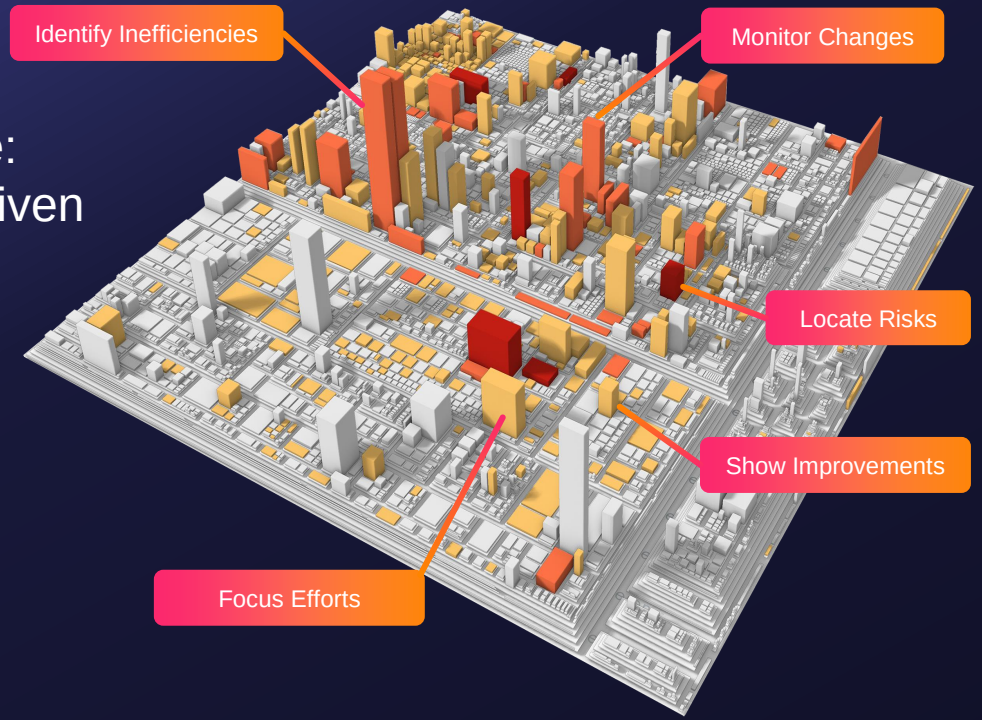
seerene
The Digital Boardroom

Towards highest Software Excellence: Using Analytics to establish a data-driven Continuous Improvement Process



Dr. Johannes Bohnet
Founder & Co-CEO
Seerene GmbH

September 22nd, 2022, Conference in Potsdam
Automotive Software Factory
Learning Curves

DEEP DIVE (FILM)
Why software executives need actionable insights
https://www.youtube.com/watch?v=6dtqTChsfN0

Learning Curves in Software Development Organizations

Seerene observes trends and challenges across industries

- Tailor-made software is a growing strategic necessity
- Companies must build up software development competence
- Software developers are a scarce resource
  → EFFICIENT use of developer time is key

Learning Curve

How to measure and optimize software development efficiency?

Off-the-shelf Software
Purchased from tool vendors
(Microsoft Office, Webex, …)

Software Factories
Teams that build tailor-made software systems

Software in Automotive Industry

Different types of software systems and associated key challenges

Applications/Services/Products
“outside” the car

Learning Curve:

- From 3rd-party projects to long-life agile product development.
- Building up in-house software development competence.
- Closing the Biz | DevOps gap in scaled agile organizations.

Software in Automotive Industry

Different types of software systems and associated key challenges

Applications/Services/Products “outside” the car

Learning Curve:

- From 3rd-party projects to long-life agile product development.
- Building up in-house software development competence.
- Closing the Biz | DevOps gap in scaled agile organizations.

Embedded Software

Learning Curve:

- From project-to-project thinking to generic platforms and high code reuse.

Software in Automotive Industry

Different types of software systems and associated key challenges

Applications/Services/Products “outside” the car

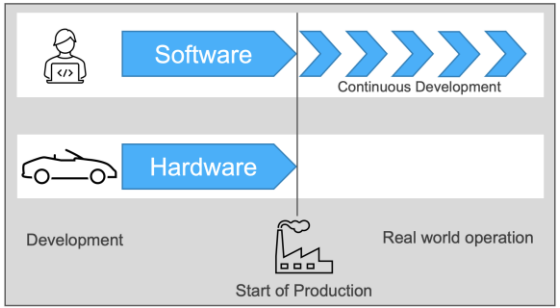
Learning Curve:
- From 3rd-party projects to long-life agile product development.
- Building up in-house software development competence.
- Closing the Biz | DevOps gap in scaled agile organizations.

Embedded Software

Learning Curve:
- From project-to-project thinking to generic platforms and high code reuse.

Embedded Software for software-defined vehicles

Learning Curve:
- Decoupling software from hardware.
- Sophisticated variant management for code components.



Optimizing Production Efficiency

Measuring KPIs as basis for optimization

Hardware Production

Established KPIs exist:

production volume, production downtime, production costs, overall operations effectiveness (OOE), overall equipment effectiveness (OEE), total effective equipment performance (TEEP), capacity utilization, throughput, defect density, maintenance costs, ...

Optimizing Production Efficiency

Measuring KPIs as basis for optimization

Hardware Production

Established KPIs exist:

production volume, production downtime, production costs, overall operations effectiveness (OOE), overall equipment effectiveness (OEE), total effective equipment performance (TEEP), capacity utilization, throughput, defect density, maintenance costs, ...

