Hot-or-Not | March 18

Sioux Technologies & Volvo Cars present:

A tour through the testing landscape of the future

Jonn Lantz

- **Technical Leader in Agile Software Development** at Volvo Cars

- Quantum Physicist
- Teacher
- Change driver
- Technical Specialist
  - Electric Propulsion Systems
  - Continuous Integration
  - Research collaboration

Bryan Bakker

SIOUX TECHNOLOGIES

- **Test Architect** at Sioux Technologies
- Frequent speaker on test matters
- Co-author *Finally… Reliable Software!*

Agenda

18:00 – 18:05	Welcome
18:05 – 18:30	Jonn Lantz, Volvo Cars: **From fossil dinosaurs to electric transport services**
18:30 – 19:00	Bryan Bakker, Sioux Technologies: **Safely crash in virtual space: battling bugs with test models**
19:00 – 19:15	Q&A
19:30	Wrap-up

How to ask your questions

- "@Jonn: <question>"
- "@Bryan: <question>"
- "@Crew: <remark>"
  (e.g. *no sound*)

From fossil dinosaurs to electric transport services

Volvo cars

From fossil dinosaurs to electric digital business

Hot||Not 2021 | Jonn Lantz

About me

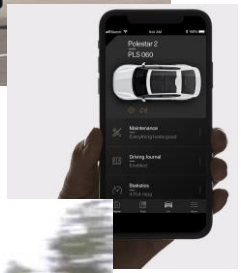
Background: researcher (physics), teacher, now agile technical leader at Volvo Cars

Experience from electrification, software and organizational change + sw engineering research

Deeply involved in the new c++ based posix platform "SPA2"

Academic network: software-center.se
+ industrial PhD student

Part 1: from adiabatic to disruptive



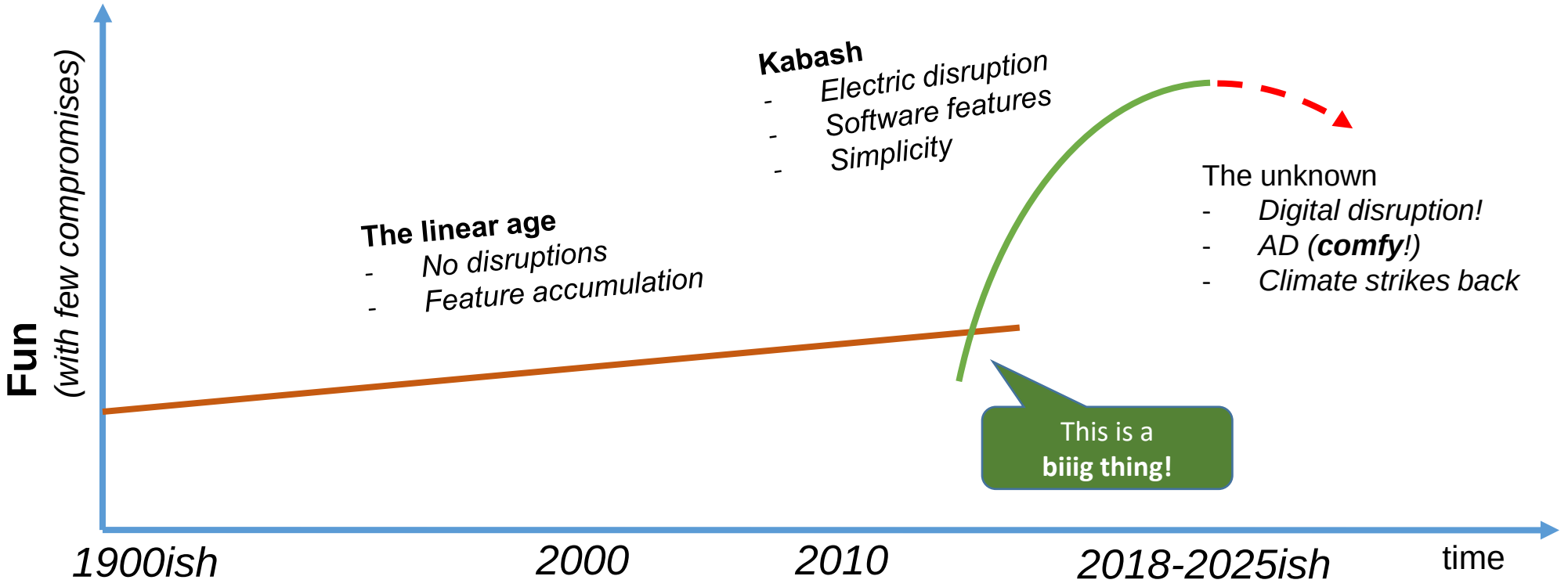
*It's electric! (**Boom**)*

It's connected to your smartphone (and the cloud)

It's fun to drive (with increasing automation)

It's Design and software! (data is coming)

Peak fun to drive is now!

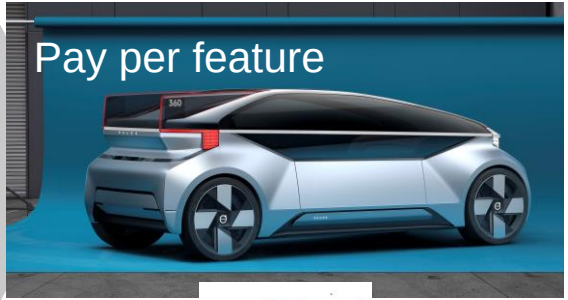


The electric disruption

"Normal+" pricing
Long range
Low and high performance
OTA and not yet OTA

Car **business** 2020+

This old fashion side generates **sold cars** (and leasing)



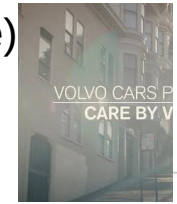
Safe, efficient transport

This new side will generate **continuous revenue** (if done well)

Car as a service?

Owning?

- Diesel/Hybrid cars are attractive to **lease** (old style car as service)
  * Quick depreciation
  * Lots of service

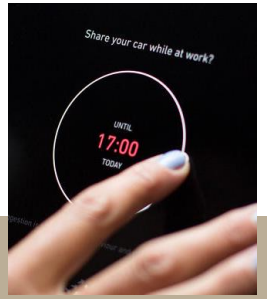


- Electric cars are different and can be attractive to **own** ...
  * Long expected lifetime, and value
  * Low owning cost
  * Less service

Zūm

Sharing?

- The value of an **owner cleaning** and **caring** for the car (for free) affects the potential business models!



Or

Features on demand or data might be easier?

Part 2: Development

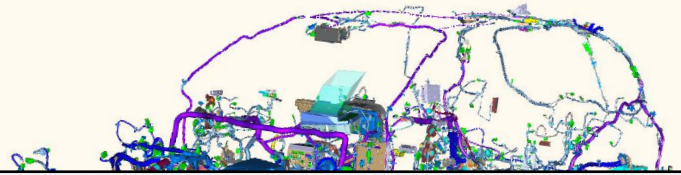
Consequences for development and testing | *continuous*

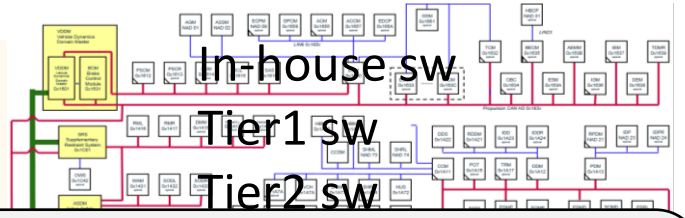
Tech

We have 2 main differentiating domains

- *Design (OK)*

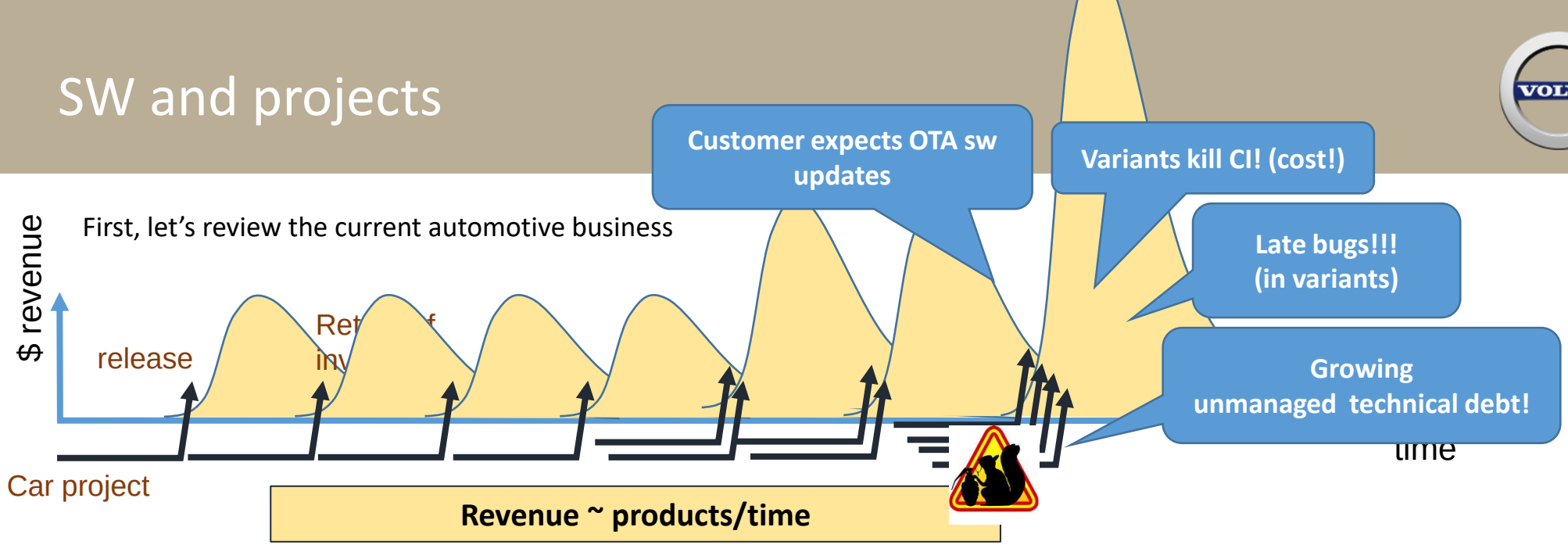
- *Software (NOK)*





Technical debt in a complex product

SW and projects



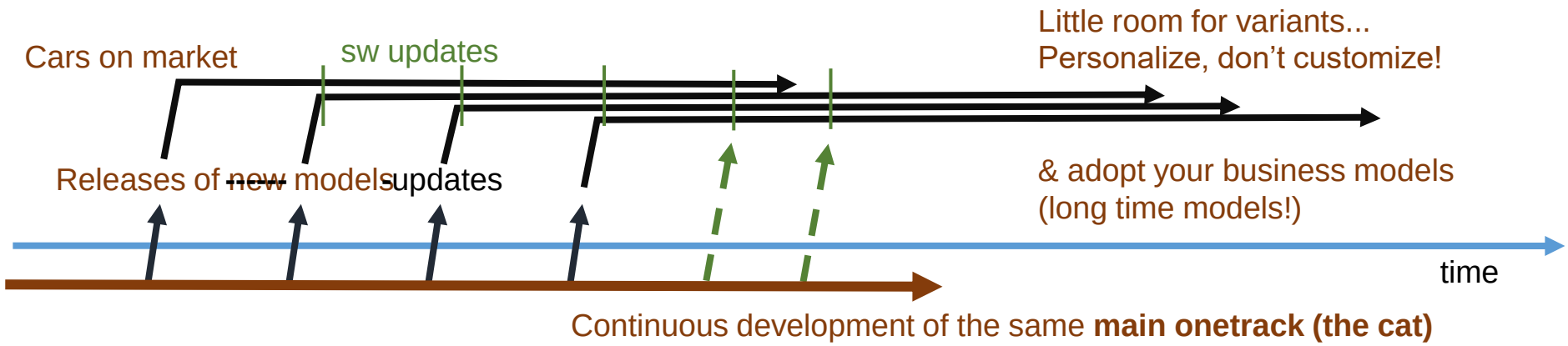
To solve this we have to look over both business model, product, organization...

Revenue ~ happy customers?

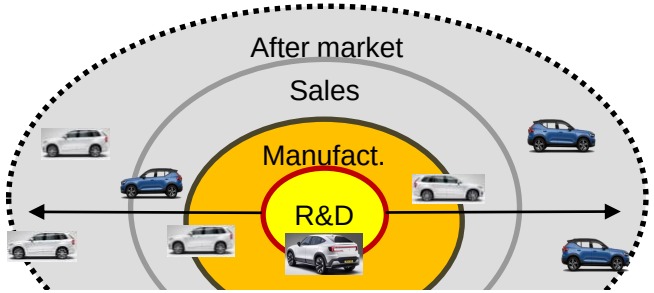
The car is a cat!

Towards single track

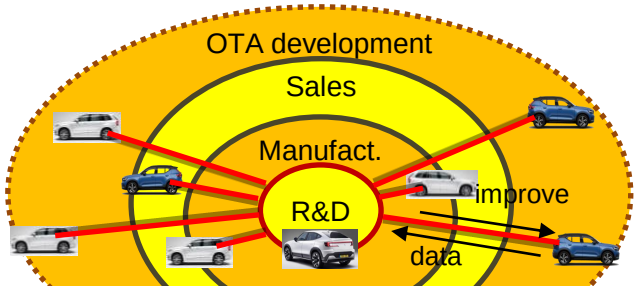
Automotive will have to adopt to the rules of other sw business:
*a better car experience over time... (requires **OneTrack**)*



Macro: A connected R&D

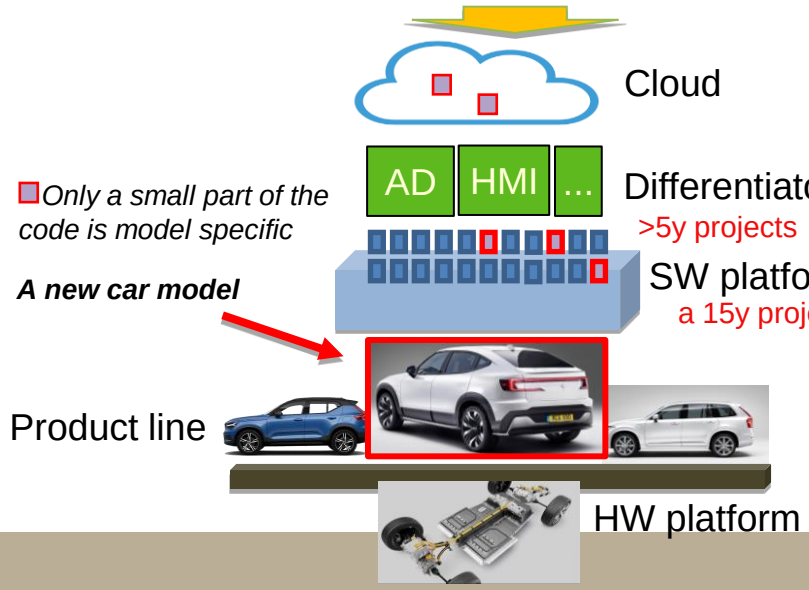


HW company
Develop and ship new models!
Customization!



