Multi-domain Model-driven Development

Developing Electrical Propulsion System at Volvo Cars

Jonn Lantz

Technical Specialist, Electric Propulsion Systems @ Volvo Car Group

Jonn.Lantz@volvocars.com

Partners

Ulf Eliasson, (PhD stud) Volvo Car Group/Chalmers
Andreas Andersson, (PhD stud) VCG/Chalmers
Rogardt Heldal, Chalmers/Göteborg Univ.
Agneta Nilsson, Chalmers/Göteborg Univ.
Eric Knauss, Chalmers/Göteborg Univ.
Patrizio Pelliccione, Chalmers/Göteborg Univ.
Jan Bosch, Chalmers

Software Center

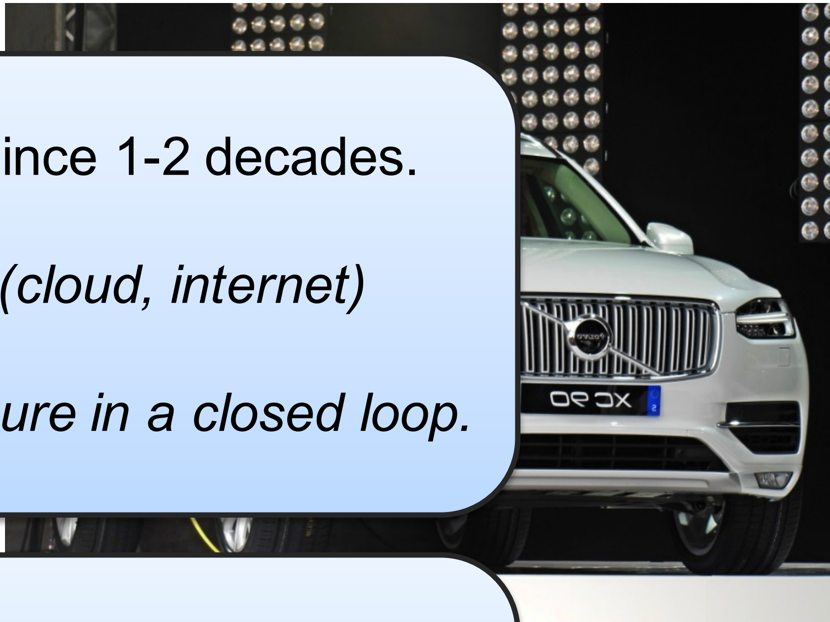
Mission: Improve the software engineering capability of the Nordic Software-Intensive industry with an order of magnitude

Theme: Fast, continuous deployment of customer value

Success: Academic excellence
Success: Industrial impact

CHALMERS
MALMÖ UNIVERSITY
JEPPESEN
ERICSSON
MÄLARDALEN UNIVERSITY SWEDEN
GRUNDFOS
AXIS COMMUNICATIONS

About Jonn Lantz, PhD:
Failed as physicist, failed as teacher, now working with car electrification, model driven sw (mechatronics) development and continuous integration flows.



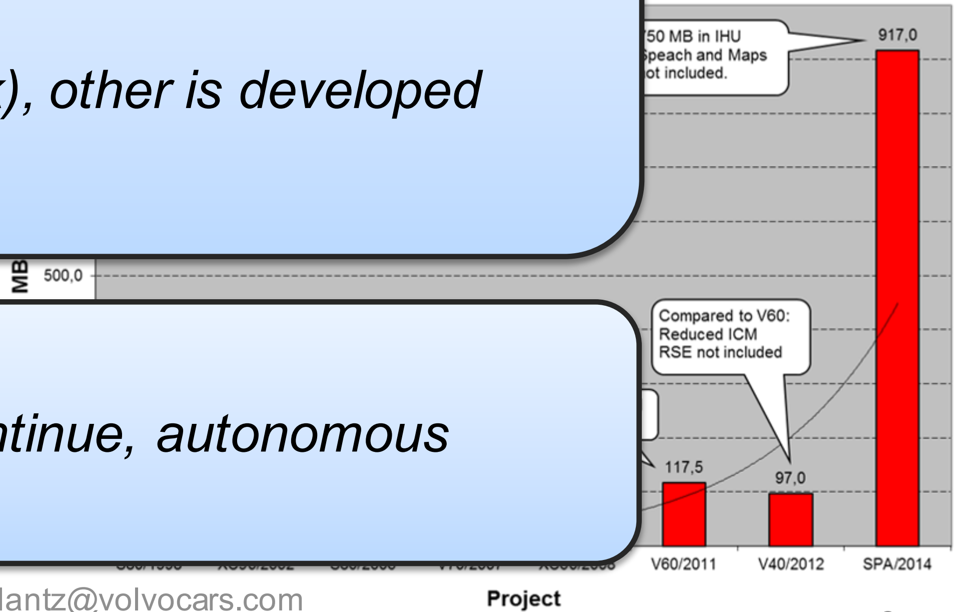
Development of a **complex mechatronic system** – but only since 1-2 decades.
Growth: ~ 10 x software in 7 years!
*The new SPA platform has over **100 ECUs** and is **connected** (cloud, internet)*

***Mechatronics** is an tricky domain! Real time software and nature in a closed loop.*

An **integration oriented business;** *mainly a* ***mechanical business**; professionals in "black box integration" of features.*

*Some sw is developed in-house (**MBD focus in this talk**), other is developed externally.*

Design ← V-model → Test



An industry in **rapid change**
The amount of software grows exponentially! This will continue, autonomous driving is approaching...