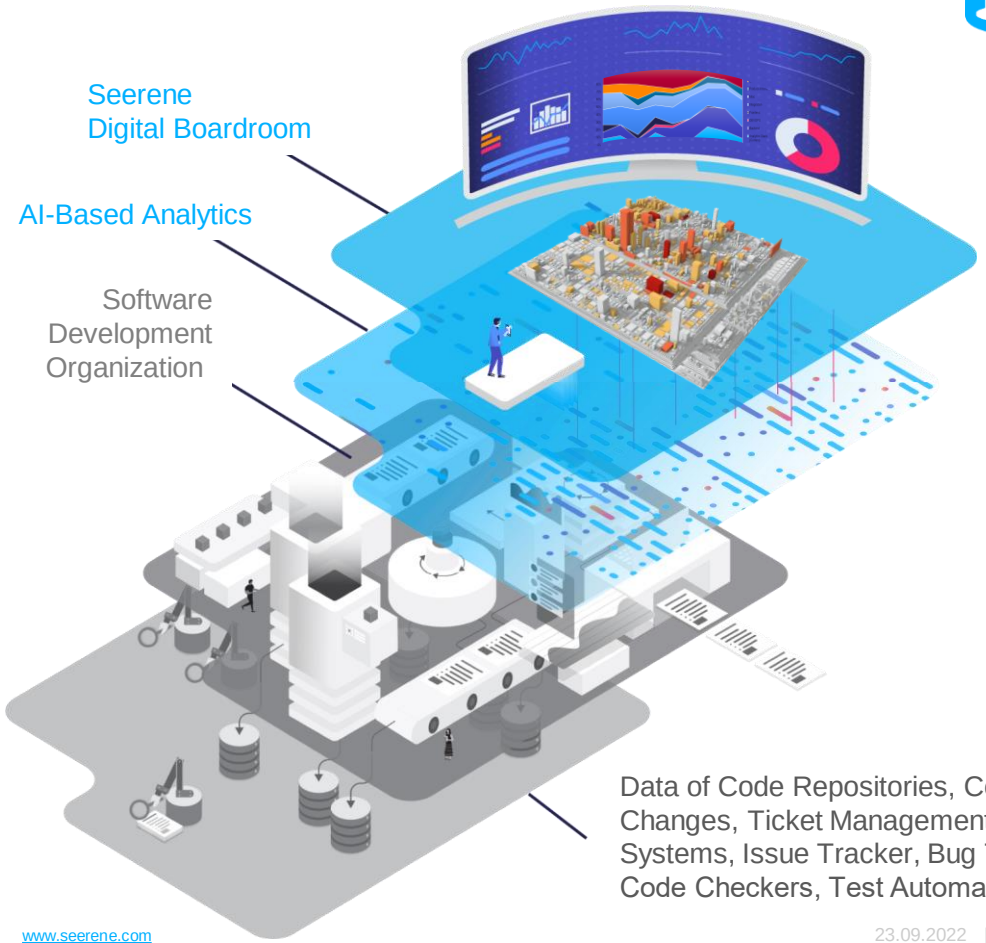
Software Production

Established KPIs and measurement approaches do not exist yet:

Analyzing code is too simplistic to reflect coding reality.

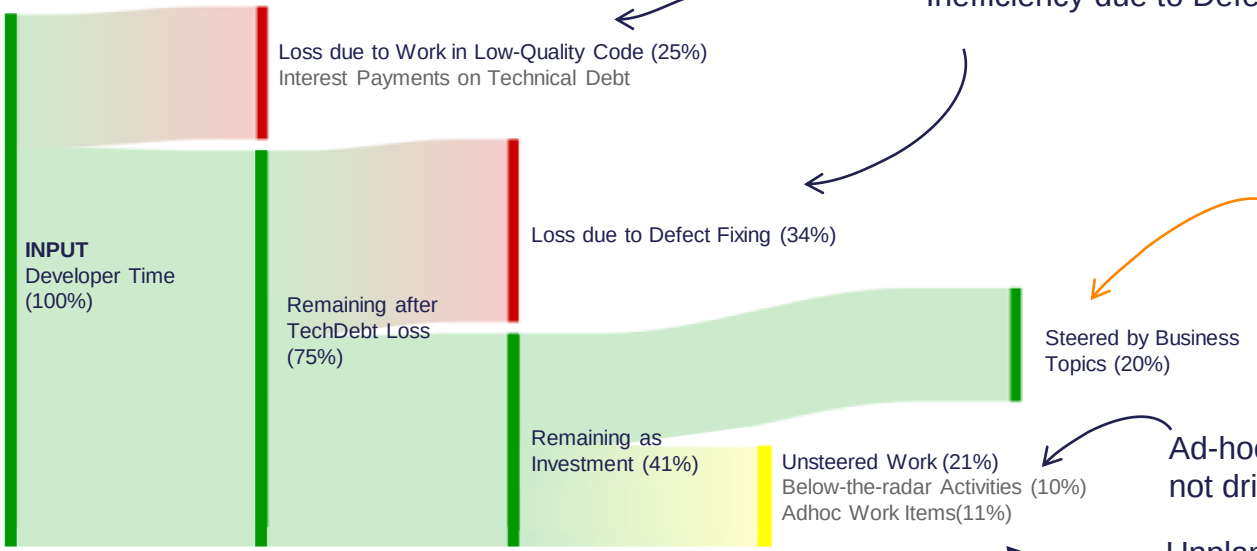
Making use of the already existing data in the software development infrastructure

- Data-driven Analysis of actual Software Development Activities
- No Interference to Methodologies, Developer Tools & Processes
- Disruptive Use of existing Software Development Data such as Commits and Tickets
- Reports on Trends, Risks, State, Business Criticality, and spent Budgets
- Activating Self-Reflection across all Stakeholders, creating Awareness



Data-Driven Analysis how Developer Time is used

Are developer resources in terms of time and money used efficiently?



Inefficiency due to Low-Quality Code

Inefficiency due to Defect Fixing

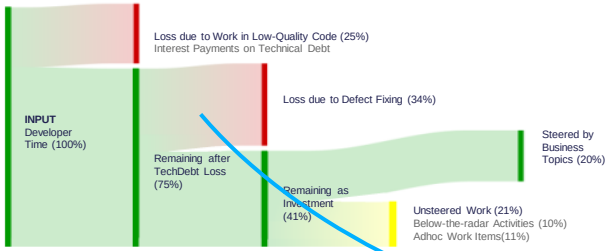
Only a small Fraction of Developer Time is used for implementing Business Topics!