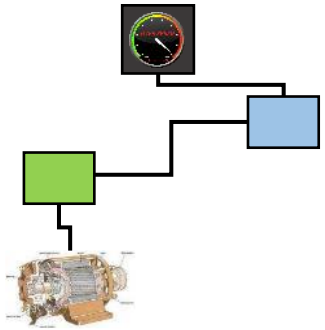
SW company
Continuous data driven development

A new sw stack
A product line is continuously developed as a OneTrack using CI.
Few variants and digital personalization



Micro: System design is dead

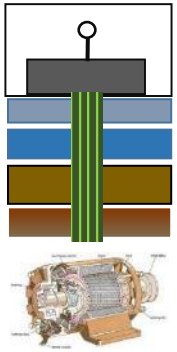
Long live evolution and backward compatibility!



Every system is unique!

Spaghetti architecture
Projects
Safety by separation / wires

A tough period



Service design in a common platform with backward compatibility

Scalable architecture
Product
Safety by layers and redundance*

** not discussed this time :(*

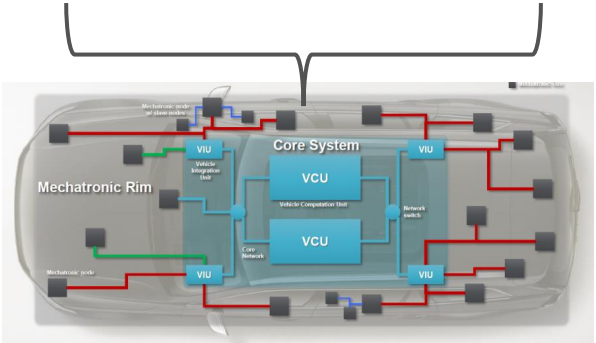
Cheers to Helena Holmström Olsson & Jan Bosch

User centric

The A/B test

Everyday my car experience is better...

- **Empowered agile teams**
- **One sw track**



OTA sw Updates

Create customer value

Usage data collection (Data pipelines)

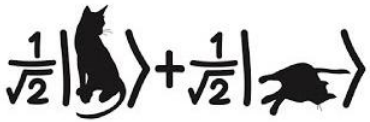
Selected Customer

A: standard sw
B: changed sw

All company cars today are fully connected!

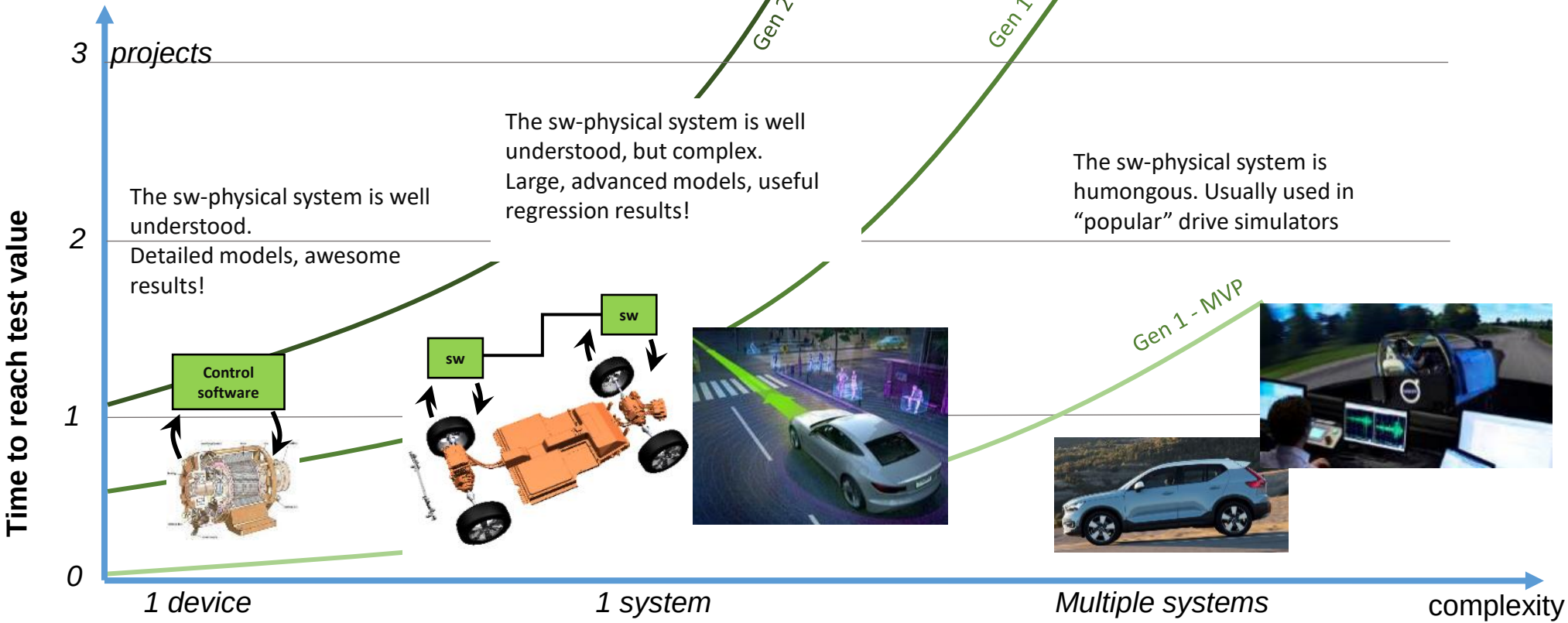
Note: A Data Pipeline is not a NAS with all data!
It's *infrastructure*

Part 3: methods and models



Some consequences for test and development

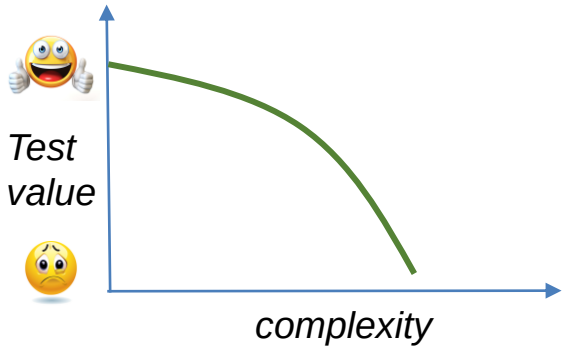
The challenge of scaling Virtual testing



Model based testing

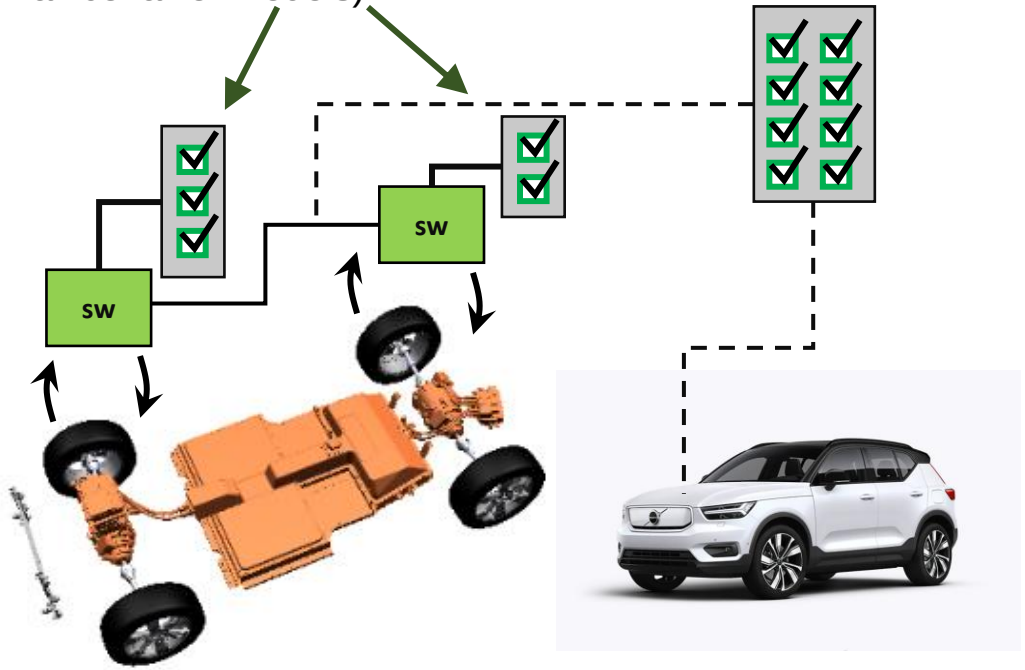
- Rule based testing is excellent using "plant models"
- Most "simple" control systems can be tested & regression tested using MBT
  (… if the plant model can be updated from real data)

- How about MBT of Virtual machines? *(integrate and run on a virtual ECU + plant model on your PC...)*
- Will grow in AI/ML!? *(while not too complex contexts)*



Rule based testing

Test using “programmed requirements”, “oracles” checking the code runtime, is a success!
(small behavior models)



Can be scaled,
- to integration,
- to rigs
- to vehicles on roads...

Is a *small data* solution

The Test Strategy

About forms of development, the testing, *and the role of code certification...*

BAPO – make sure you enable the future (with good Architecture)

Test: Optimize for growth, architecture, aim for fast complete level feedback early

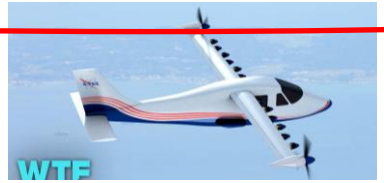
Δ(BA)PO – business by differentiation

Test: Optimize for feature speed, develop a strong defensive CI

Δ(BA)PO – drive business agility

Test: Optimize for customer KPIs (OTA testing is core business!)

High quality codebase -> innovate!

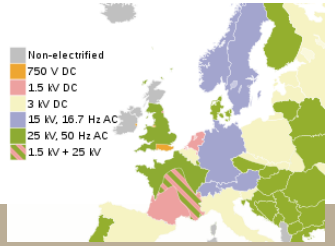


Certify this!

New code

Low quality legacy code (spaghetti & branches) -> do something...

Old/mature code



OTA code

Time to think about AI...

... but how about this certification?

BAPO = Business, Architecture, Process & Organization – the order to do things

Thanks!

Data driven development

Consequences of the change

Customer value is ***my*** concern!

Business is ***my*** concern!

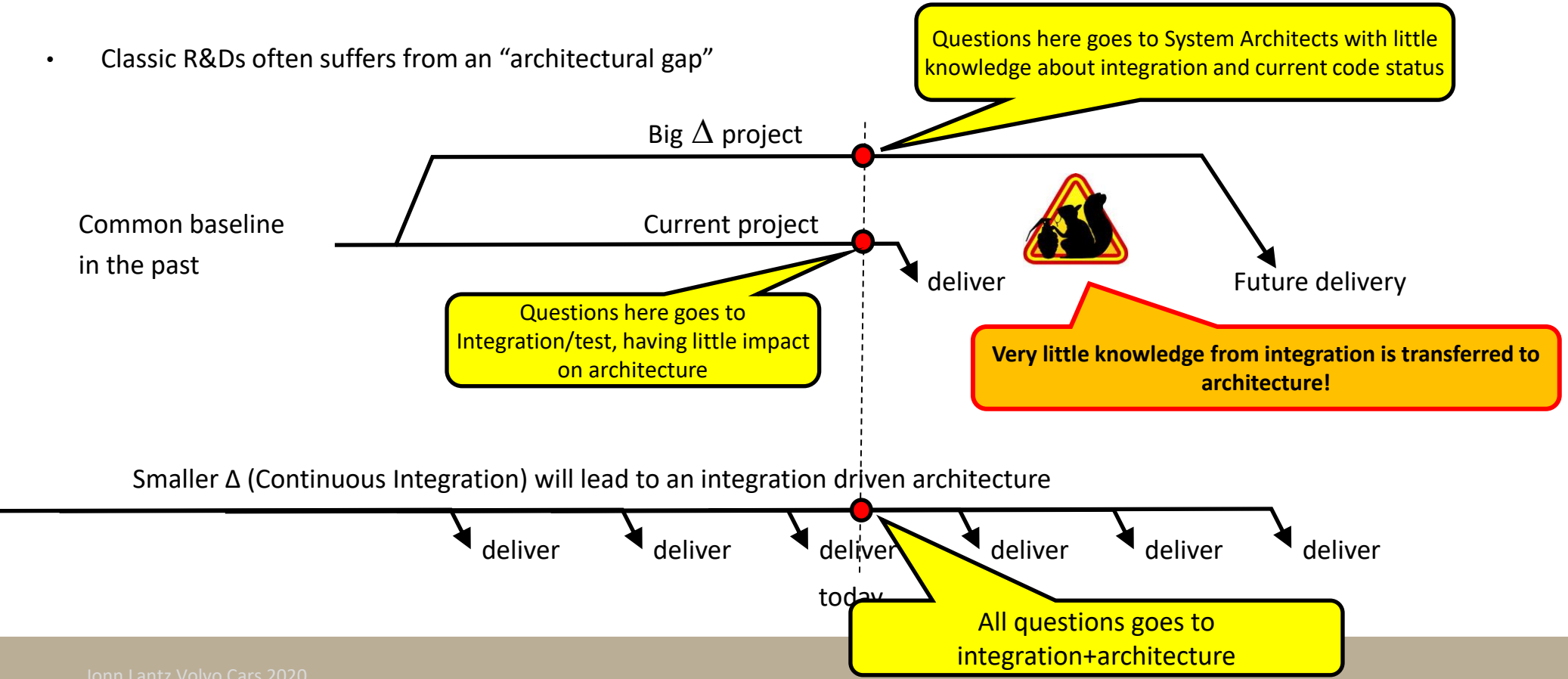
How shall we organize?