Buzzwords (some of them...)

Word: *Meaning:*

- **Continuous Integration (flows)**

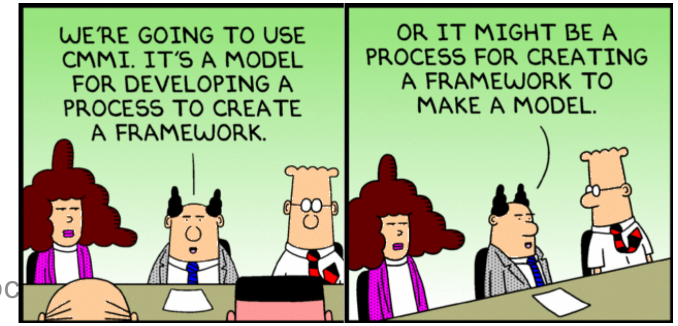
- Maintain 1 (*in-house*) software and be prepared to deliver at any time.
- Minimal time Δt between a new delta, ΔF, integration test of the new code and eventual delivery (to verification).
- *Fast feedback* and ***Automation*** are crucial.

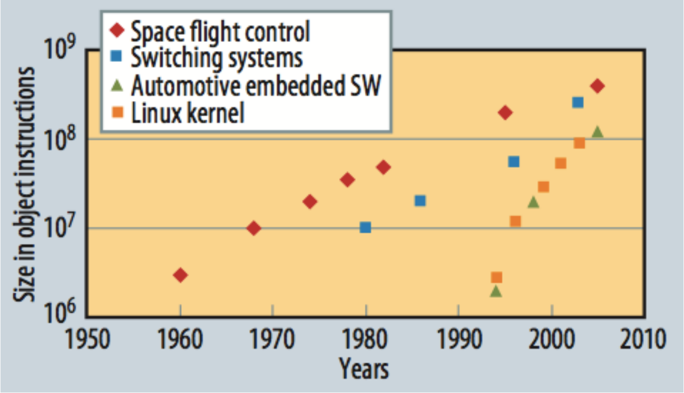
- **Cross Functional Teams**

- Work in teams organized by product functions or tasks.
- Organize groups by competence.
- Remove handovers

- **Model Driven Development**

- Utilize Domain Specific Languages to Gain abstraction and reuse
- Replace real integration with virtual

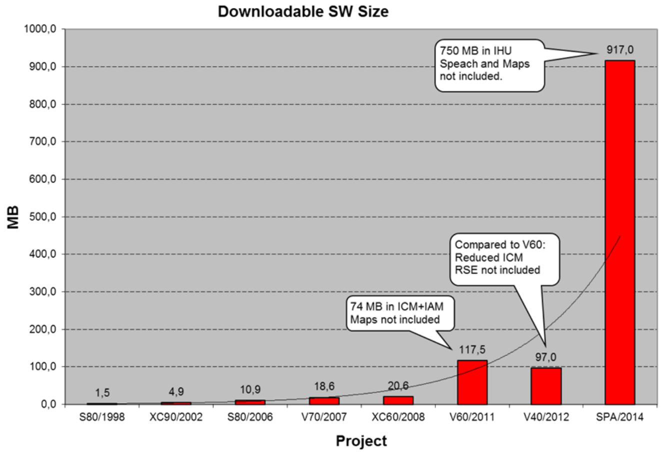




SPEED

is **relative***

and demands **control**



But, what if the (software-mechatronic) system we have is too complex to control?

Can we continue to be an integration company, when the software complexity explodes?

And, how can we be more of a software company when so many new functions are mechatronic? – electrification, autonomous drive, etc.

* Yes, we only have to compare ourselves with competitors as BMW, Audi, etc.
Or? Electric cars are simple, but yet a tiny market…

The age of software

Is here for the automotive business

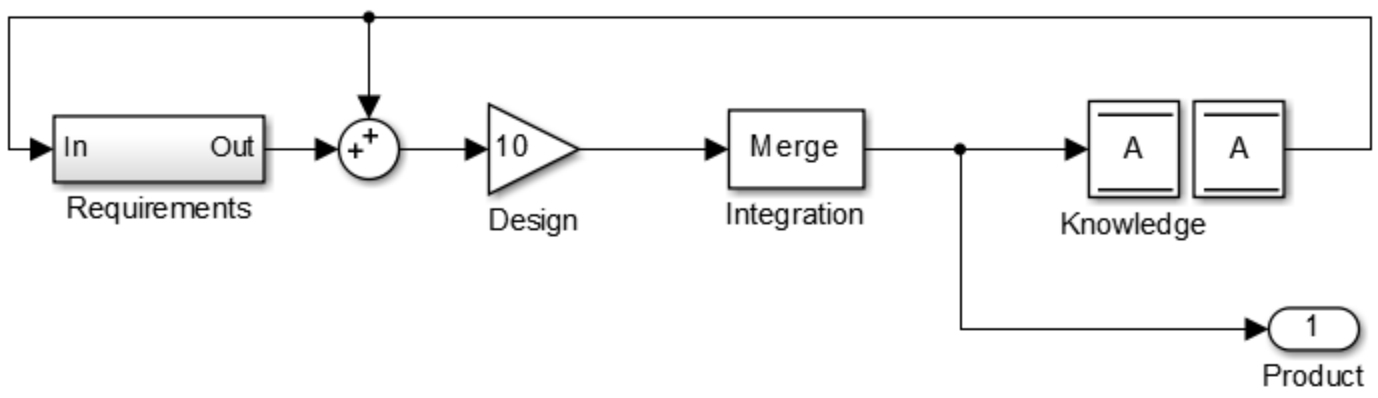
What if we have a new class of electric premium/luxury cars driven mainly by software development? Electricity is simplicity is speed.