Ad-hoc Coding that is not driven by any Work Item

Unplanned Coding Activities that do not fit in the bigger Picture

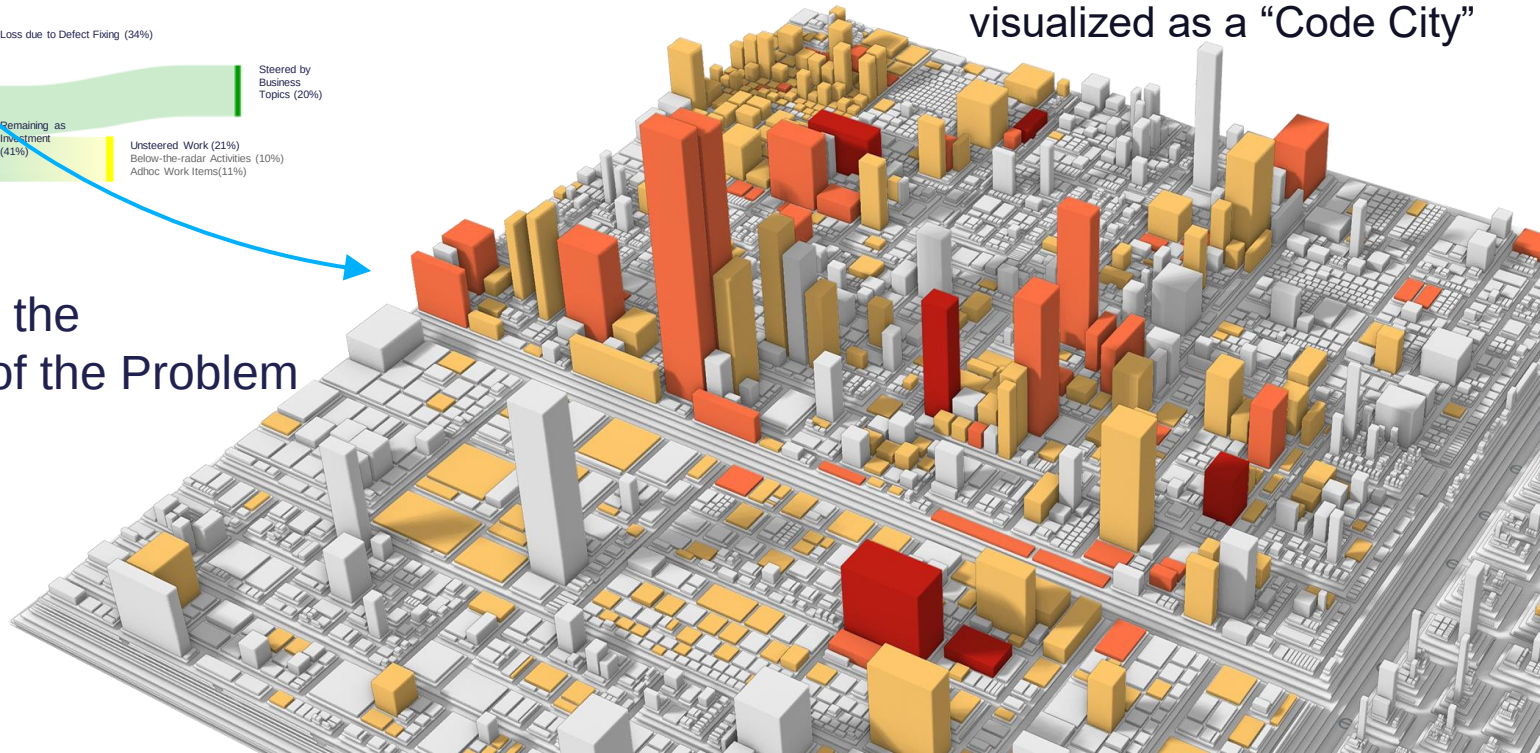
From Inefficiency KPIs to Root Causes

Finding and fixing code that unnecessarily consumes developer time



Navigating to the
Root Cause of the Problem

Code Landscape
visualized as a “Code City”

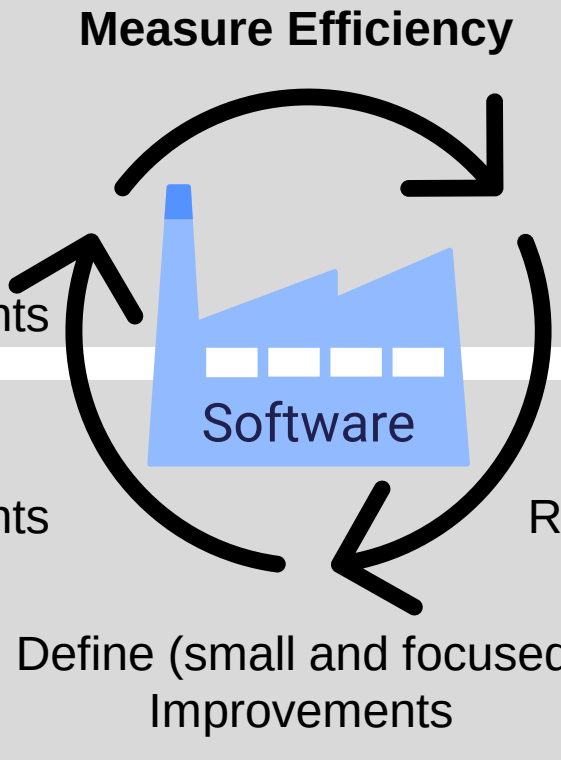


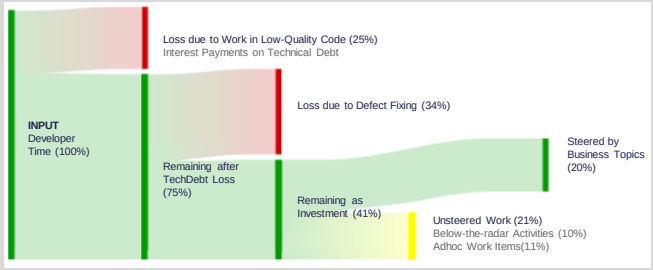
Analytics-Driven Continuous Improvement Cycle

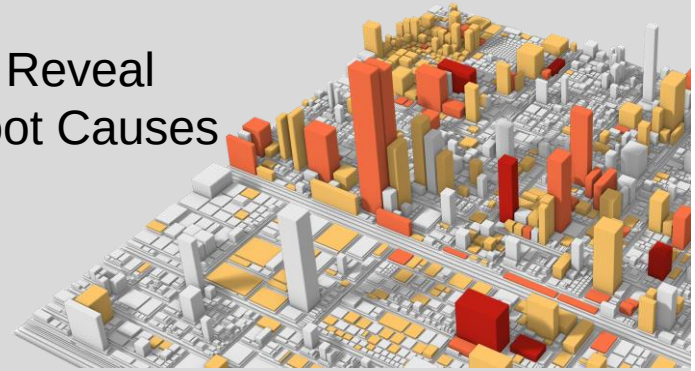
Key is to close the loop and measure the impact of improvement activities

Strategic

Operational







Software in Automotive Industry

Different types of software systems and associated key challenges

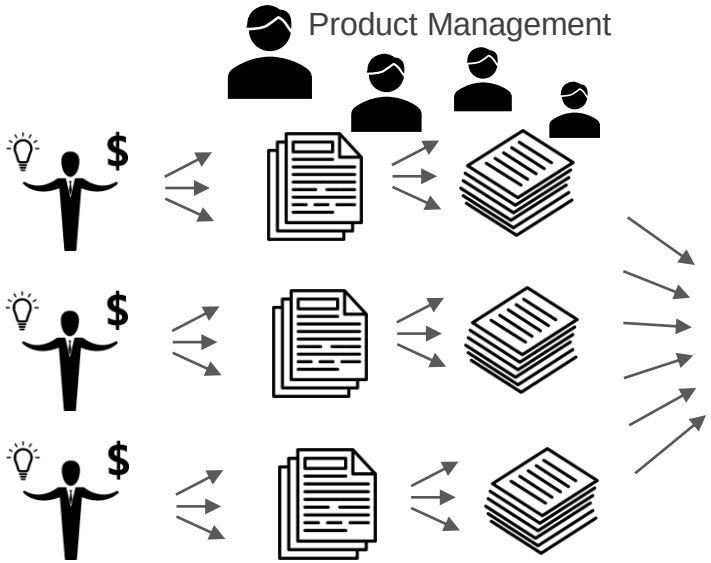
Applications/Services/Products
“outside” the car