Test is ***my*** concern!

Complete level integration is ***my*** concern!

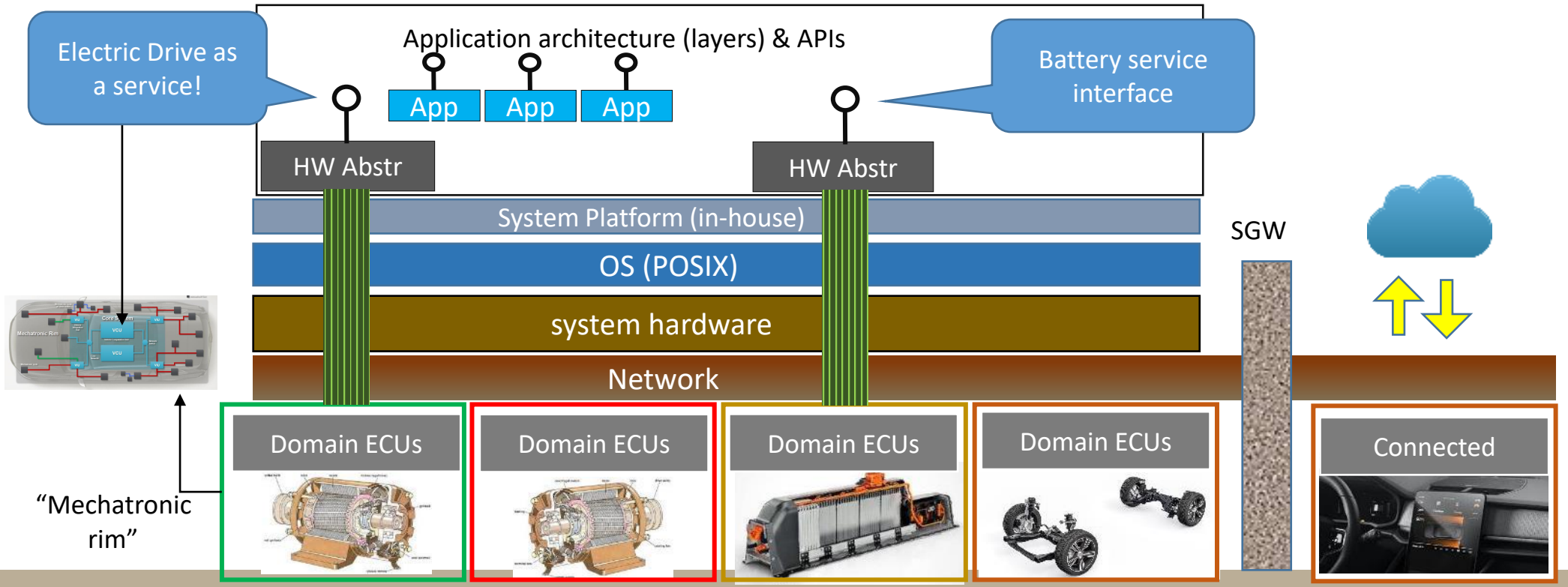
Integration (test) driven architecture

- Classic R&Ds often suffers from an “architectural gap”

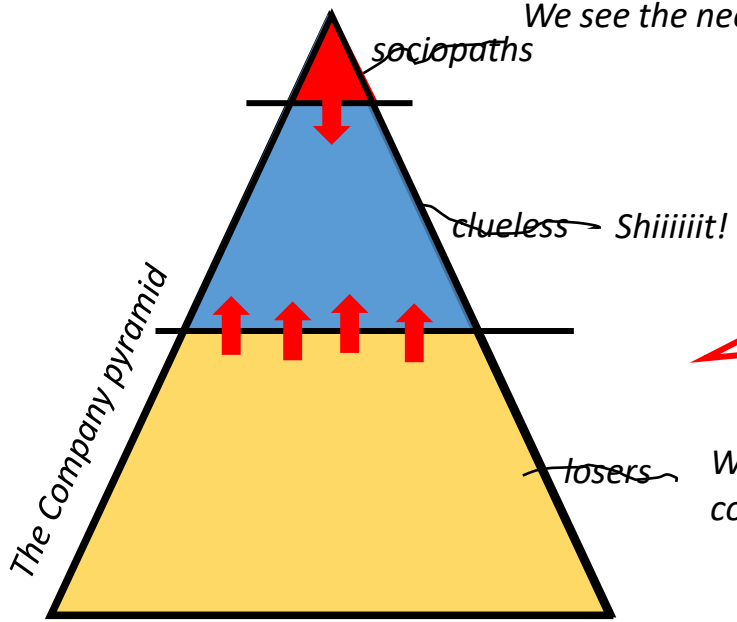


The mechatronic service architecture

We, the OEM, needs control over the main software.



Changing the organization – is not trivial

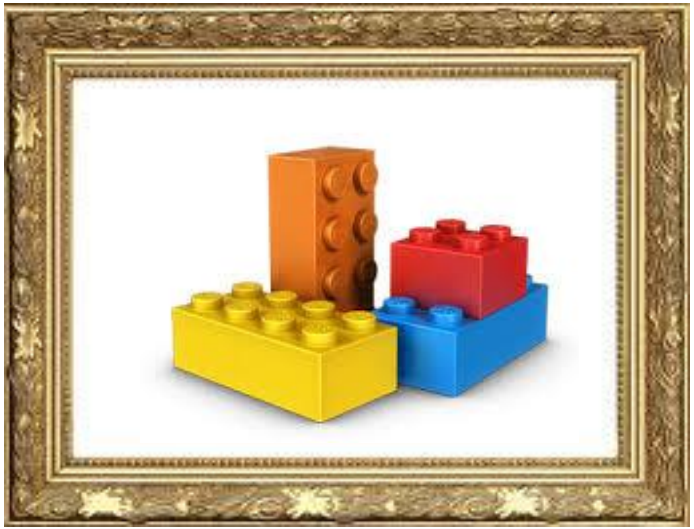


We see the need for change! (Go agile [SAFe], Go empowerment[ish]!)

The hard problem:
Culture eats ~~strategy~~ for breakfast!
~~change~~
everything

We are a (large) mix of both engaged and frustrated or conservative communities.

No. More. Lego!



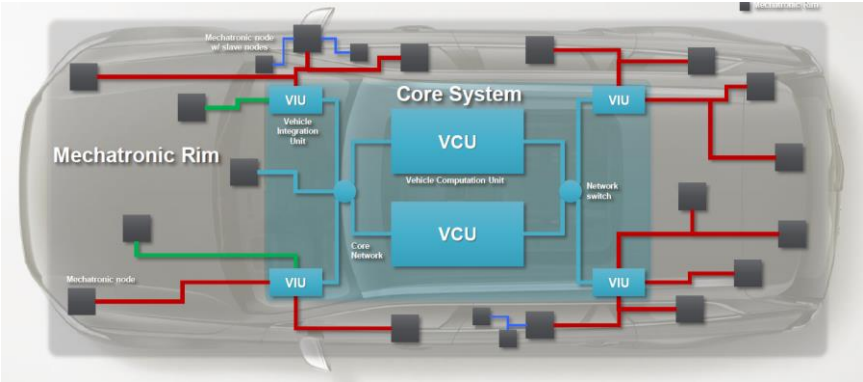
Flexible modules
Modelling
Hardware focus
Test per module

Will change
Processes
Methods
Tools
Culture

Single track strategy
Backward compatibility
Technical debt management
Continuous Integration

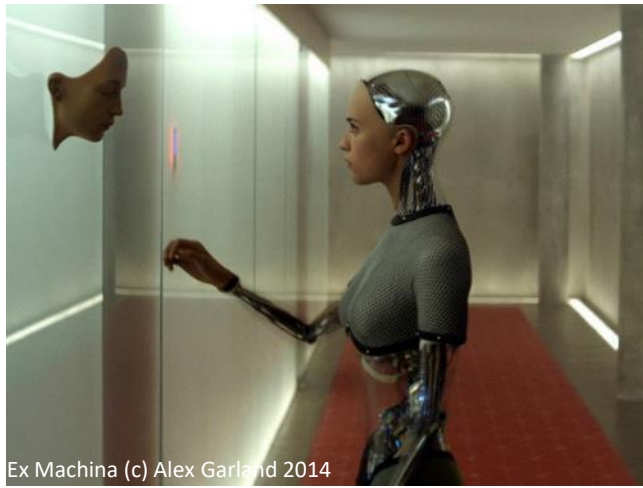
Towards a posix platform

Centralize software development
S*eparate hardware from software*!
Aim for a **Service strategy**



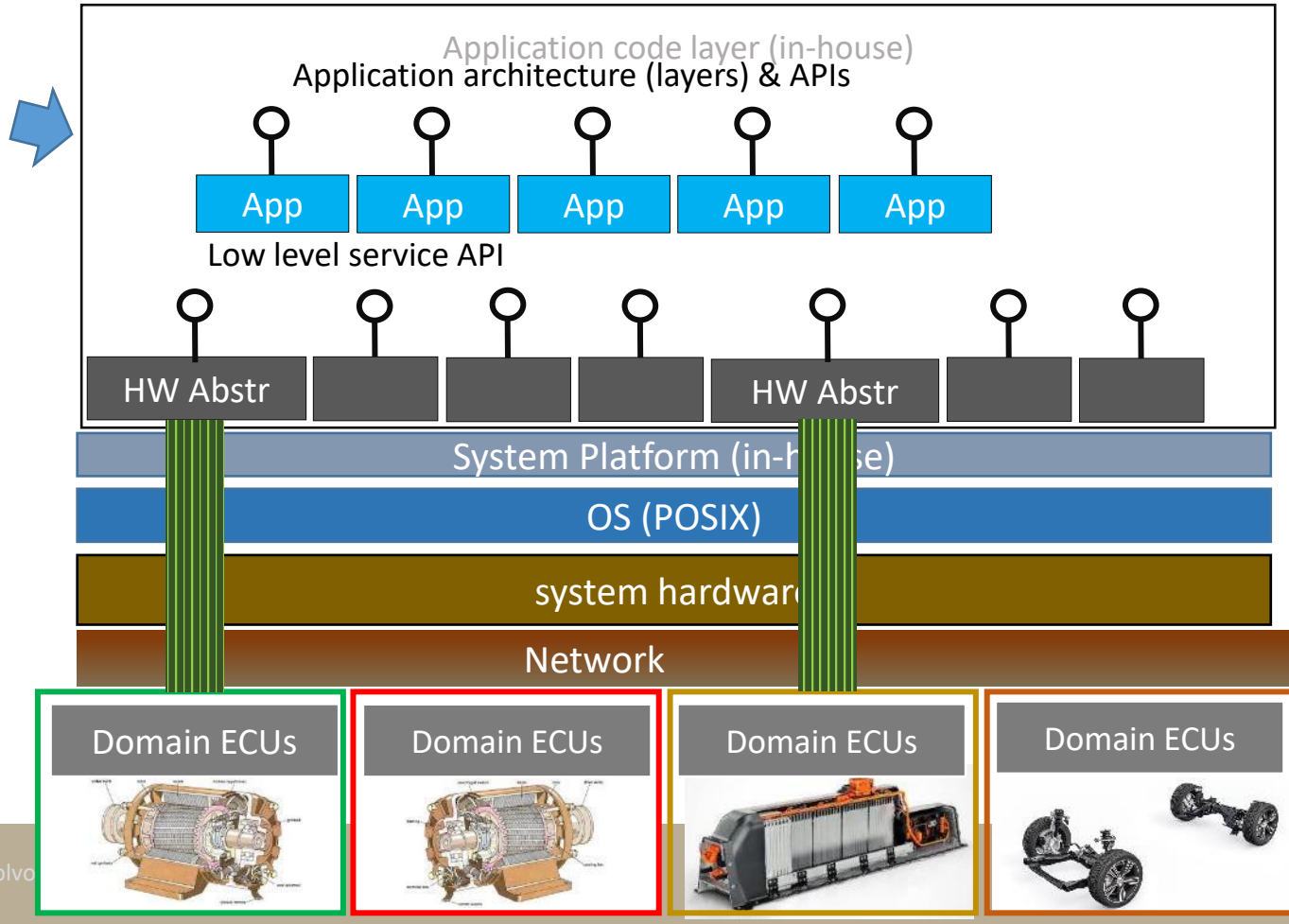
Create a platform
Linux / POSIX OS
Ethernet (backbone)
Service architecture / devices as services