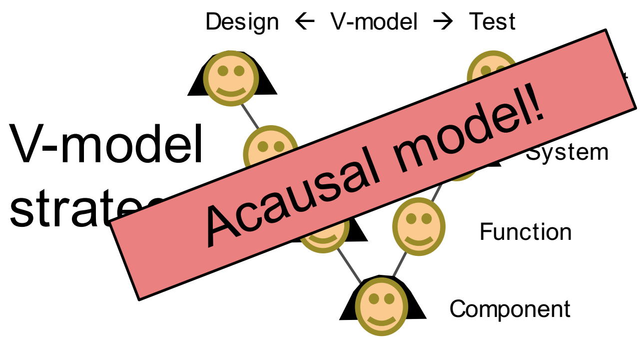
Thus, if we want to play on this market we better learn about electrification. Fast.






Actually... The main challenge is:

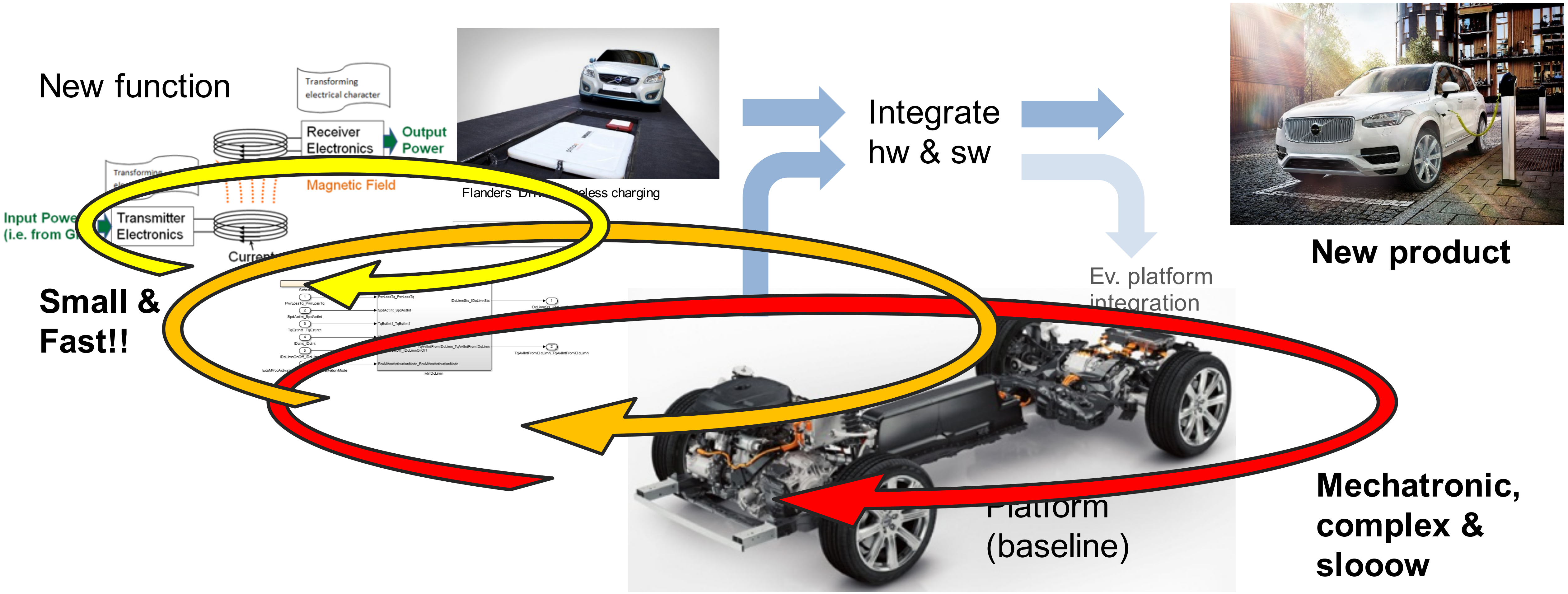




Continuous Integration

From planning to experimentation!