Learning Curve:

- From 3rd-party projects to long-life agile product development.
- **Building up in-house software development competence.**
- **Closing the Biz | DevOps gap in scaled agile organizations.**

Anatomy of an Agile Software Development Organization

The steepest learning curve is on the “Biz” side of the organization



High-Level Business Topics

Feature (larger work items)

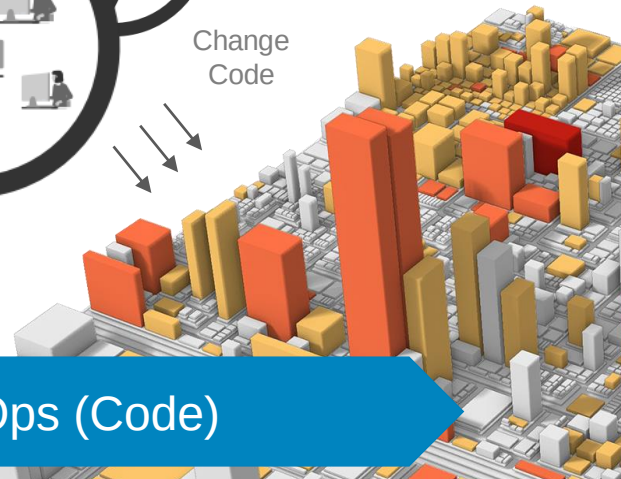
Stories/Tasks (work items)

Backlog of Work Items

Developers

Change Code

Source Code (visualized as city)

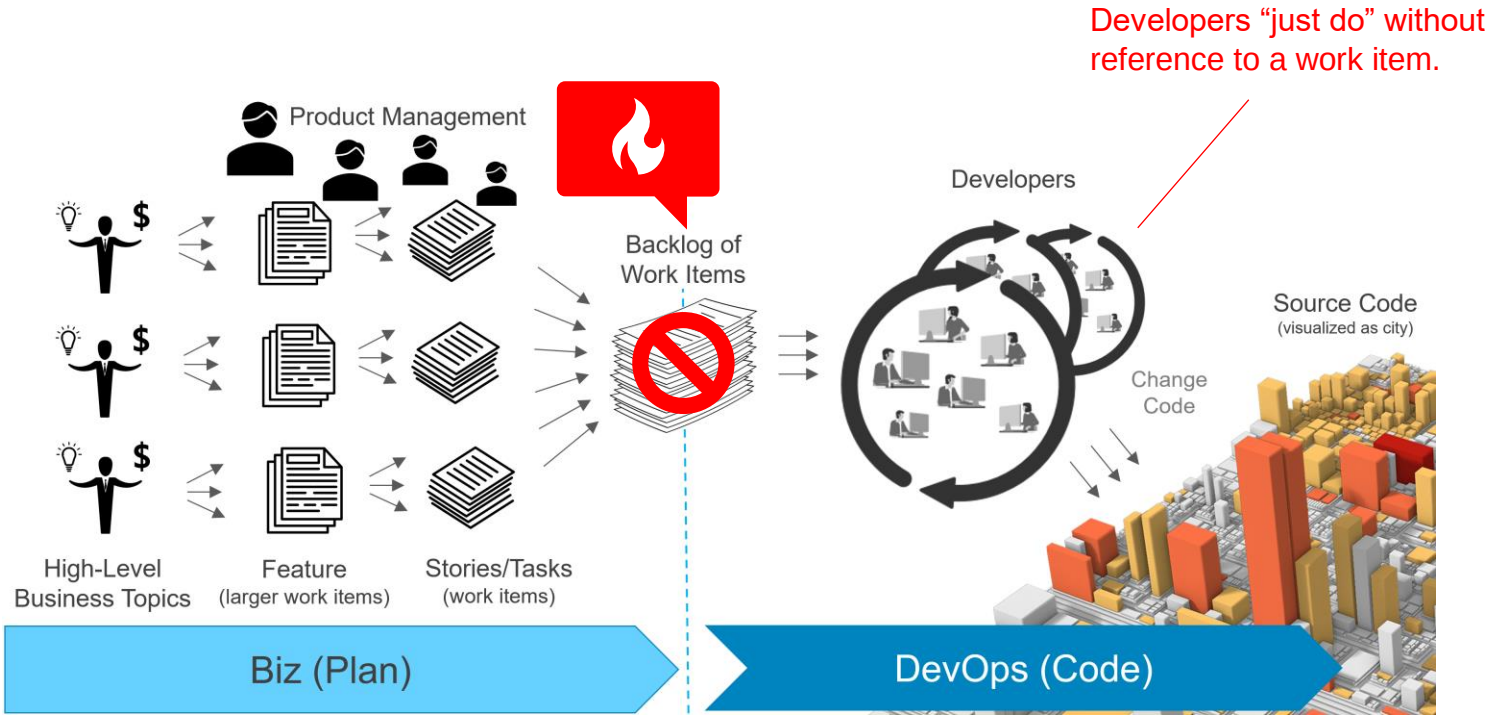


Biz (Plan)

DevOps (Code)