The age of robotics


Ex Machina (c) Alex Garland 2014

The mechatronic service architecture

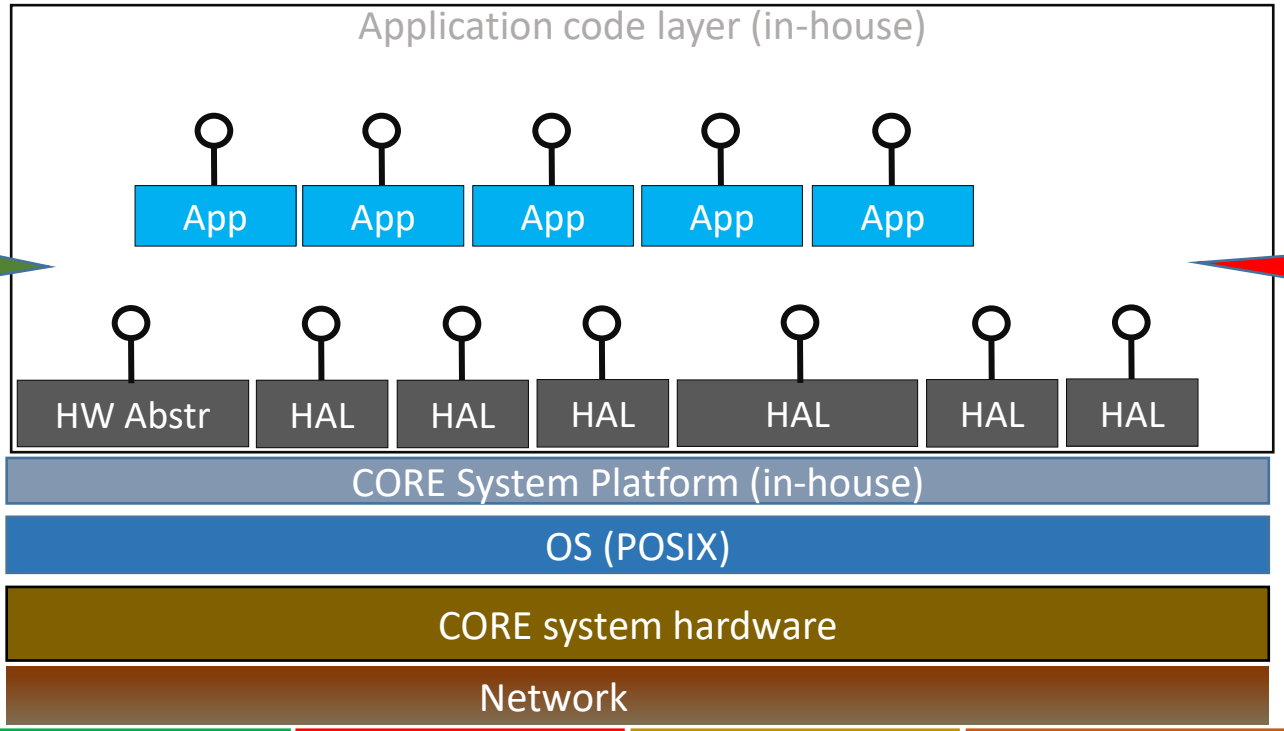
*This is **not** A smart phone...*



Nirvana for Model Driven Engineering???

Yes

- *Full control*
- *Service architecture (modules)*



Weeeell...

- *C++, OO, more code*

As before:

- *Complex robotic system*
- *Complex physical constraints*
- *Multiple domains*
- *Variants?*

Tool business with In-sourcing

The old way

Tool Supplier

Awesome tool!

OEM engineer

Value!!!

The new way

OEM engineer

More Value!!!

Internal eco systems

ecialized tools

feedback & internal

continuous deployment!

Tool Supplier

Generic tools

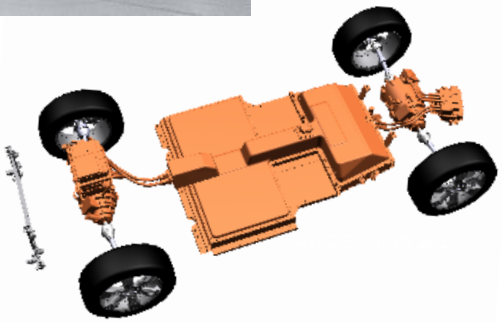
Open source etc.

OEM

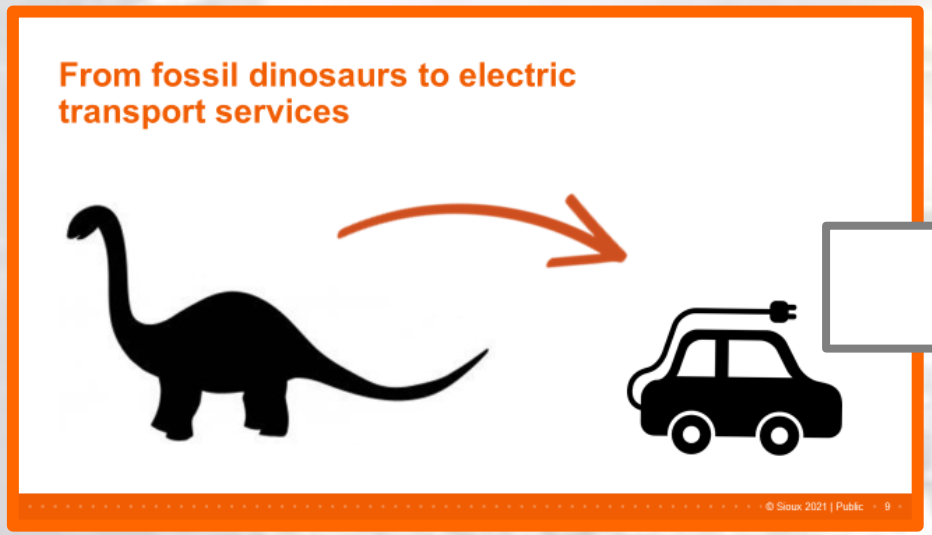
In-house service developers (tool eco systems)

Must integrate with other (usually open source tools)

Root cause



Ask your questions



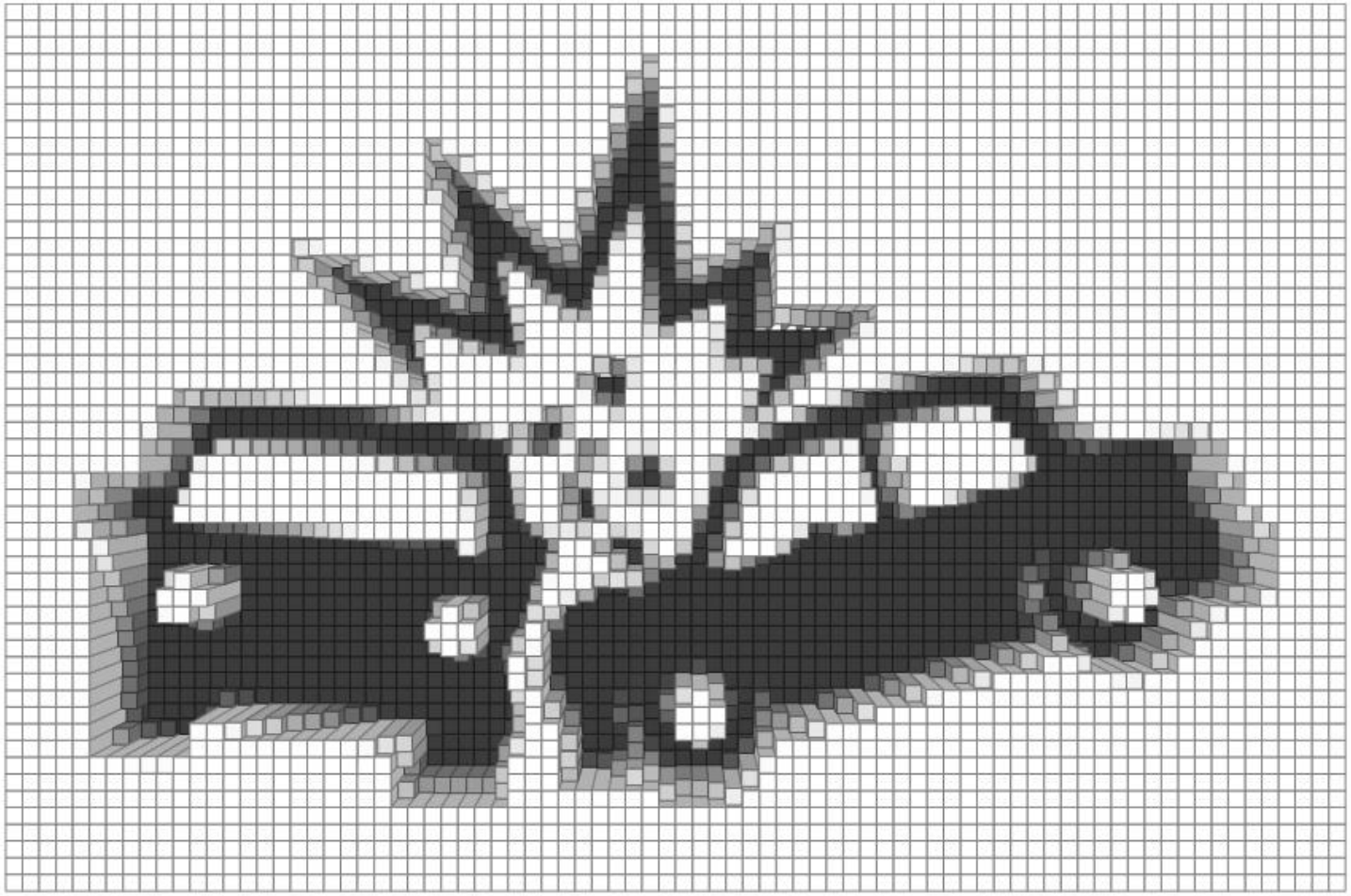
“@Jonn: <question>”

“@Crew: <remark>” (e.g. *no sound)*

Safely crash in virtual space

Battling bugs with test models

SIOUX TECHNOLOGIES



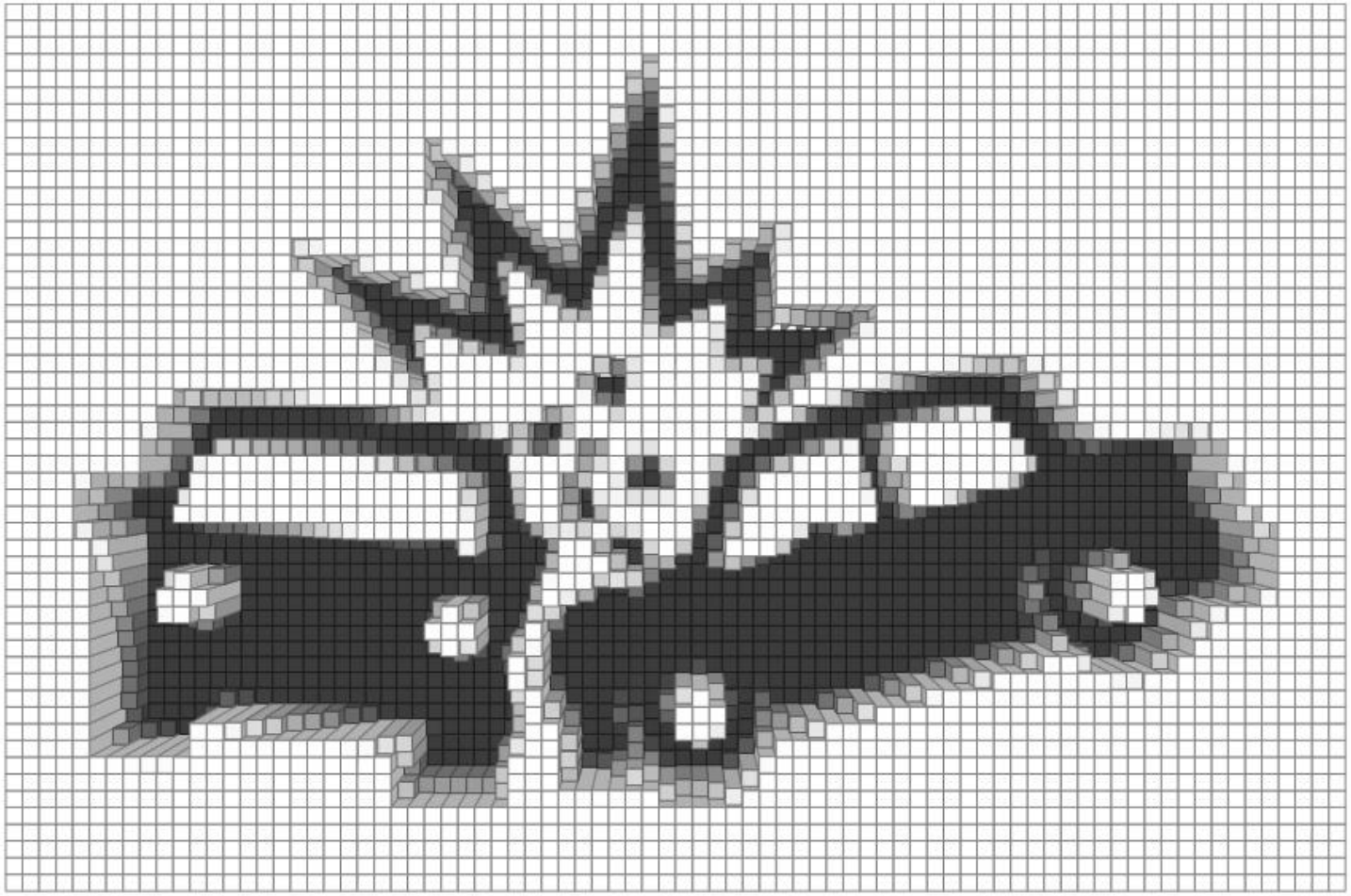
Bryan Bakker

Dirk Coppelmans

Safely crash in virtual space

Battling bugs with test models

SIOUX TECHNOLOGIES



Bryan Bakker

Dirk Coppelmans

Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. Taking up the challenge
4. Future?