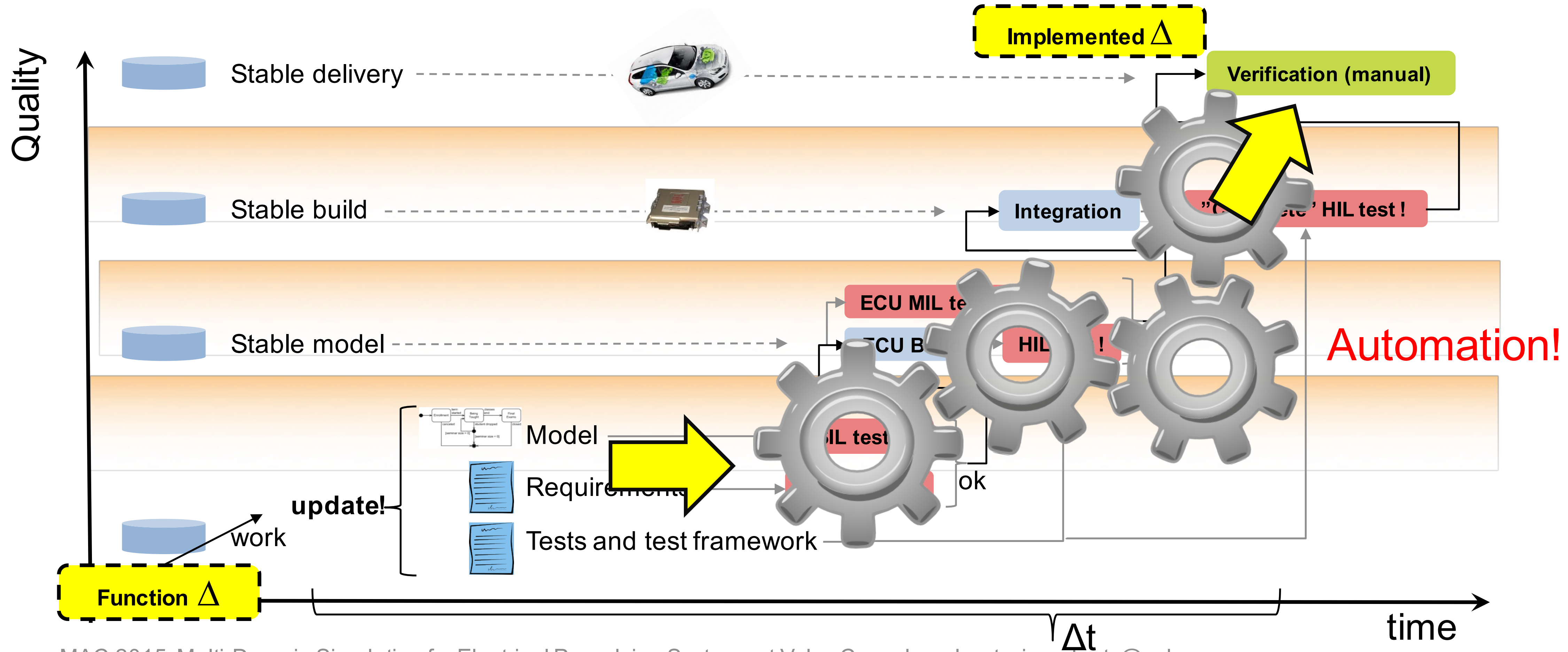
This is a software business trend, but we see it as well. Time is valuable and requirements change rapidly.



Speed – by [Model based] Continuous Integration

Eliminate the delivery

The software strategy of massive automation is translated to the automotive mechatronics domain



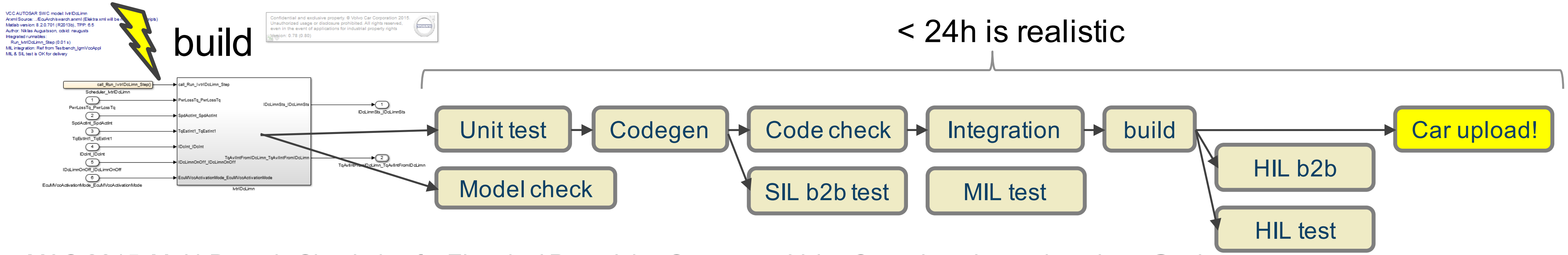
Continuous Integration in vehicles

- **Cars are complex systems.**
- We can consider them as **Product Clusters** more than individual products
- The product cluster may also change during the car's lifetime.

Some parts will be "hidden" (Tier1 software, of the shelf-parts) but in-house software can be integrated in the "latest" vehicle (baseline)