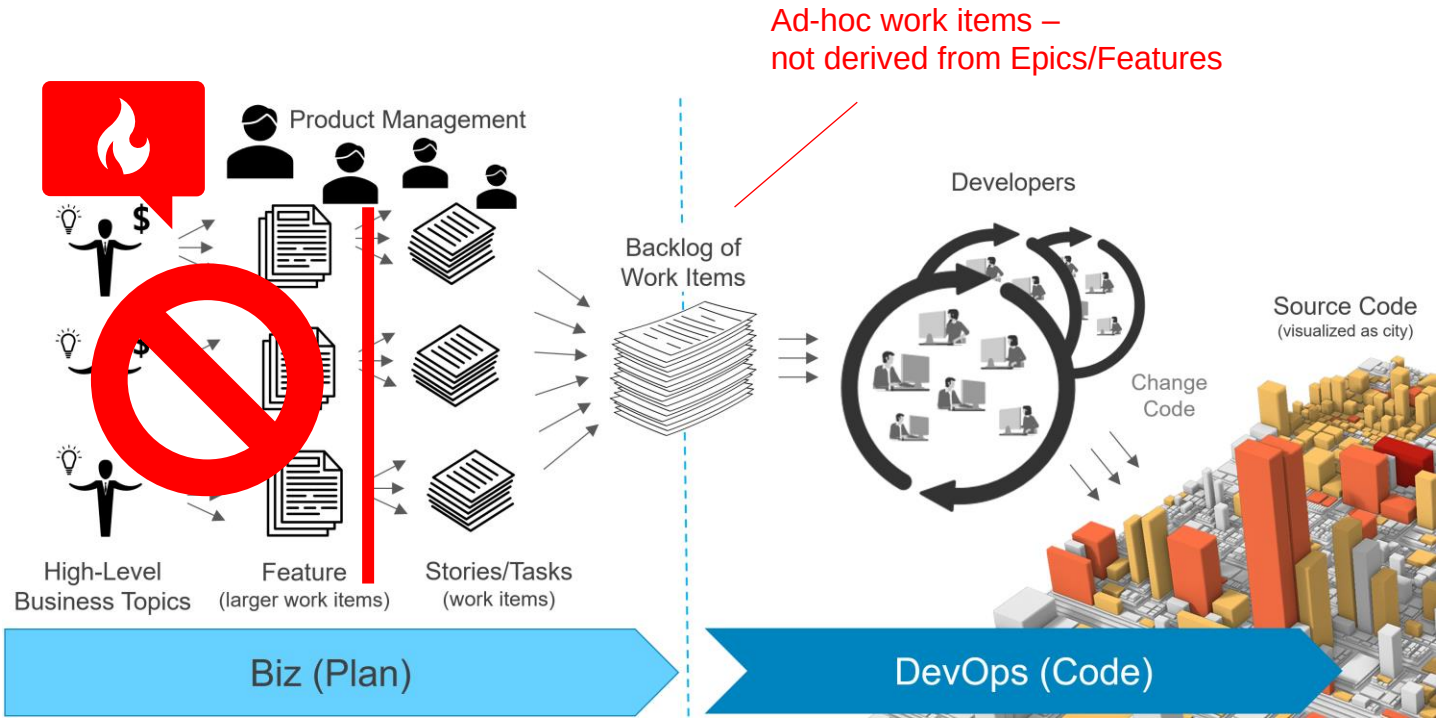
Workflow between Business and Tech side is Broken

Product Owners and Software Engineers do not work as a team



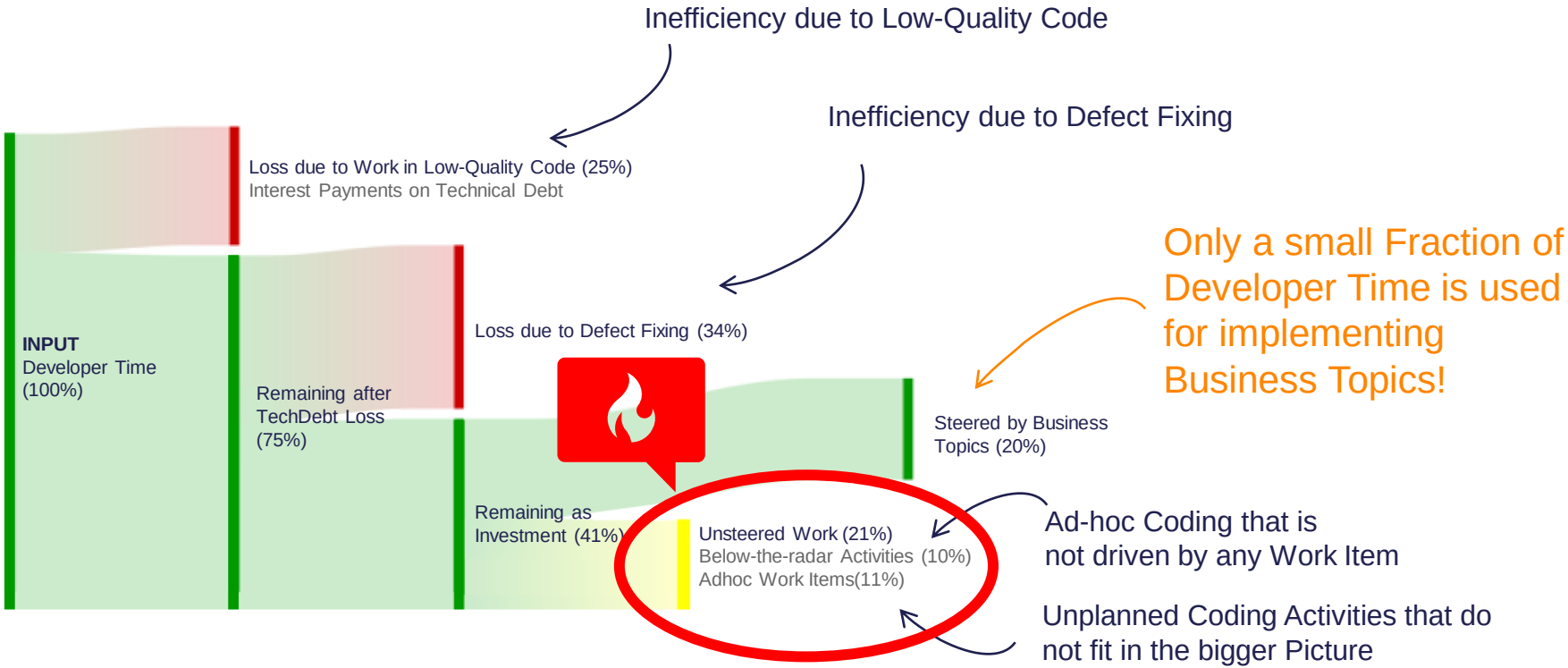
Business Side does not Maintain a Big Picture

Product Owners fill backlogs in an ad-hoc manner



Measuring the Inefficiencies due to Low-Mature Agile Processes

Establishing a continuous improvement cycle to increase agile maturity



Software in Automotive Industry

Different types of software systems and associated key challenges

Embedded Software

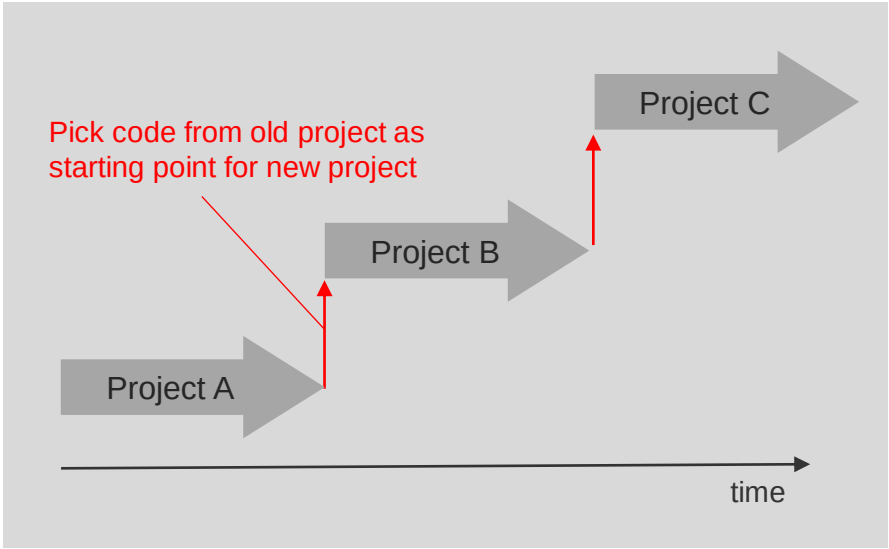
Learning Curve:

- From project-to-project thinking to generic platforms and high code reuse.