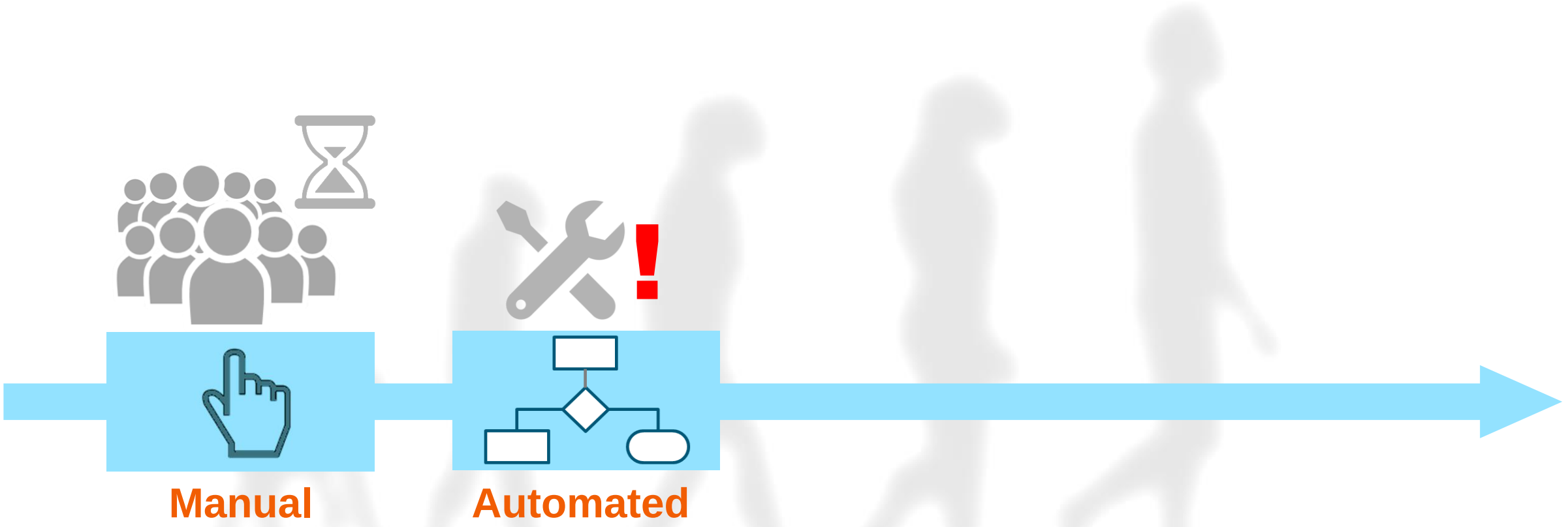
Safely crash in virtual space

1. **Evolution of software testing**
2. The next best thing
3. Taking up the challenge
4. Future?

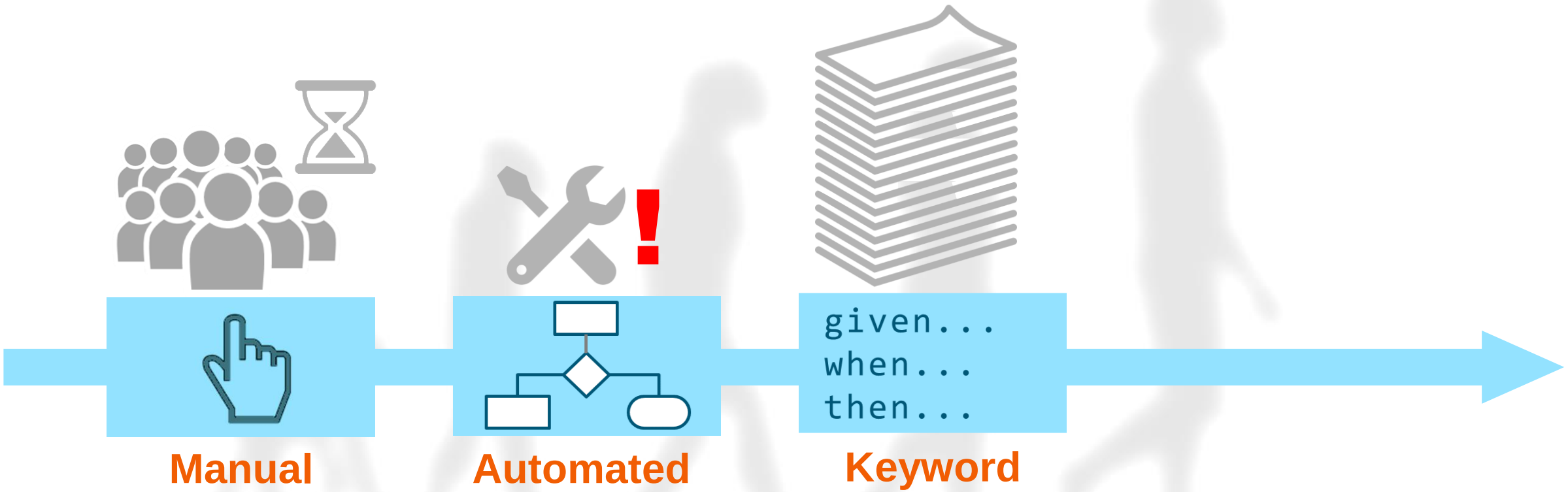
Evolution of software testing

Evolution of software testing

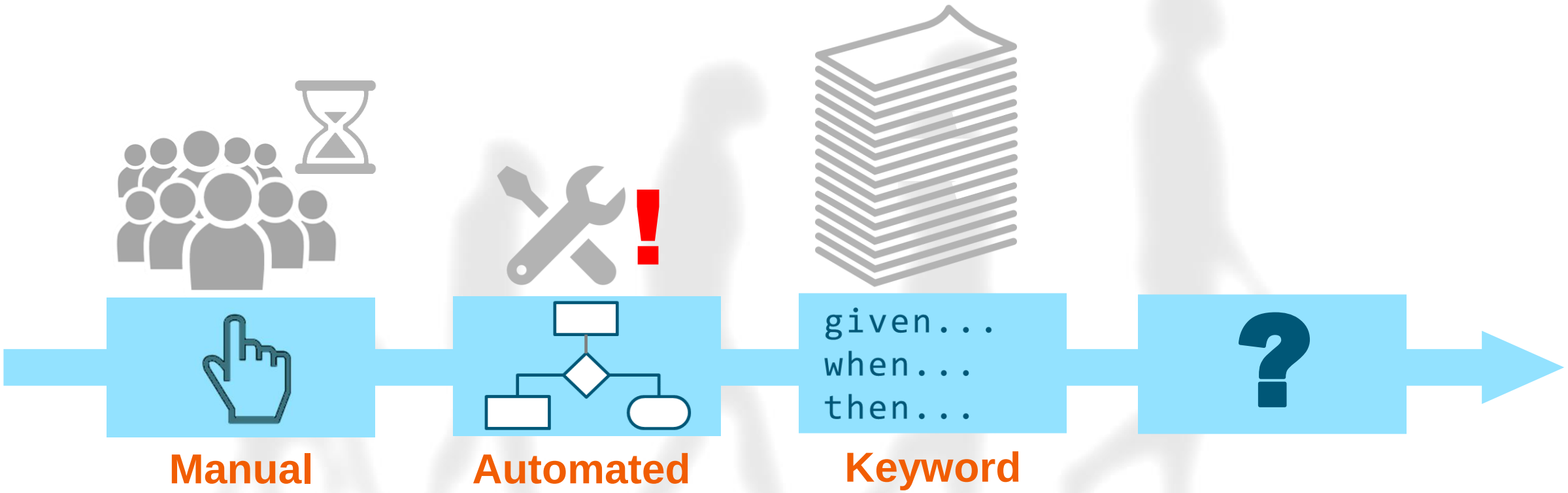
Evolution of software testing



Evolution of software testing



Evolution of software testing



Evolution of software testing – Compare to SW development

- Increase in use of formal models
- Originates from research & universities
- No longer limited to safety and reliability critical environments, like automotive and aviation
- Applied to manage complexity

Type	Tools
System design	• ASD / Dezyne (Verum) • mCRL2 (TUe, verification engine for ASD/Dezyne)
Interface	• Pact • ComMA

verum.

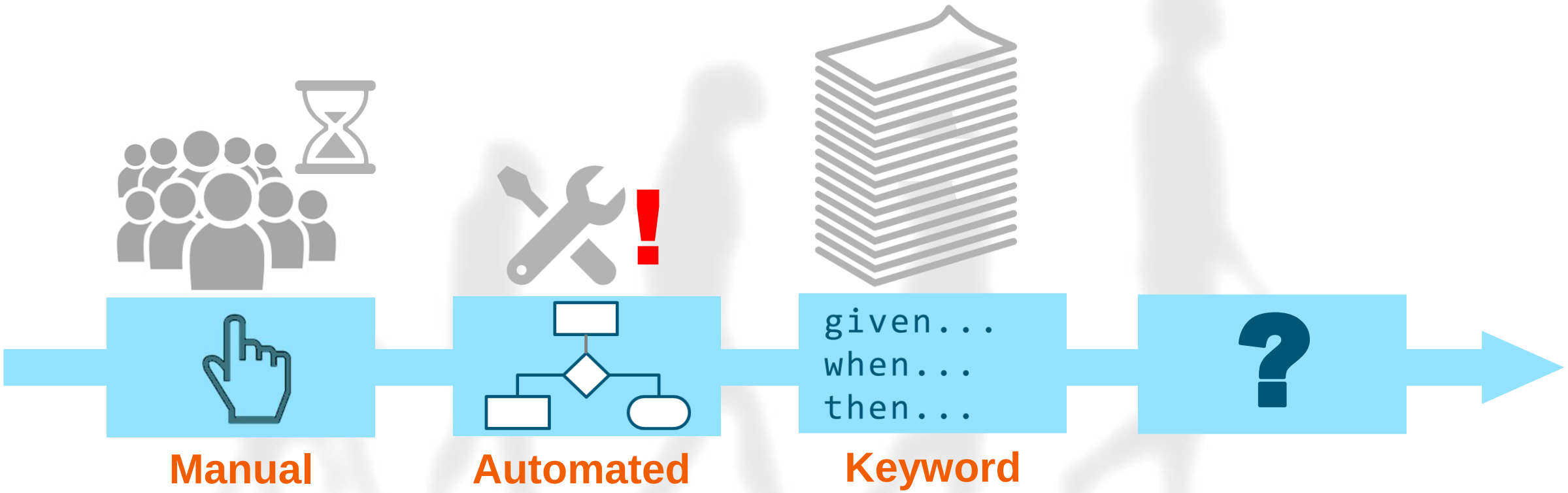
PACTS

Evolution of software testing – Compare to SW development

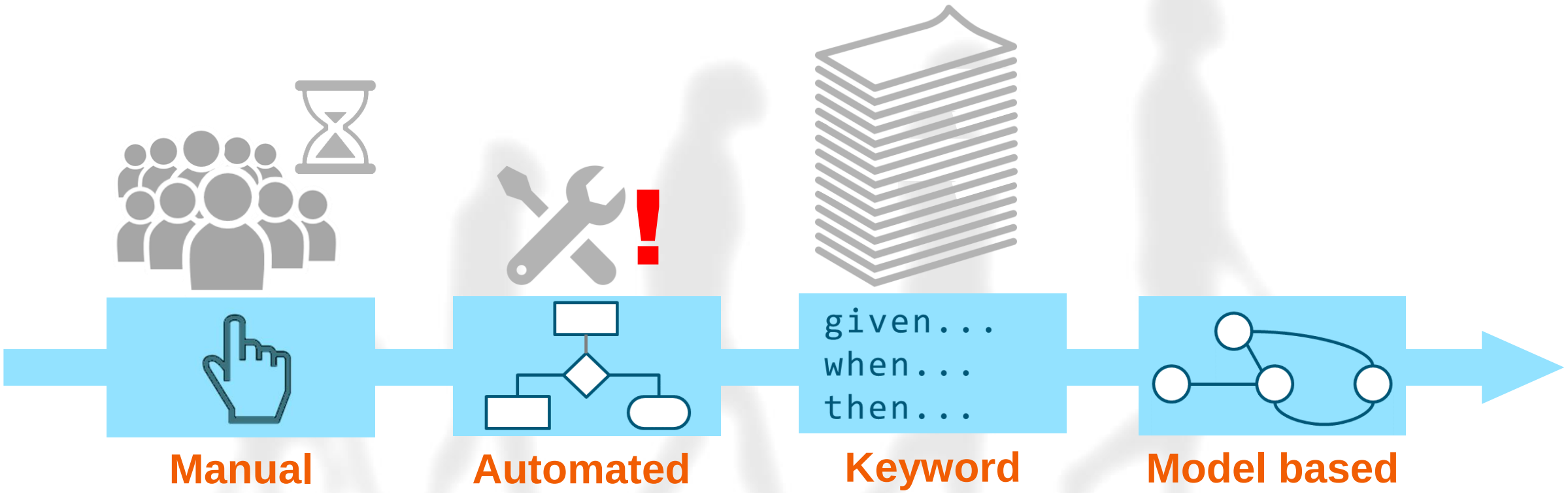
- Development models are design models and by far flawless
- They are verified through:
  - Model checkers
  - Model-in-the-loop testing (MIL)
  - Software-in-the-loop testing (SIL)

- By modeling behavior, new test possibilities arise

Evolution of software testing



Evolution of software testing



Safely crash in virtual space

1. Evolution of software testing
2. **The next best thing**
3. Taking up the challenge
4. Future?

The next best thing – MBT Definition

Model based testing is

automated test generation and execution

based on an abstract

behavior model

The next best thing – Developing a behavior model

1. Derive the behavior model from the requirements
2. Construct a behavior model based on collected field data (process mining)