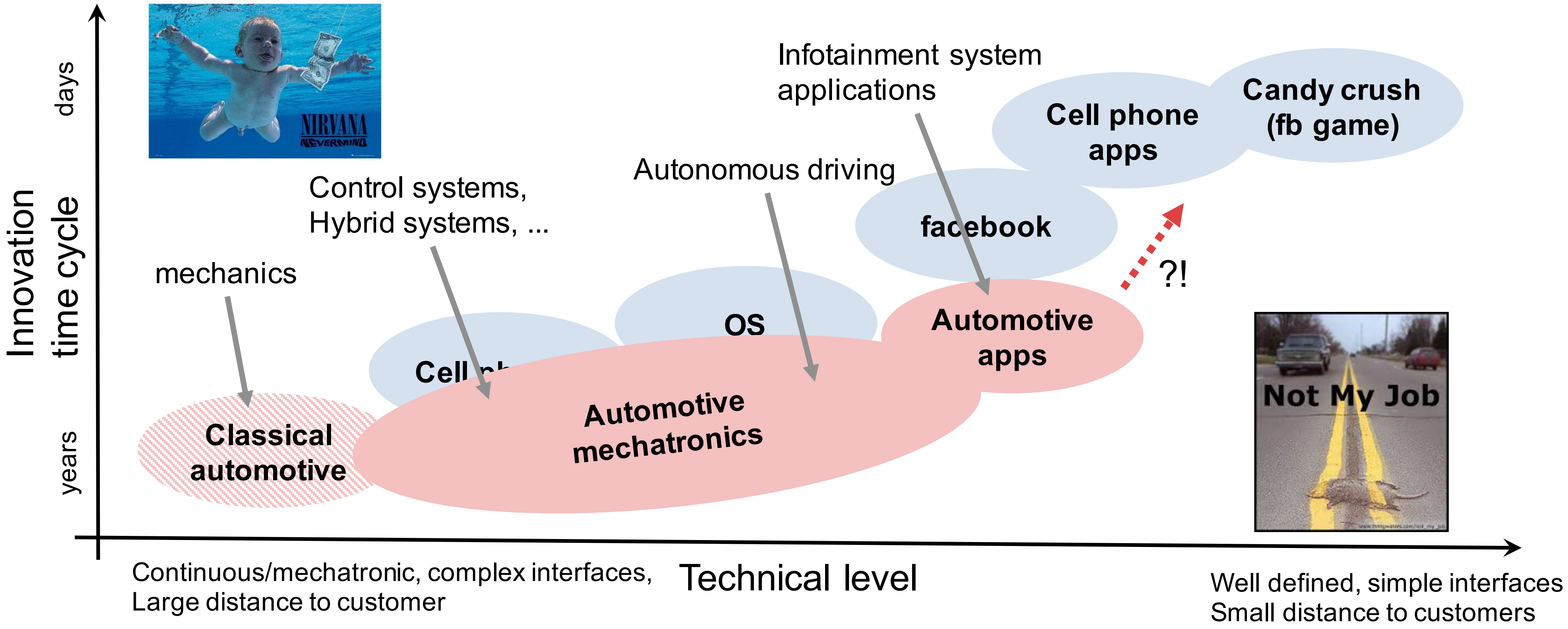
Hence, it is possible to have **Continuous Integration** and even Continuous Deployment – of sub system functionality

- Imagine a build button in your model, initiating a build flow finishing with an 4G sw-upload to one or several cars.



From one product to a cluster!

From one integration to multiple “integratables” (including the platform)