Delivering Customer-/Hardware-specific Software

Code reuse as key for efficient project delivery

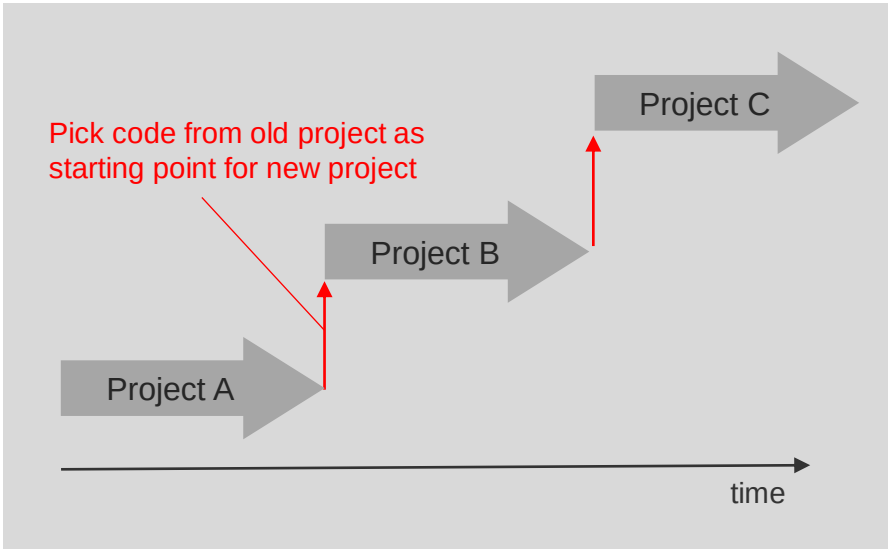
Innovating from Project to Project



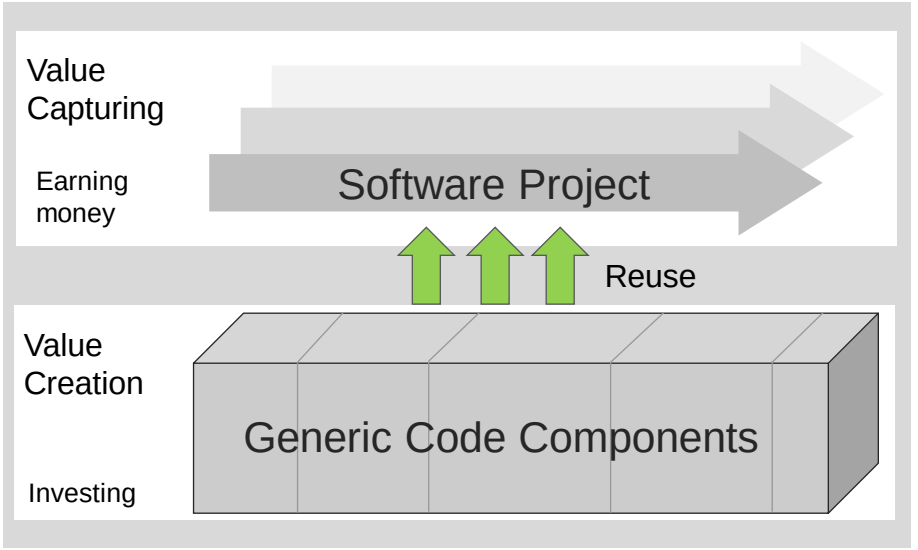
Delivering Customer-/Hardware-specific Software

Code reuse as key for efficient project delivery

Innovating from Project to Project



Systematic Reuse Approach

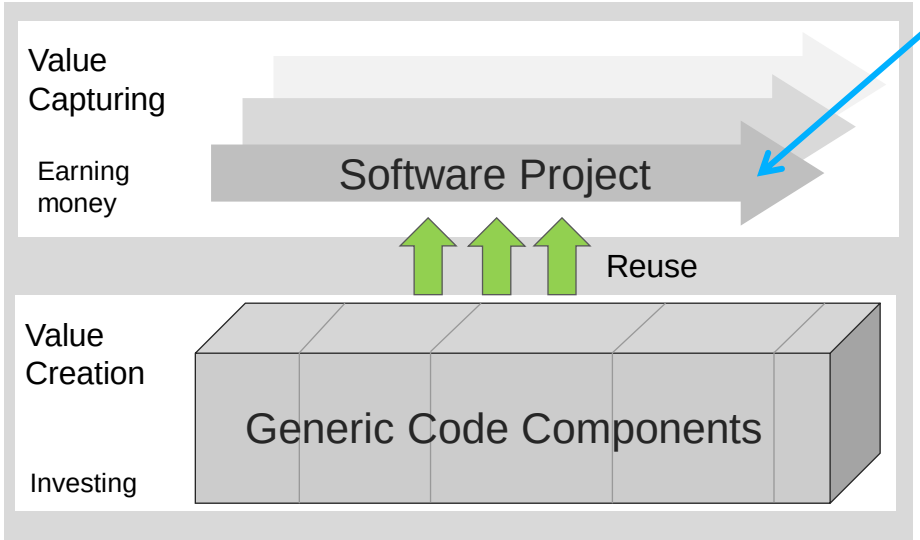


Applying Analytics to Measure and Increase Code Reuse

Code map views reveal concrete improvement actions

The Perfect Project

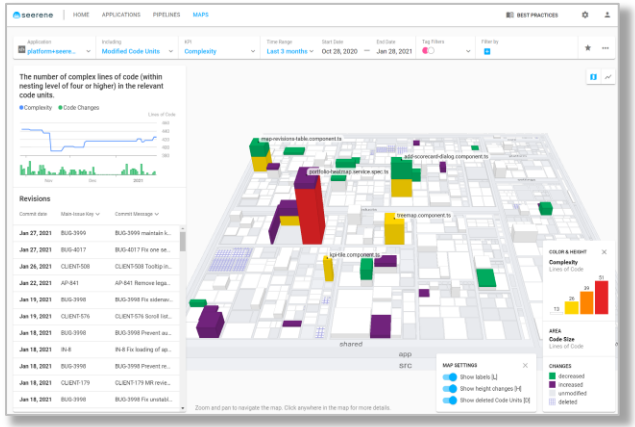
- takes pre-manufactured code components
- does not require significant coding effort



Measure project-specific coding effort.