The next best thing – Developing a behavior model

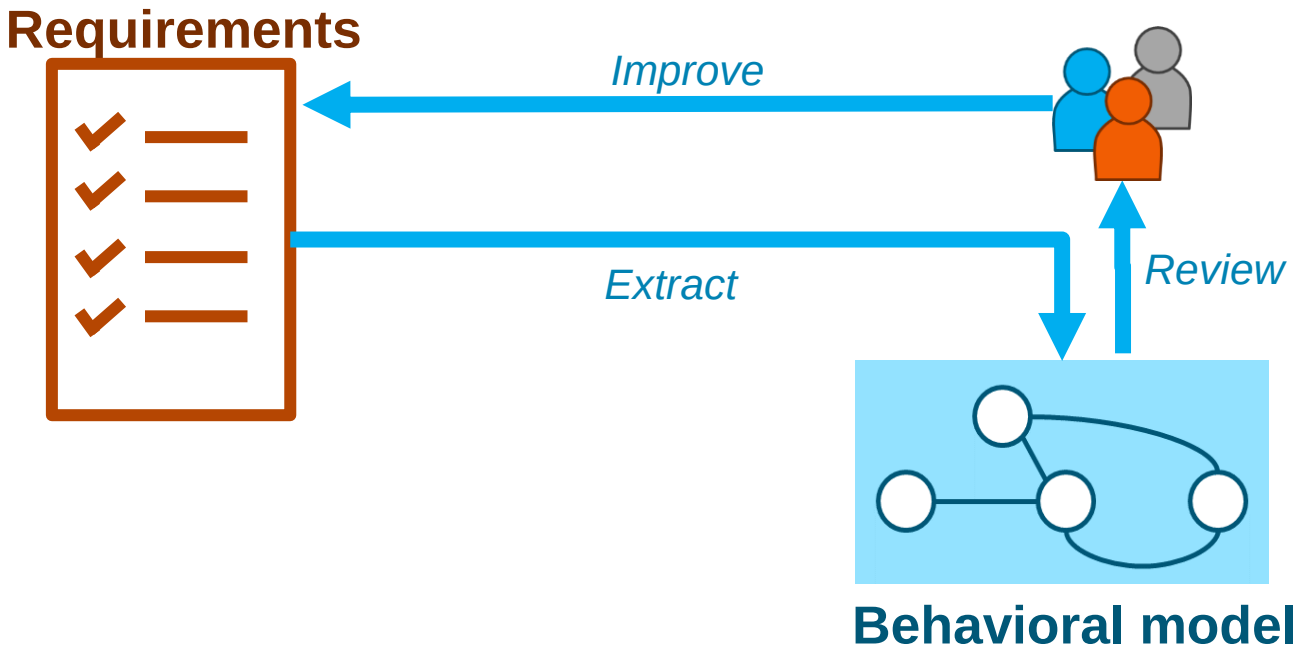
1. **Derive the behavior model from the requirements**
2. Construct a behavior model based on collected field data (process mining)

Derive from the requirements

Development

Test

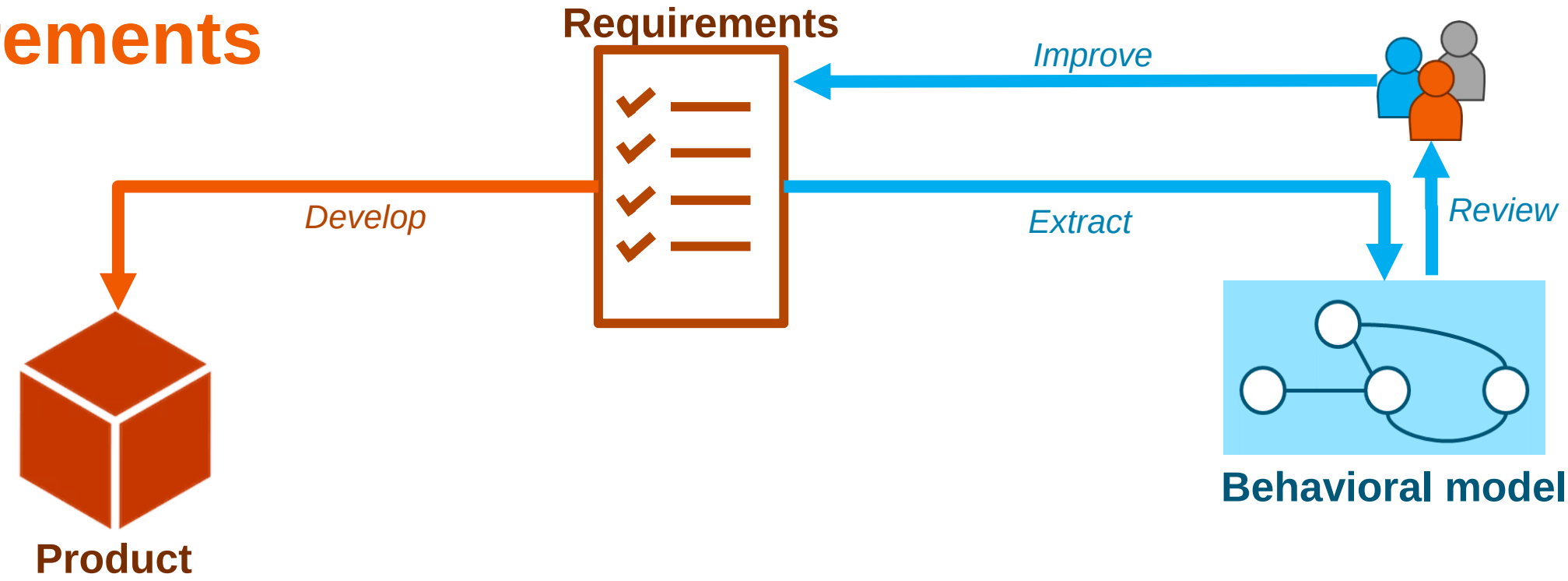
Derive from the requirements



Development

Test

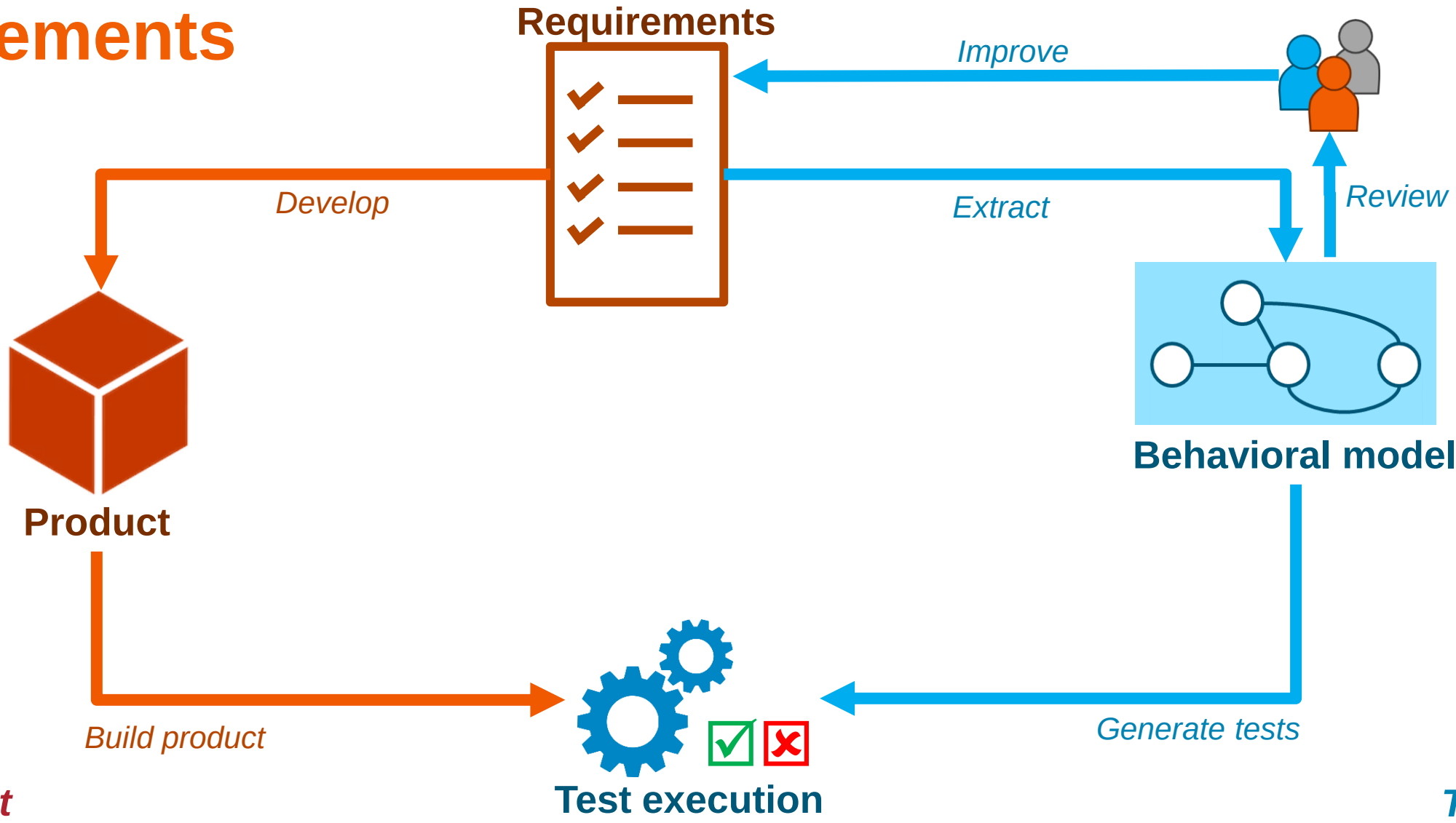
Derive from the requirements



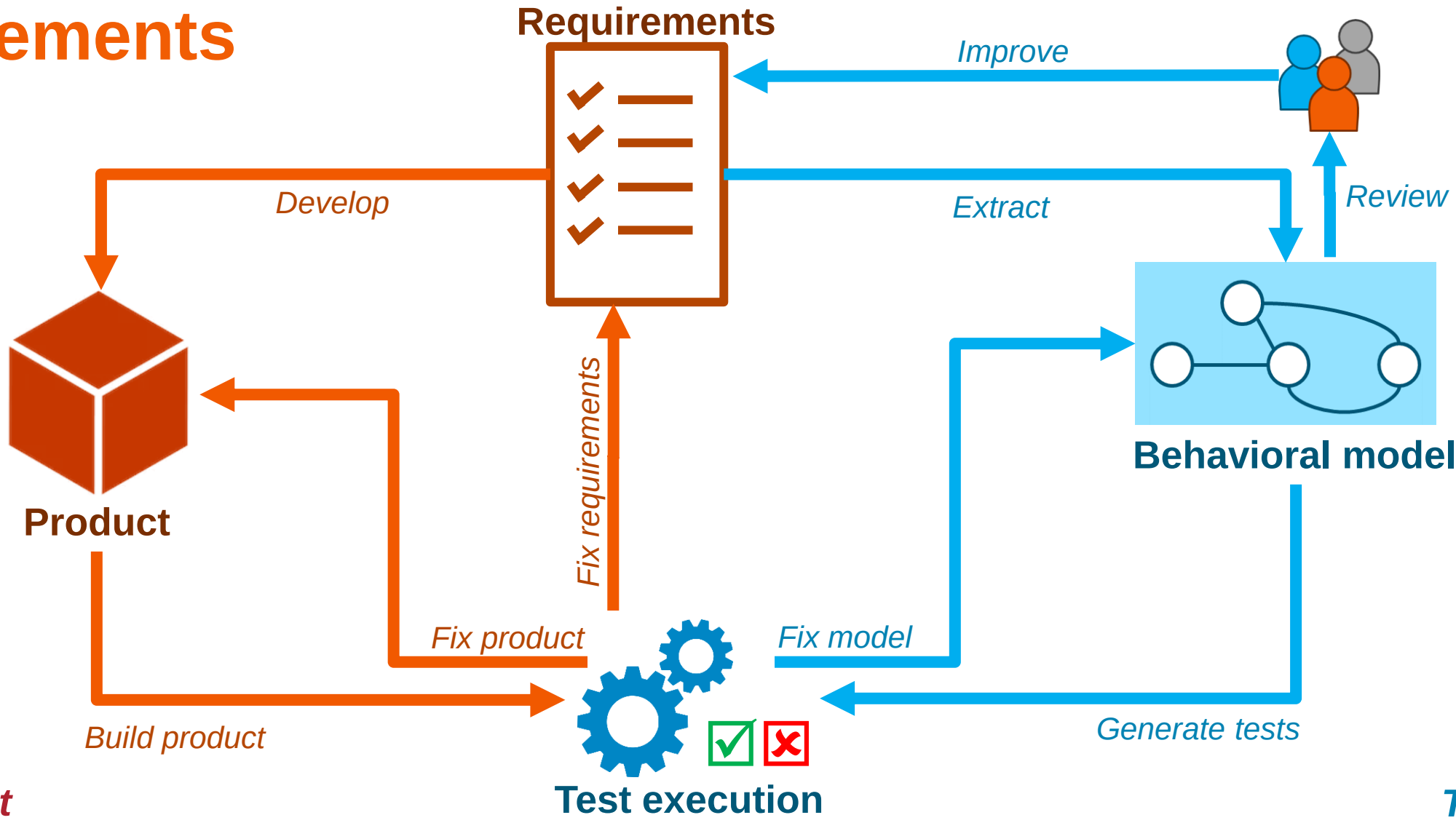
Development

Test

Derive from the requirements



Derive from the requirements



Development

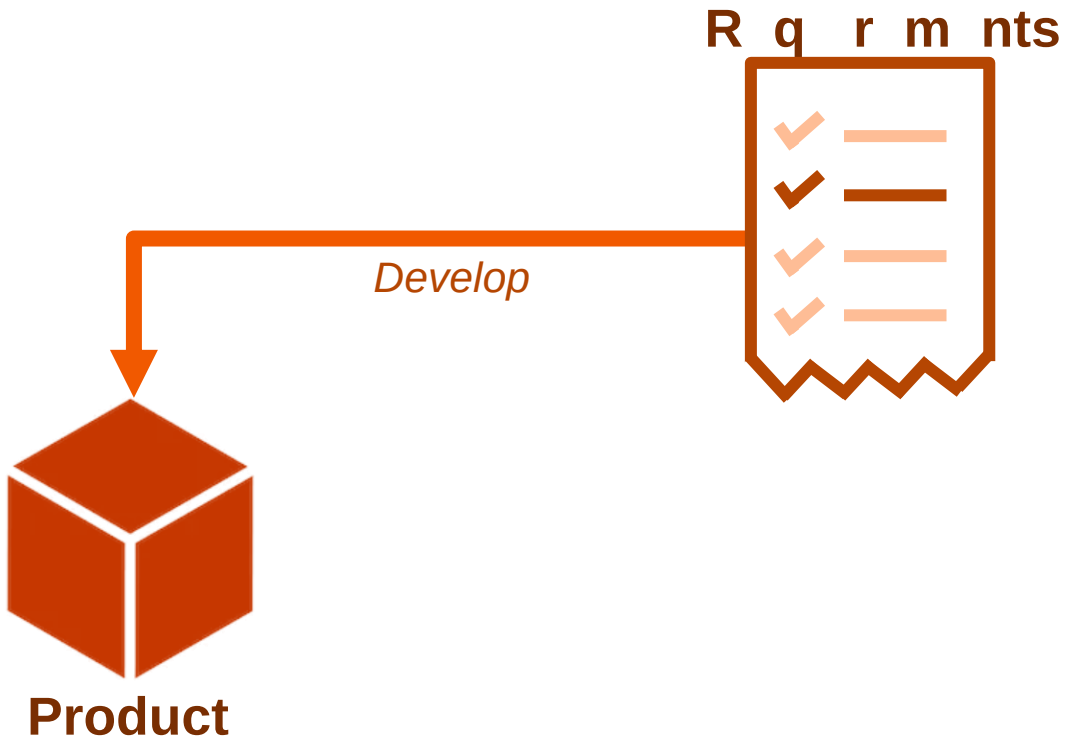
Test

The next best thing – Developing a behavior model

1. Derive the behavior model from the requirements
2. **Construct a behavior model based on collected field data (process mining)**