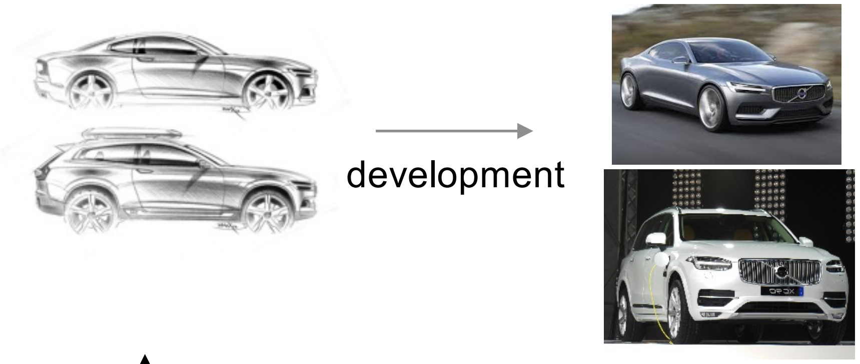


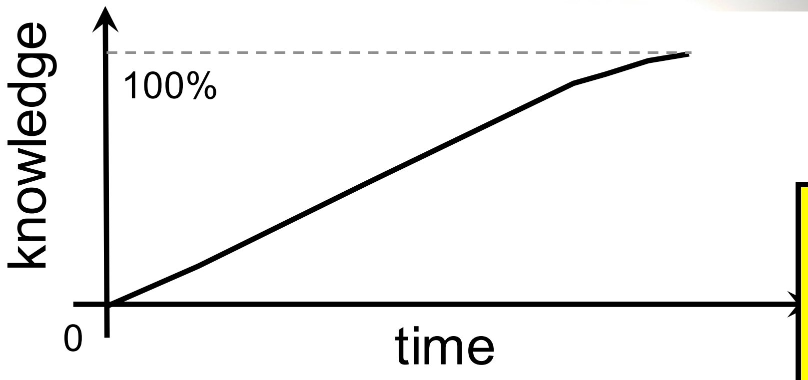
The evolving product

Legacy & Incremental development. A project is a Δ

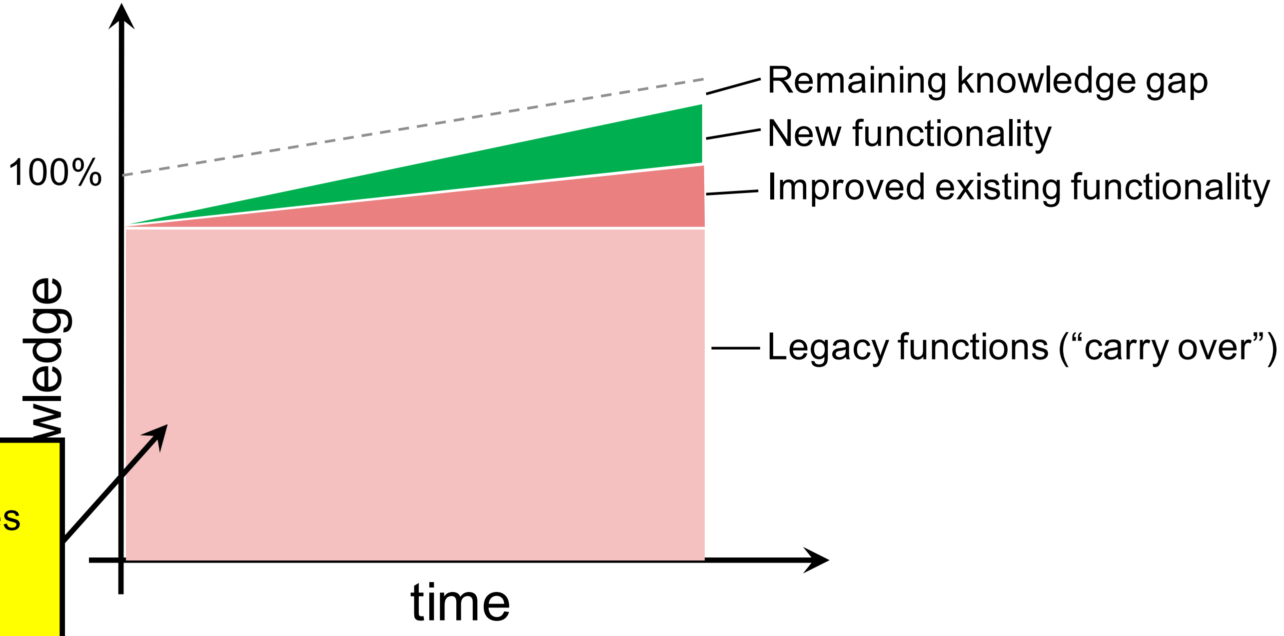
The popular understanding of a project, developing from scratch to a successful product, is wrong.

Popularized development:



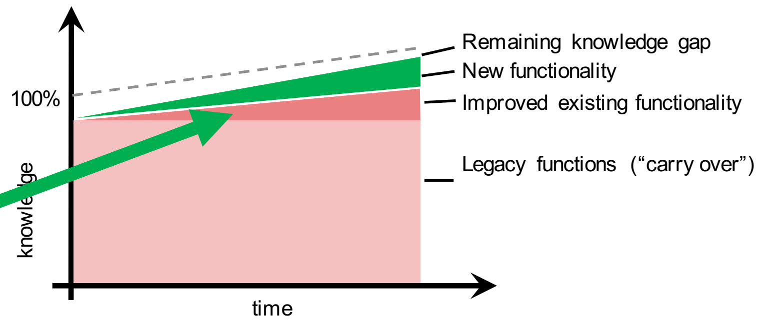


Product evolution:



In-house software and **modeling** makes sense! **Reuse** is important!

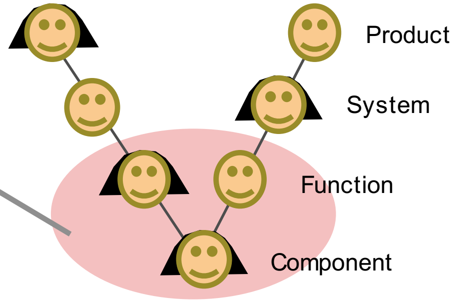
So, how can modelling help us?



For developing new functions:

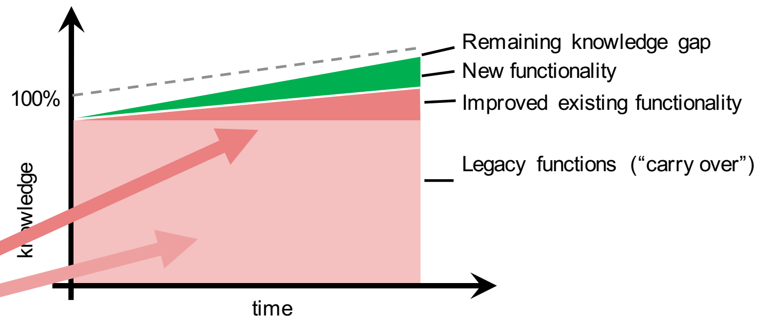
- It works! In the SPA platform (the new XC90) a significant part of the control software (combustion powertrain, active safety, hybrid system, body functions) are developed using Simulink using code generation to AUTOSAR platforms.

- Rapid [agile] development/feedback/learning before real hardware/mechanics is available.
- Maintaining knowledge – as "executable specifications"
- A cross functional team working with one common test bench



This was the old part. Now its time to take over the System level!

So, how can modelling help us?



For software platform* development:

- It works! In the SPA platform the body functionality is now in-house.
- **ECU-platform independence** – we can switch Tier1 without loosing knowledge.
- Rapid [agile] developm... functions)
- Maintaining knowledge –...
- A cross functional team w...