Reveal code units that are reshaped in project context.

→ These code units lack reusability.



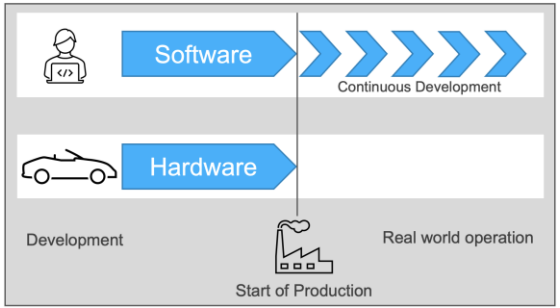
Software in Automotive Industry

Different types of software systems and associated key challenges

Embedded Software for software-defined vehicles

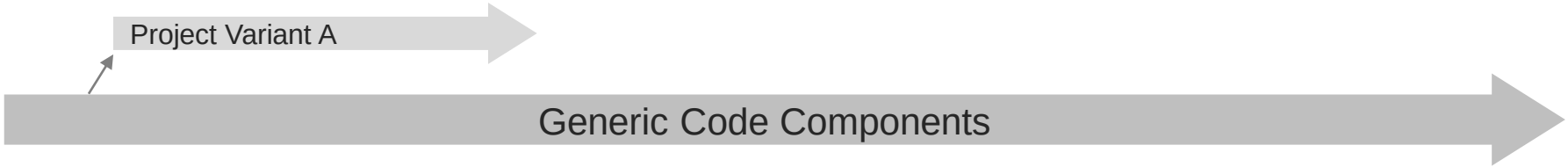
Learning Curve:

- Decoupling software from hardware.
- Sophisticated variant management for code components.



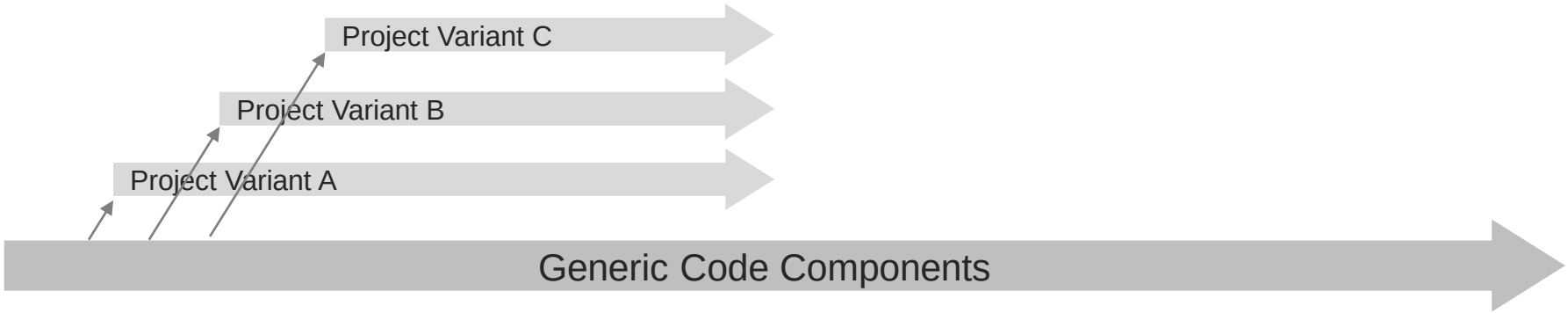
Deviating Project Variants Lead to a Cost Explosion

Additionally, the stream of innovation is blocked



Deviating Project Variants Lead to a Cost Explosion

Additionally, the stream of innovation is blocked

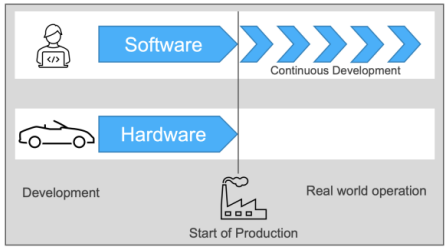


Deviating Project Variants Lead to a Cost Explosion

Additionally, the stream of innovation is blocked

Software-defined Vehicle:

→ Project variants live forever



Project Variant H

Project Variant G

Project Variant F

Project Variant E

Project Variant D

Project Variant C

Project Variant B

Project Variant A

Generic Code Components

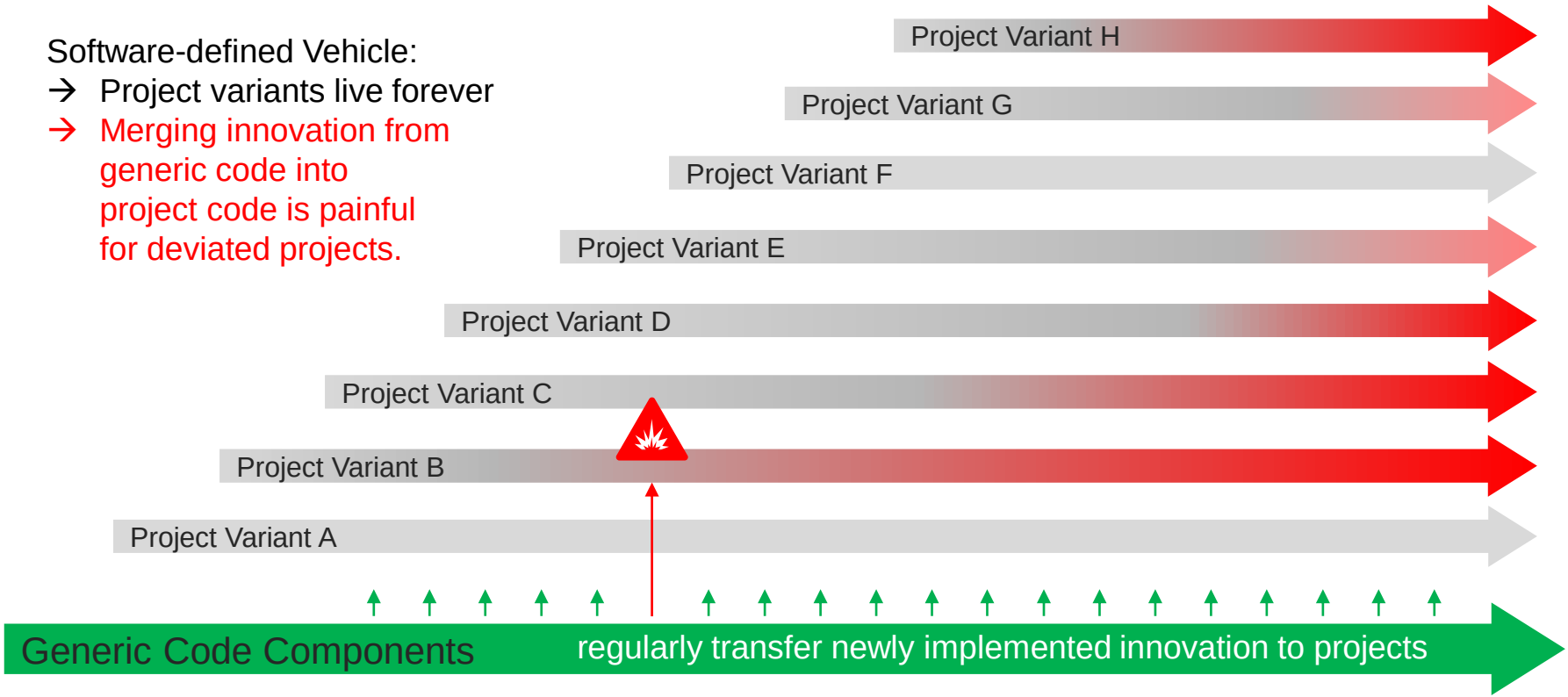
Deviating Project Variants Lead to a Cost Explosion