Process mining

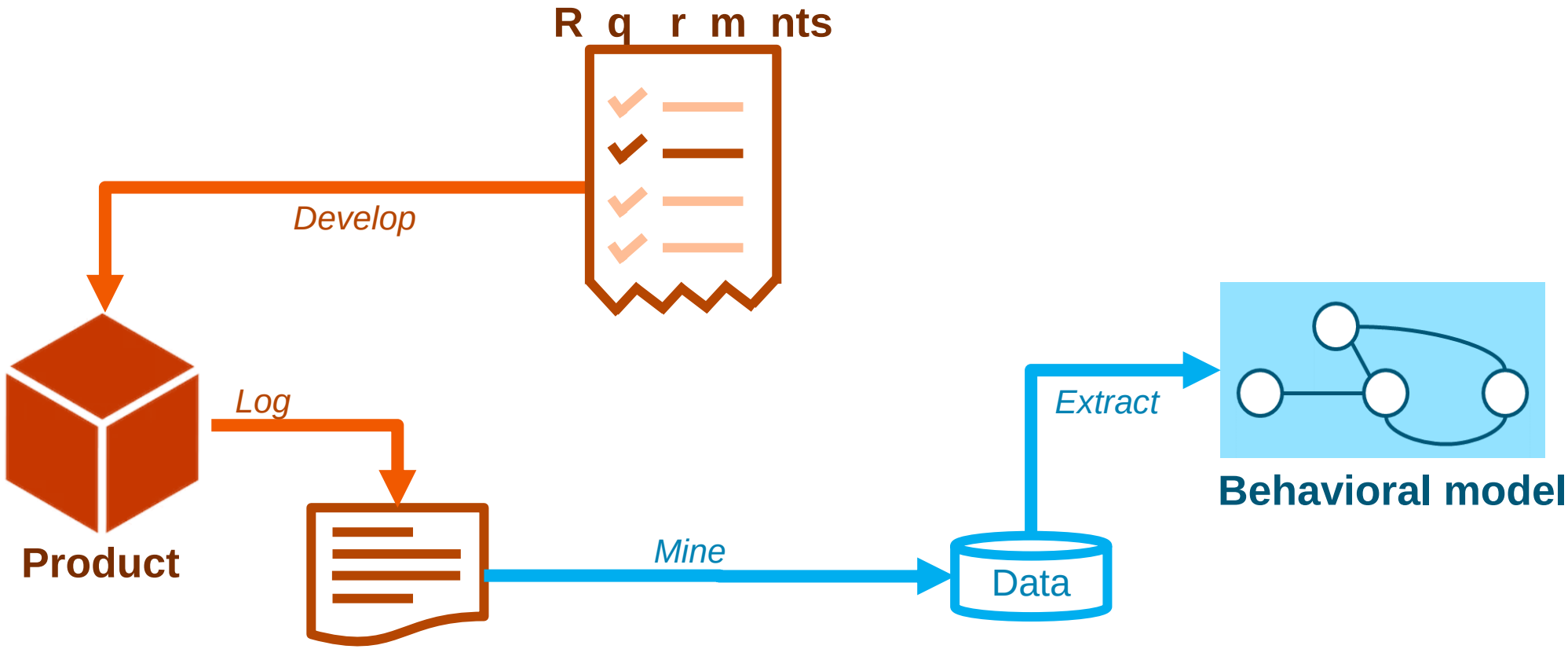
Process mining



Development

Test

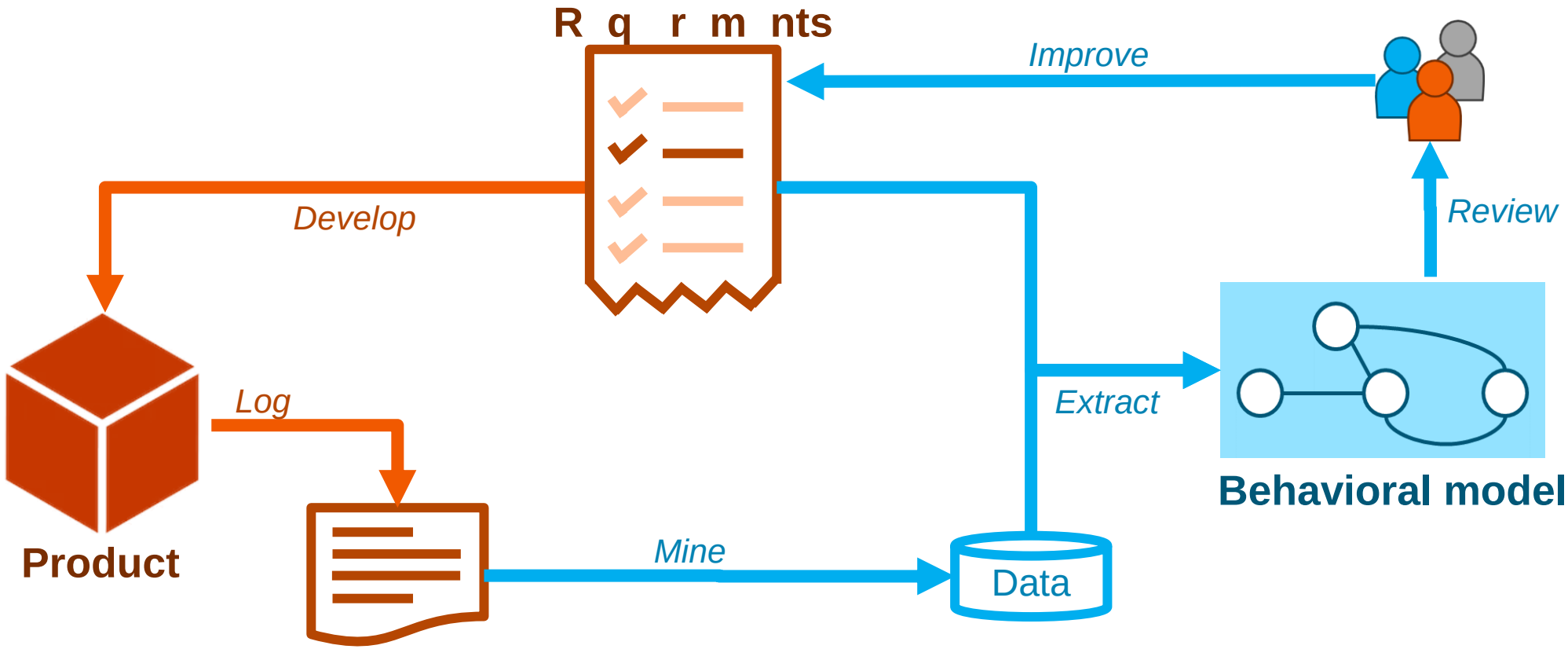
Process mining



Development

Test

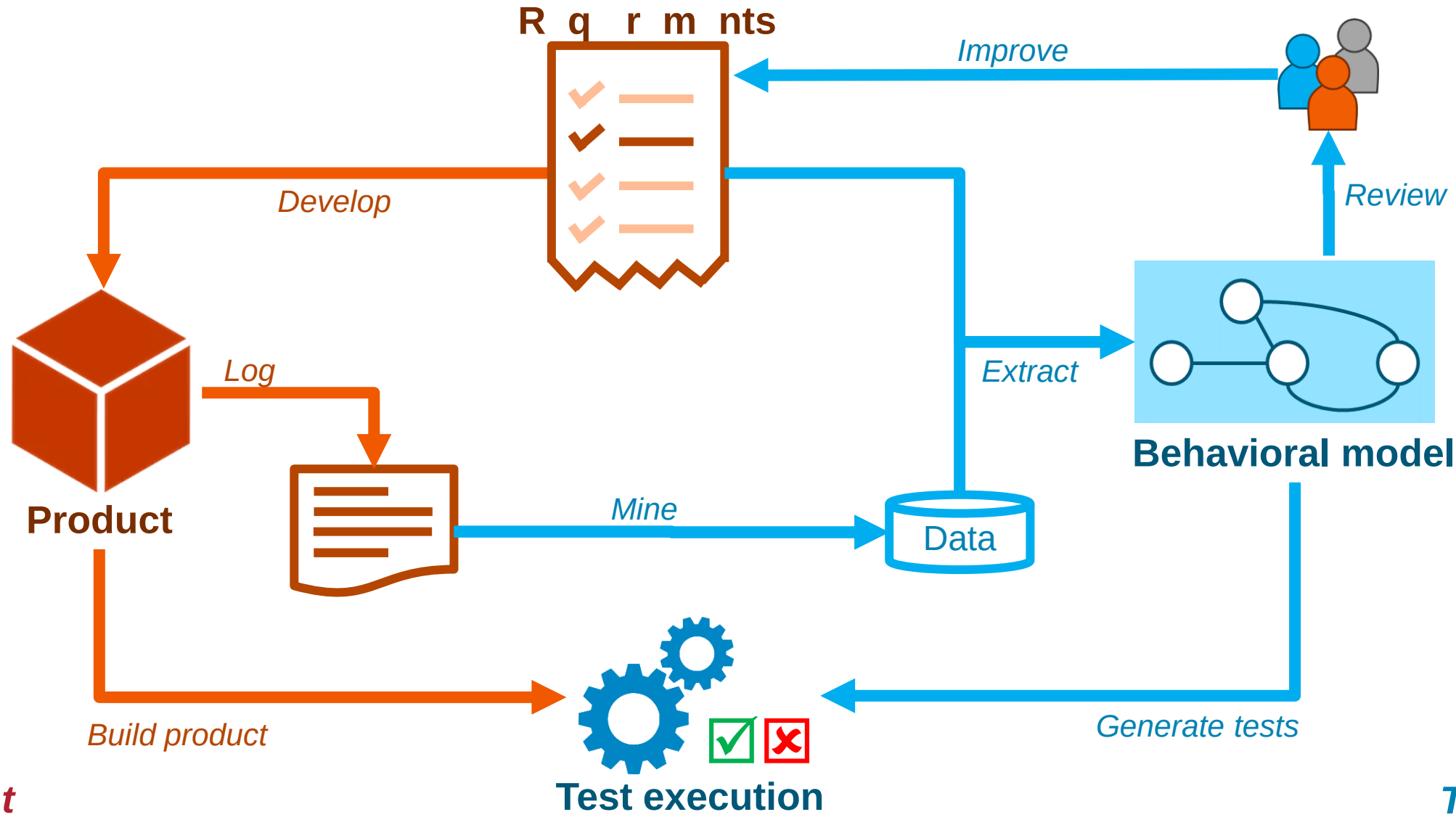
Process mining



Development

Test

Process mining



Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. **Taking up the challenge**
4. Future?

Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. **Taking up the challenge**
   a) **Complexity**
   b) State space explosion
   c) Our strategy
4. Future?

Challenges of MBT - Complexity

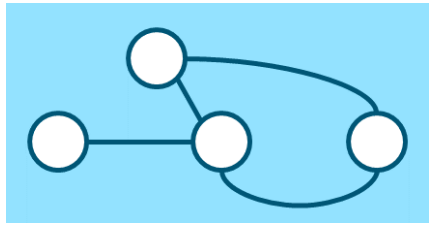
- Stakeholders have different expectations of MBT
- Modeling is a specialized skill
- Models need to be developed in the right level of detail
- There is a lack of mature tools

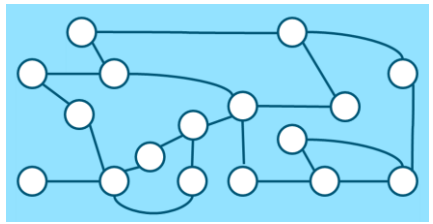
Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. **Taking up the challenge**
   a) Complexity
   b) State space explosion
   c) Our strategy
4. Future?