Extremely important with (full) AUTOSAR support – and design transparency, and readability (a challenge with AUTOSAR)

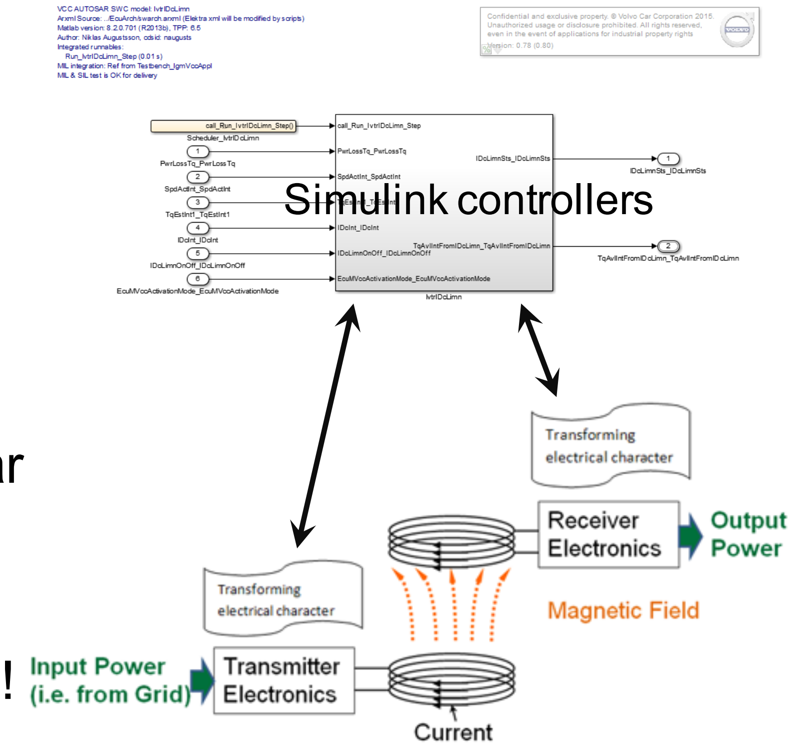
** Tricky definition: base functionality required in the car and for other (new) functions*

Let's take it further!

For virtual verification:

- The hybrid system in the new XC90 T8 is developed using Simulink with Simscape plant models. This approach helps rapid learning and experimentation on subsystem level (e.g. a battery or a motor) and **brings people together**.
- A cross functional team working with one common test bench!
- Other groups at VCG, e.g. at Powertrain, are using other similar languages (Dymola/Modelica, etc.)

Message: The **Domain Specific Language** (DSL) is here to stay!

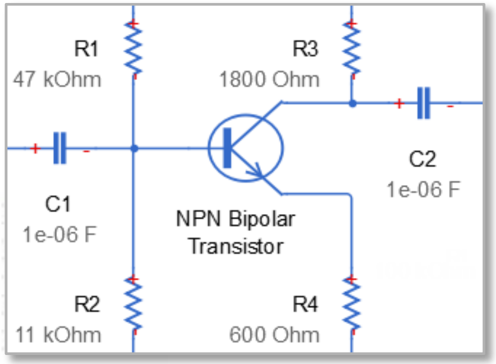


Physical DSLs

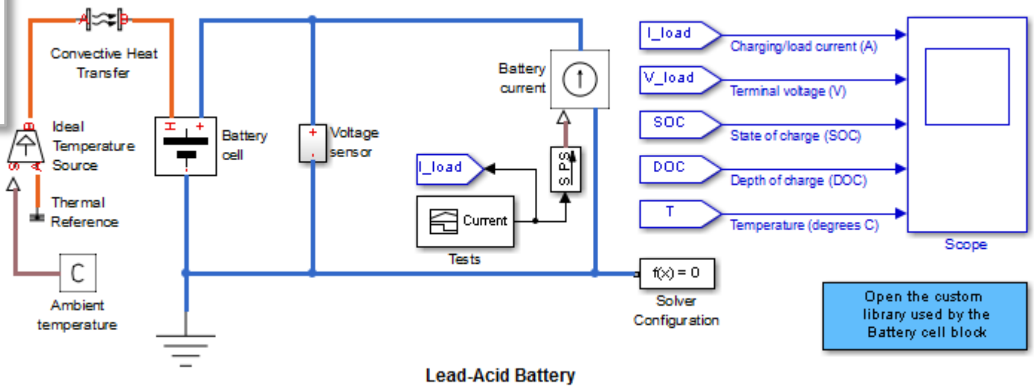
Introduction to (physical) DSL – Domain Specific Languages

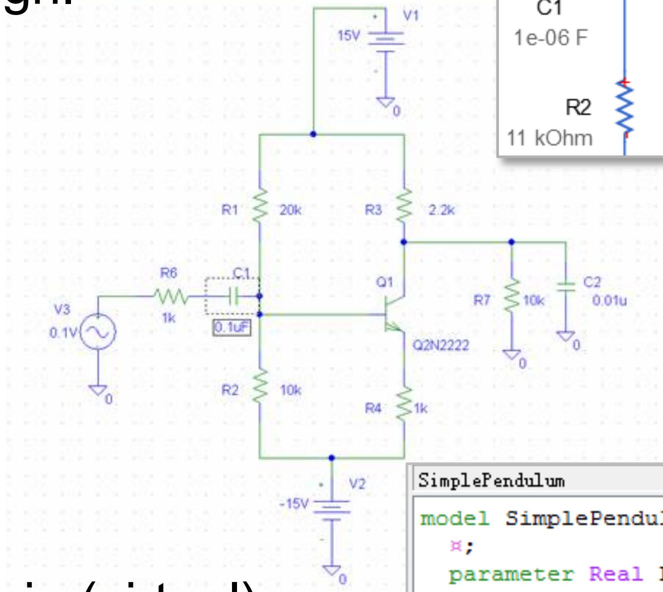
Optimize the language for your modeling, not the modeling for the language!