Additionally, the stream of innovation is blocked

Software-defined Vehicle:

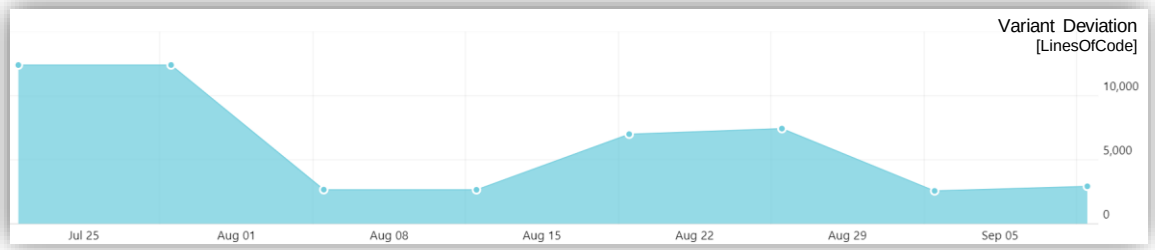
→ Project variants live forever

→ Merging innovation from generic code into project code is painful for deviated projects.



Analytics-Based Measurement of Project Variant Deviation

Systematically converging project variants with generic components



Converge Project Variant with Generic Components

Project Variant X

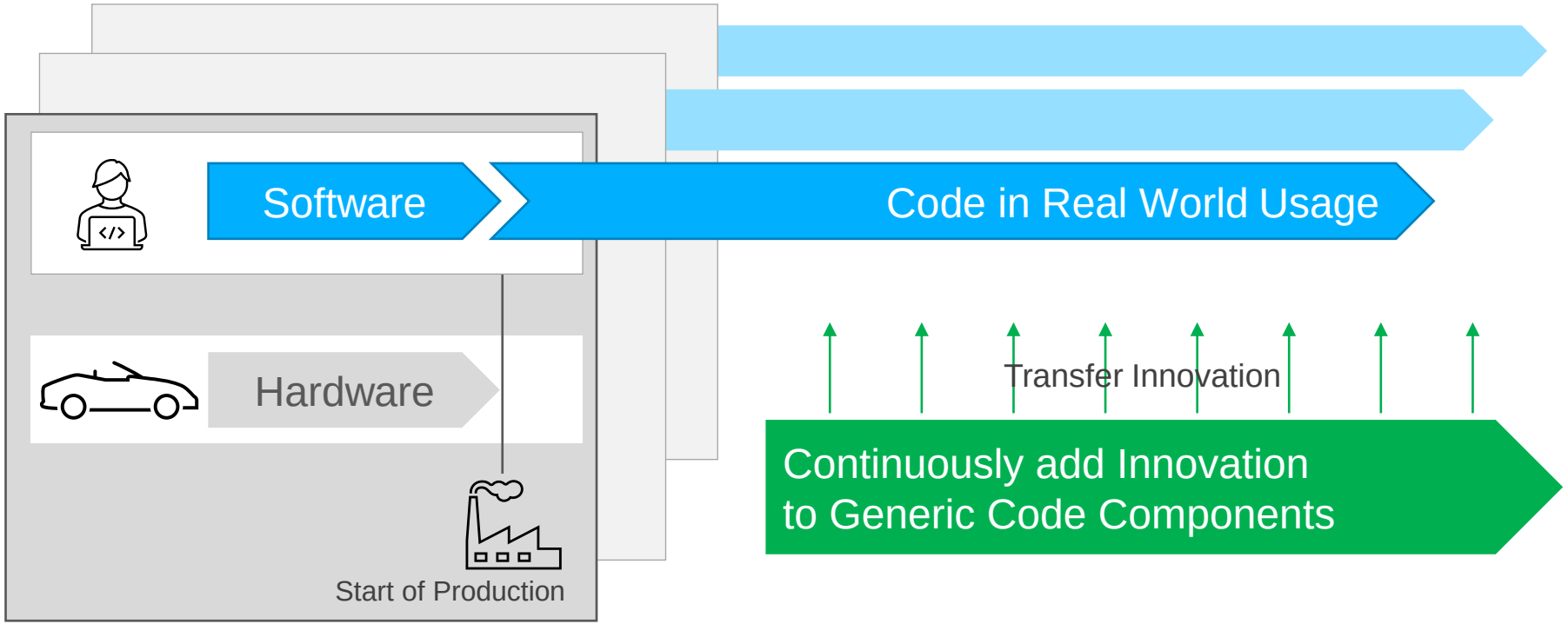
Measure deviation of project variant from generic code

Reveal affected code areas

Generic Code Components

Non-Deviating Code as Key for Delivering Innovation after SoP

Applying analytics to minimize friction in innovation transfer



Learning Curves for Software Organizations

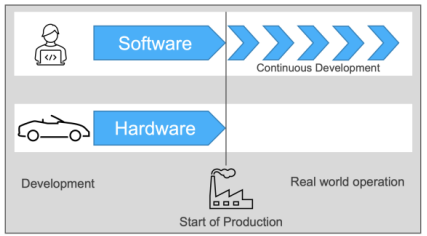
Navigating to highest software excellence with analytics

Analytics-Driven
Improvement Process

Generally:
→ Measure and increase efficiency of software development organizations

Software “outside” the Car
→ Avoid the Biz|DevOps gap in scaled agile organizations

Embedded Software
→ Increase code reuse systematically



Embedded Software
for software-defined vehicles
→ Avoid code deviation in deployed software

Thank You!

Dr. Johannes Bohnet
johannes.bohnet@seerene.com