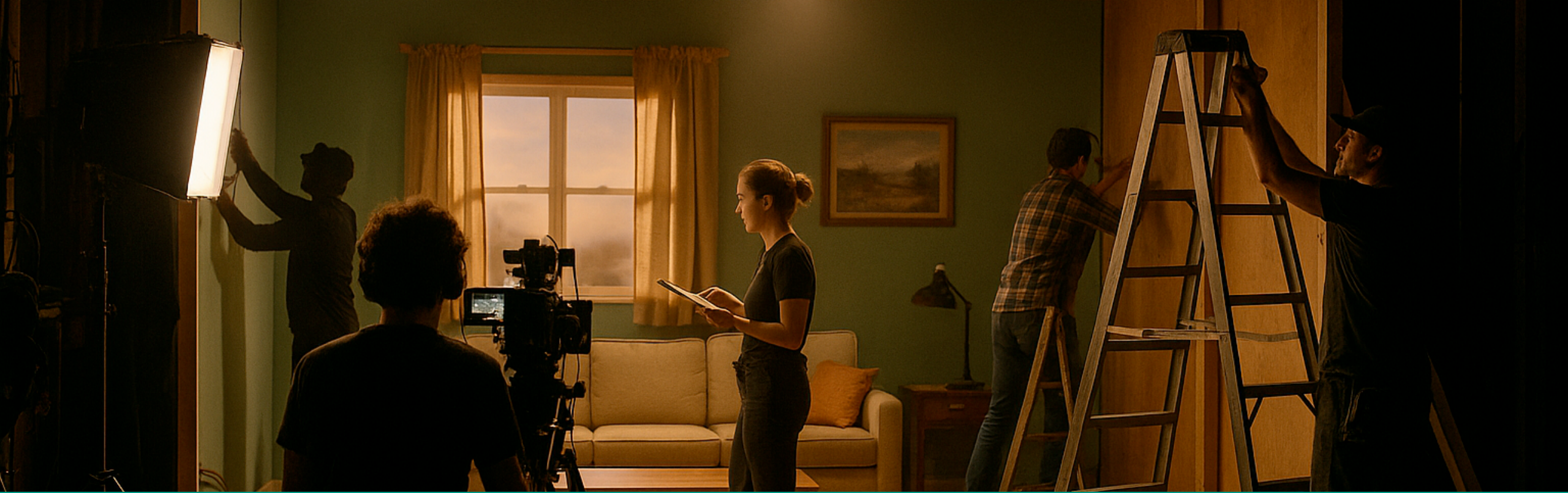


Collaborative Software Design

Introduction

Manuel Giffels (Manuel.Giffels@kit.edu)
KSETA Course — October 2025

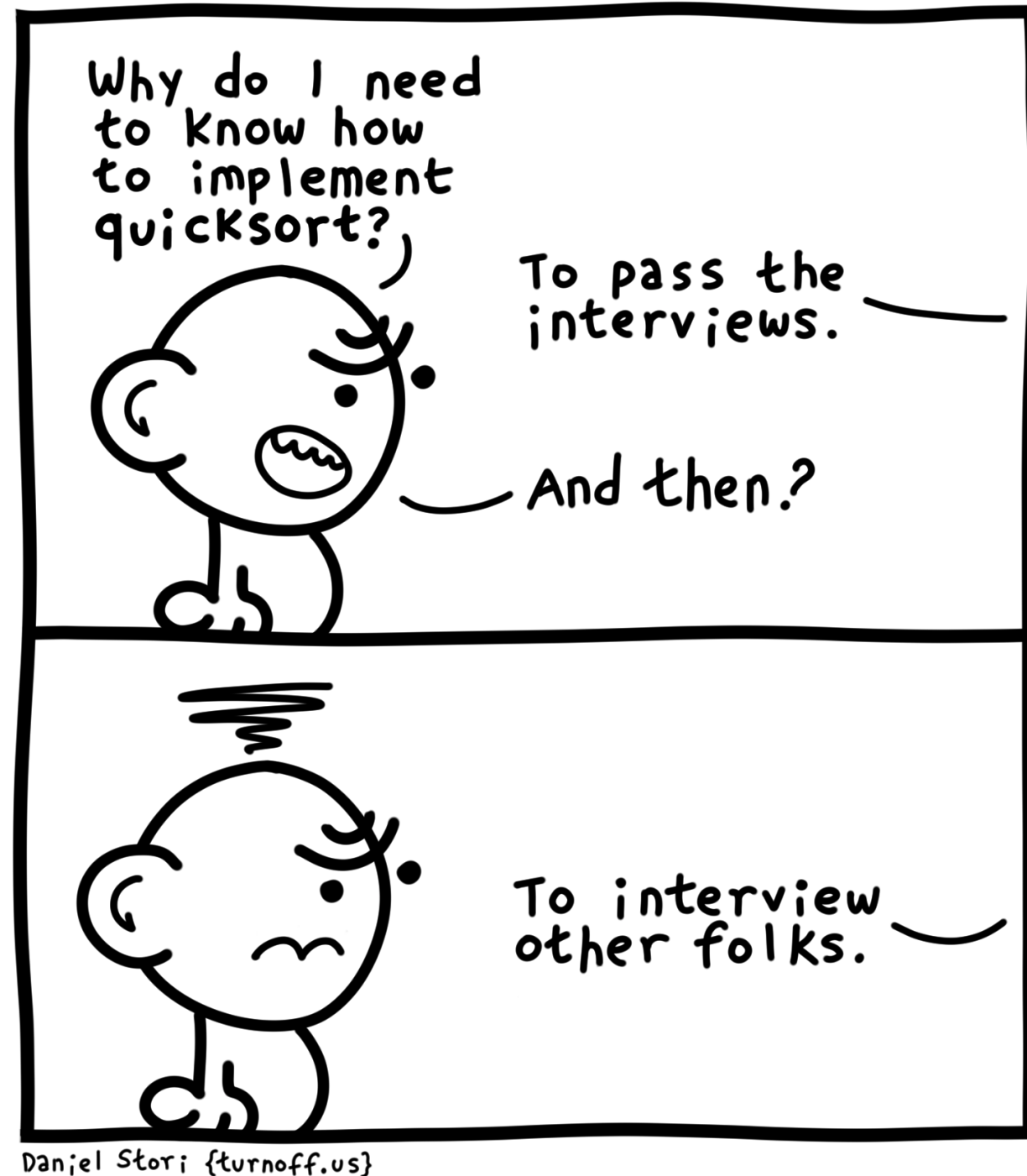




Setting the Scene

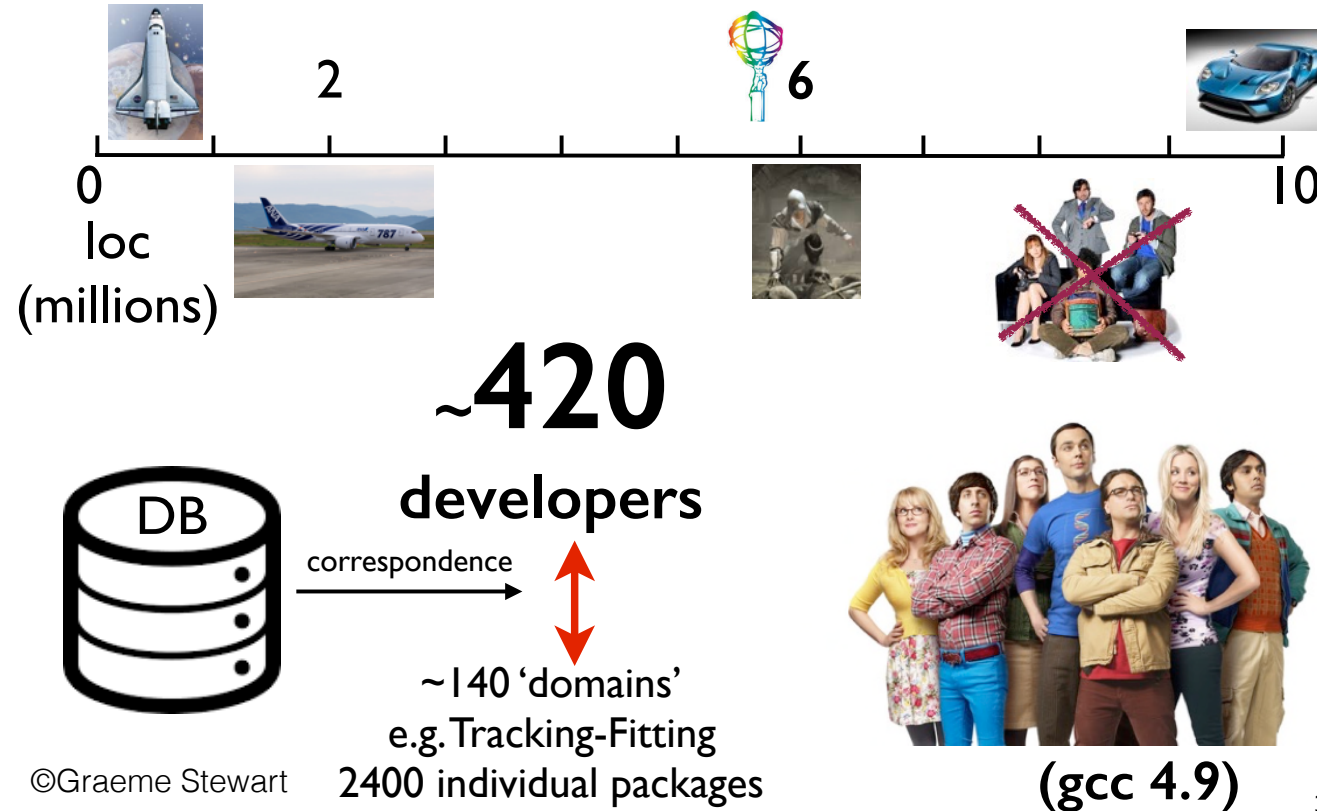
Why to learn about Collaborative Software Design?

- What is your motivation to be here?
- What kind of code are you working on?
- Do you use already Collaborative Software Design Tooling?
 - Git
 - Agile Development
 - Unittesting
 - Continuous Integration / Continuous Deployment (CI/CD)
 - Test Driven Development



Research Software Today

ATLAS code



Impossible to achieve without Collaborative Software Design Tooling

The Schedule

Day 1

- **Introduction & Motivation**
Why Collaborative Software Practices are essential in Research & Engineering
- **Git & Workflows**
 - Git essentials: clone, branch, commit, merging, pull request
 - Collaborative workflows: Feature Branches, GitHub/GitLab Flow
 - Hands-on
- **Agile Development**
 - Software Development Methods
 - Extreme Programming, Scrum, User Stories, etc.
 - Hands-on

Day 2

- **Unit Tests & Test-Driven Development**
 - Unit Tests
 - Test Coverage
 - Test Driven Development
 - Hands-on
- **Continuous Integration/Continuous Deployment**
 - Advantages of using CI/CD
 - Tools
 - Hands-on
- **Documentation Basics**
 - Types of documentation and target audiences
 - Tools & workflows
 - Best practices
 - Hands-on

Disclaimer

- This course provides a high-level overview of collaborative software design topics
- It is intended to introduce the tools, methods, and concepts used in the field
- The course is not intended to be a comprehensive or complete course, we can cover only the major components
- The course does not provide exhaustive tutorials or detailed instructions for every tool or technique
- Students are encouraged to explore specific tools and practices in depth through further study
- The goal is to give you a broad understanding of what exists and how collaborative software design can be approached