Challenges of MBT – State space explosion

- Expectations vs reality
  - Abstraction level of models
    Unclear scope of models leads to wrong abstraction level of models:
    - too abstract : model has limited to no added value
    - too detailed : high costs, state space explosion





  - MBT applied on too many areas → high costs, disappointing benefit
    - Apply only for high risk areas
    - Performance, reliability & security

Challenges of MBT – State space explosion

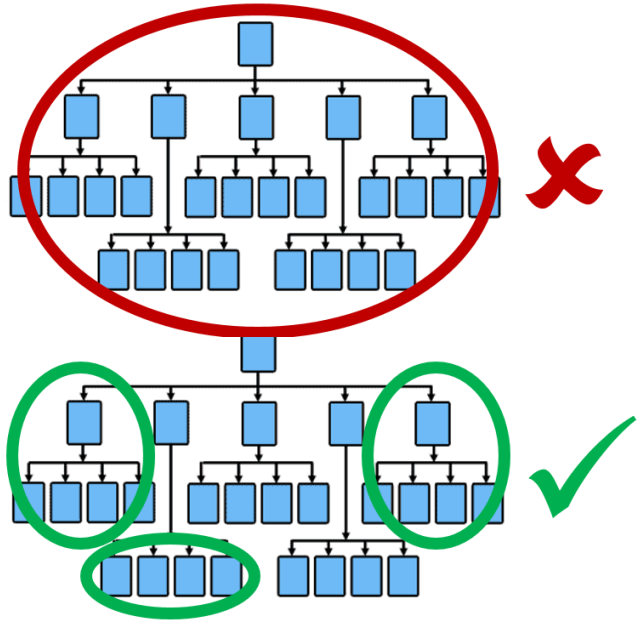
Dealing with state space explosion

Scope Focus on **components / subsystems**

instead of the **entire system**

a) simple models for different test purposes

b) based on risk analysis



Abstraction Focus on **behavior**, instead of **design**

a) model the behavior instead of the design

b) limit number of data values (use S, M, L iso range 0-1000)

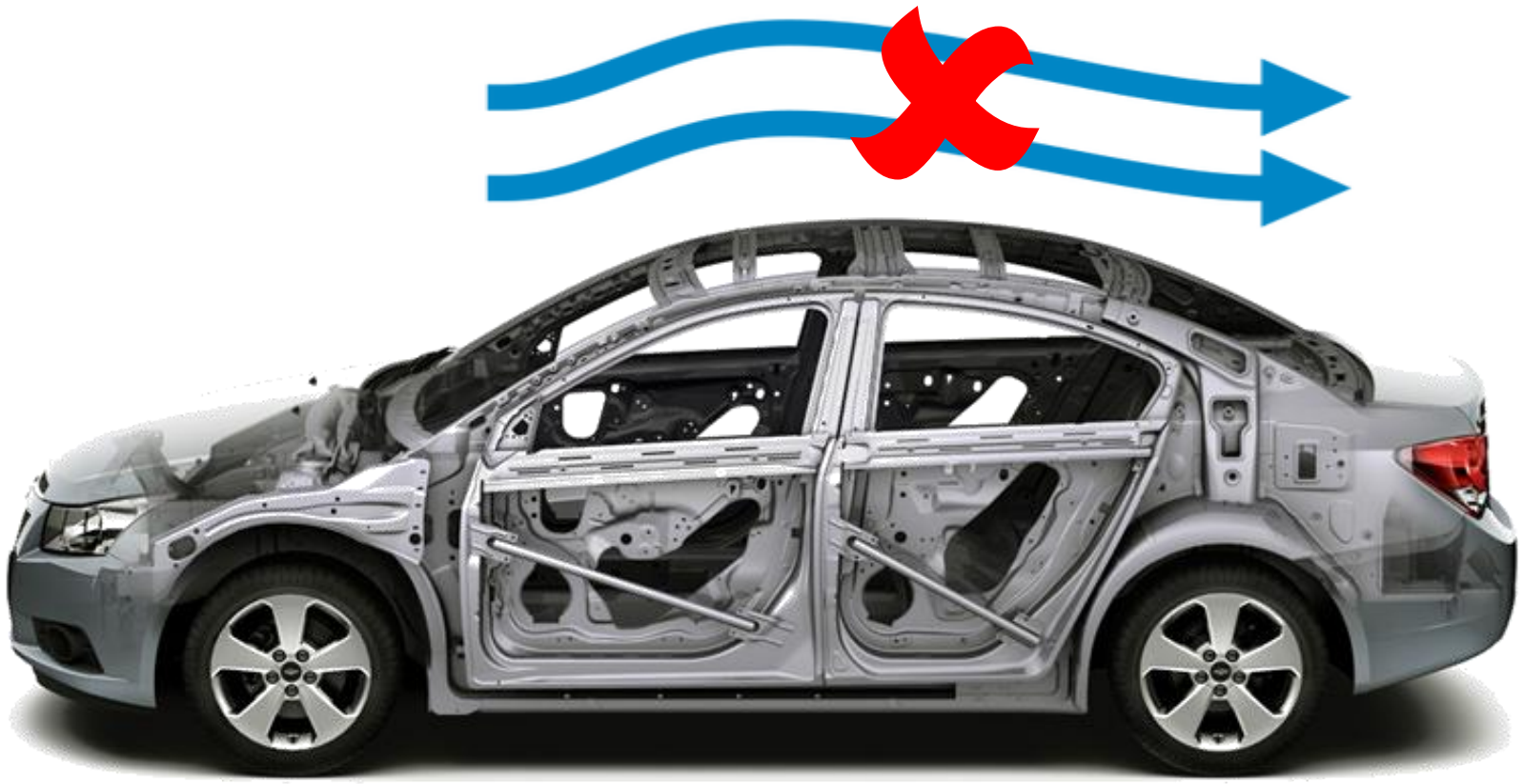
See also: Groote, Kouters, Osaiweran: Specification guidelines to avoid the state space explosion problem

Challenges of MBT – State space explosion

Different models for different purposes

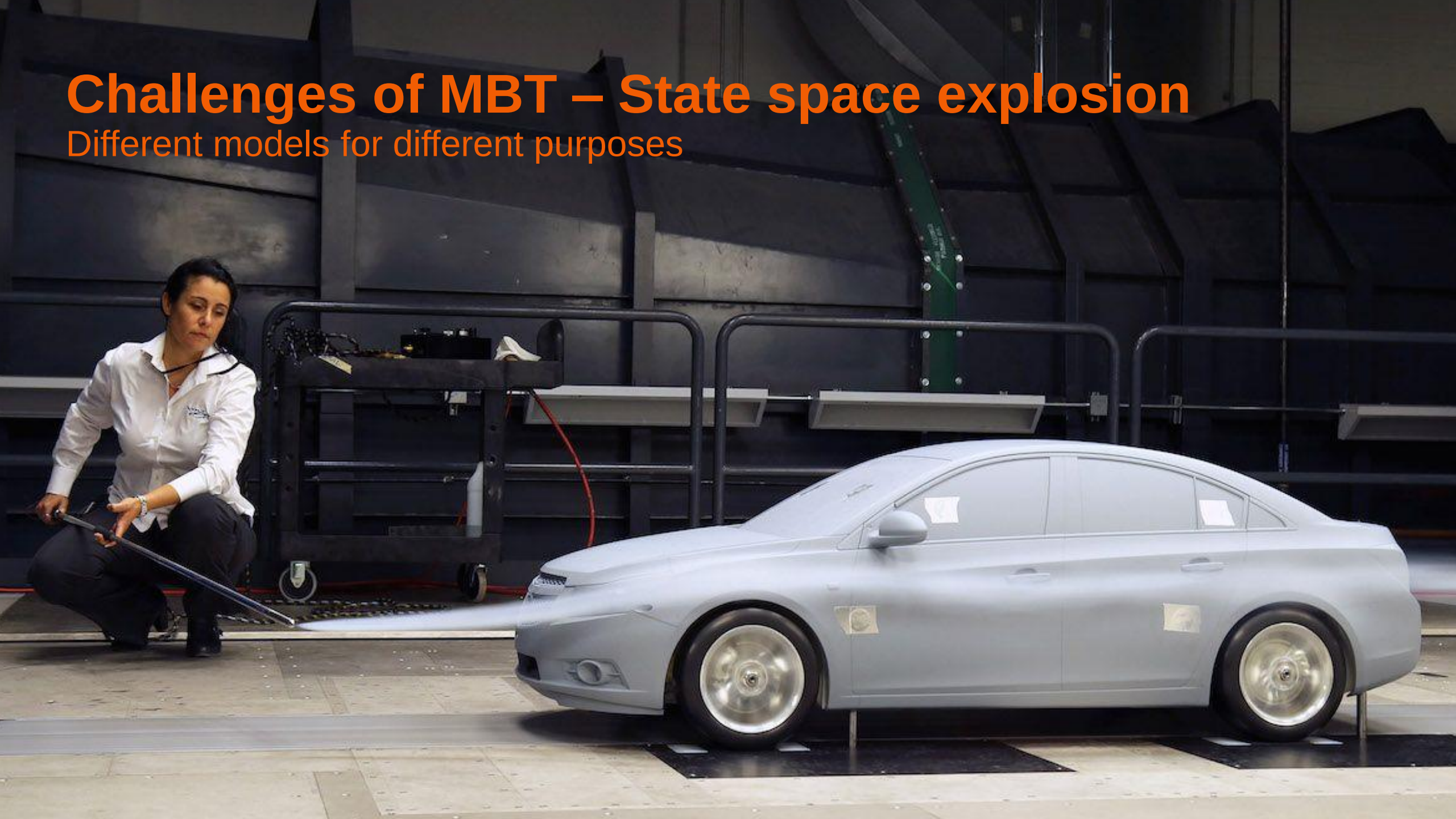
Challenges of MBT – State space explosion

Different models for different purposes



Challenges of MBT – State space explosion

Different models for different purposes



Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. **Taking up the challenge**
   a) Complexity
   b) State space explosion
   c) Our strategy
4. Future?

Challenges of MBT – Our strategy

Make MBT better accessible and practically applicable

- By education & experience
- By finding the proper tools
- By experimenting in smaller projects
- By divide & conquer (it’s all about abstraction):
  - combination of small and simple models vs one big model
  - create DSLs that generate (pieces of) models

Safely crash in virtual space

1. Evolution of software testing
2. The next best thing
3. Taking up the challenge
4. **Future?**

Future? – Skills of a SW tester

- SW engineering skills become (even) more important
  - Test Automation
  - Testing moves further to the left
- Modeling skills get into scope

Future? – Soon, this is not needed anymore

Erlkönig (Camouflaged prototype)

Future? - Iterating in virtual space

- Growing use of digital twins
- Fully virtual target environment
- Including autonomous driving

The Magic Roundabout, Swindon, England

Future? – Use of machine learning

- Manage state space explosion
- Find the critical/weak SW hot-spots
- Improve virtual environment with data mining

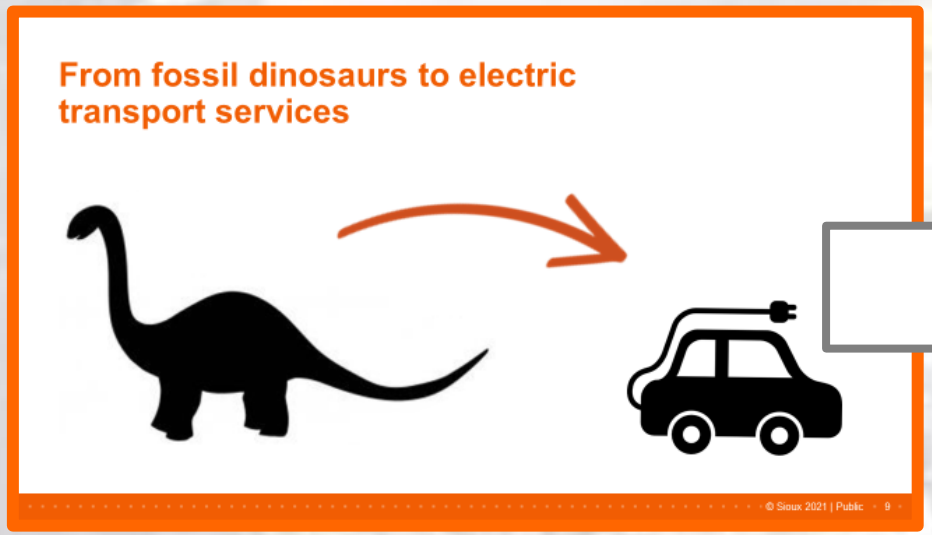
The Magic Roundabout, Swindon, England

Questions?

bryan.bakker@sioux.eu
dirk.coppelmans@sioux.eu

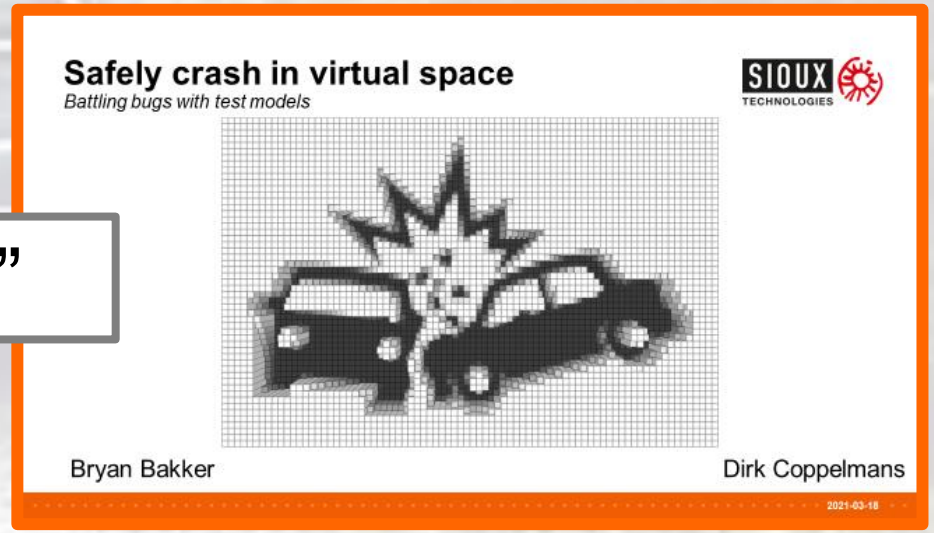
The Magic Roundabout, Swindon, England

Ask your questions



"@Jonn: <question>"

"@Bryan: <question>"



"@Crew: <remark>" (e.g. *no sound)*

Hot-or-Not Poll

Join the journey

- Contact us
  - Bryan.Bakker@Sioux.eu
  - Dirk.Coppelmans@Sioux.eu
- Follow-up session
  - You will be invited for our follow-up session on

    Tuesday 18-May-2021 18:00 (save the date)
- Watch the highlights of this Hot-or-Nots session
  - Visit our Sioux Technologies YouTube channel


TESLA


TESLA
KAPWING

Join the journey

- Contact us
  - Bryan.Bakker@Sioux.eu
  - Dirk.Coppelmans@Sioux.eu
- Follow-up session
  - You will be invited for our follow-up session on

    Tuesday 18-May-2021 18:00 (save the date)
- Watch the highlights of this Hot-or-Nots session
  - Visit our Sioux Technologies YouTube channel