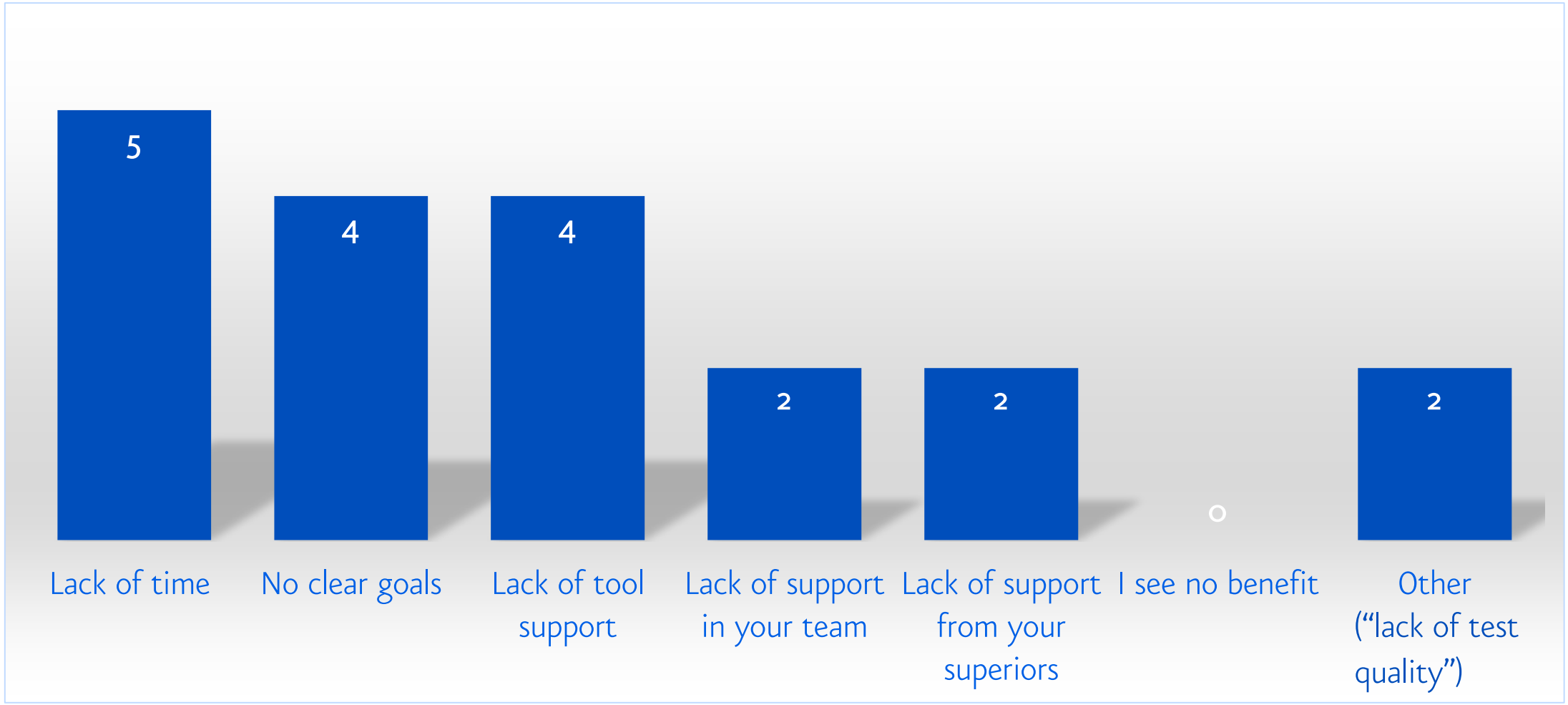
- **Prerequisite: Unit tests**

- **Define common objectives (team/company)**
  - Modeling guidelines should be met
  - Peer review should be passed

- **Tools**
  - Software: Test tool
  - Software: Guideline checker (with repair)
  - Software: Refactoring tools
  - Experience with basic refactoring operations

Refactoring – practical problems

- Which of these problems do you encounter during refactoring?



Refactoring - practical problems

- **When it comes to refactoring, what is the biggest problem you face?**

"Lack of test quality"

"To ensure/verify the functionality of the software after refactoring -> high risk -> low motivation for refactoring"

"Lack of understanding of the software"

"Logical distribution of functional requirements"

"Restructuring of the design"

"Maintaining readability and complexity"

"Inconsistency design pattern and model configuration"

"Previous resolved issues are no longer used/ relevant"

"Workflow"

"Refactoring based on ISO-262606 and following general guide lines of model base development"

"We are not well educated on that topic"

"Time and resource"

"To communicate the reasons for eliminating technical debt"

"Support from team and management"

"Some engineers follow either no legibility guidelines, or worse, their own nonsensical rules"

Refactoring - practical problems

- **Which refactoring steps take the most time (e.g. layout optimization, renaming, restructuring, etc.)?**

"Restructuring"

"Restructuring"

"Restructuring"

"Restructuring"

"Restructuring of the design "

"Renaming"

"Renaming"

"Renaming"

"Layout optimization"

"Layout"

"Layout"

"Layout optimization without snap-to-grid for signal lines"

"Moving bus selectors inside subsystems"

"Restructuring from Matlab script to Simulink blocks"

"Optimization"

"Verification"

"Routing new / deleting old signals through many layers"

"Model base design in general"

"Memory Layout"

Open discussion

Possible topics:

- What would you like to talk or learn more about?
- How do you practice refactoring?
- Do you want to share your experience?
- Do you see problems in practice?
- Do you have best practices or tips?

Live Webinars & Videos

- Check out our upcoming live webinars & our video content on our website

Click here for
Live Webinars

Click here for
Videos

Stay Connected:

model-engineers.com/mgigroup/

model-engineers.com/linkedin/

model-engineers.com/youtube/

model-engineers.com/facebook/

model-engineers.com/xing/

WeChat Official Account

MES
MODEL ENGINEERING SOLUTIONS

Model Engineering Solutions GmbH

Waldenserstraße 2 - 4
10551 Berlin
Germany

+49 30 2091 6463-0
+49 30 2091 6463-33
info@model-engineers.com
www.model-engineers.com