Optimize for readability and abstraction which is similar to text book modeling. Hide conservation laws and basic constraints from the design.




```
parameters
      c  = { 1e-6, 'F' };
      r  = { 1e-6, 'Ohm' };
      g  = { 0, '1/Ohm' };
end
equations
      v == i*r + vc;
      i == c*vc.der + g*vc;
end
```






```
SimplePendulum
model SimplePendulum "a simple pendulum"
  ;
  parameter Real L=2;
  constant Real g=9.81;
  Real theta(start=0);
  Real omega;
equation
  der(theta)= omega;
  der(omega)=-(g/L)*theta;
end SimplePendulum;
```

We gain speed and maintenance (simpler models!) but pay in (virtual) integration effort and simulation effort.

Acausal DSLs require more from solver and integration.

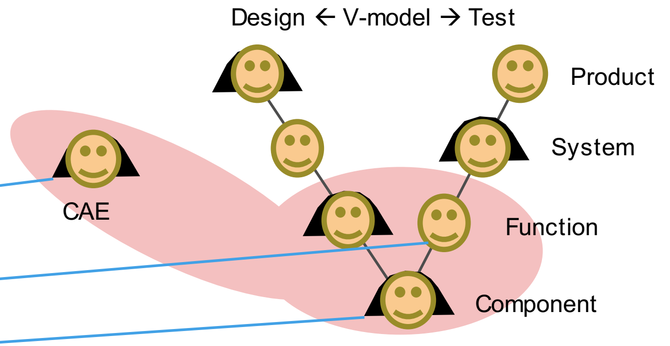
There is no free lunch!

Virtual development test, industrialized

Using Simulink and Simscape DSLs we can create an all-white box work bench:

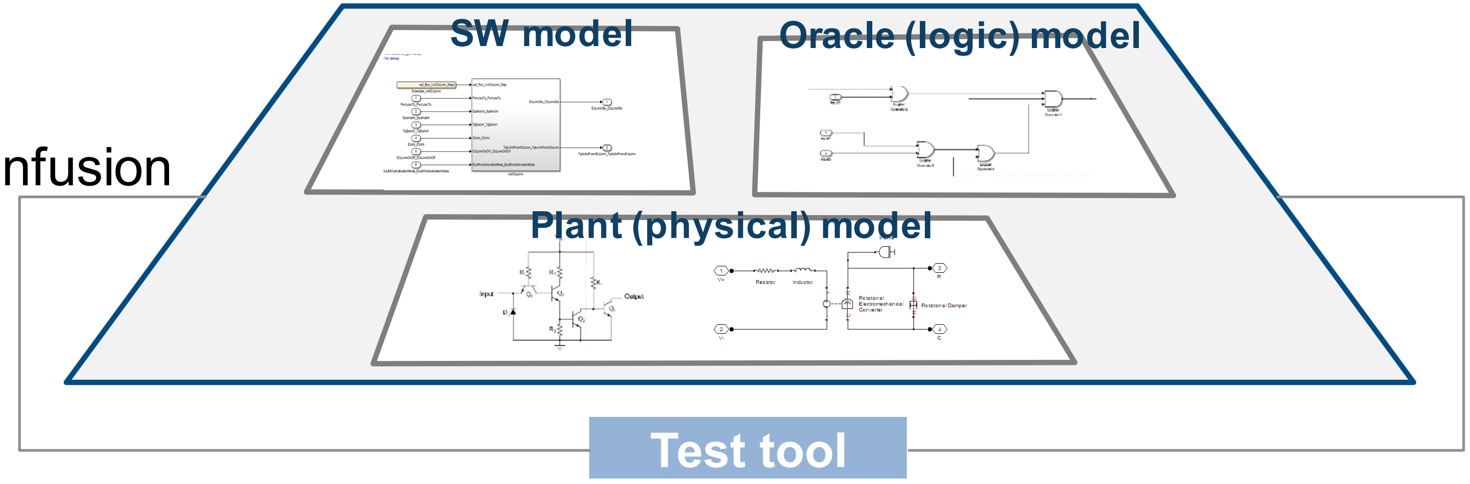
- A common model, work bench, for the cross functional team. Automated linking.
- The work bench is and interface to three "domains"

  - Plant model (Simscape or Simulink)
  - Oracle model (Requirement model, Simulink)
  - SW model (AUTOSAR Simulink model ref)



Benefits:

- One model, transparent, no confusion
- **Transparency**
- Advanced test tools can be used (as Design Verifier)
- Easy maintenance



An opportunity: “Formal” test methods on continuous systems

Formal methods are well known in software business

but the closed loop with mechanical (continuous) system creates an infinite state space!

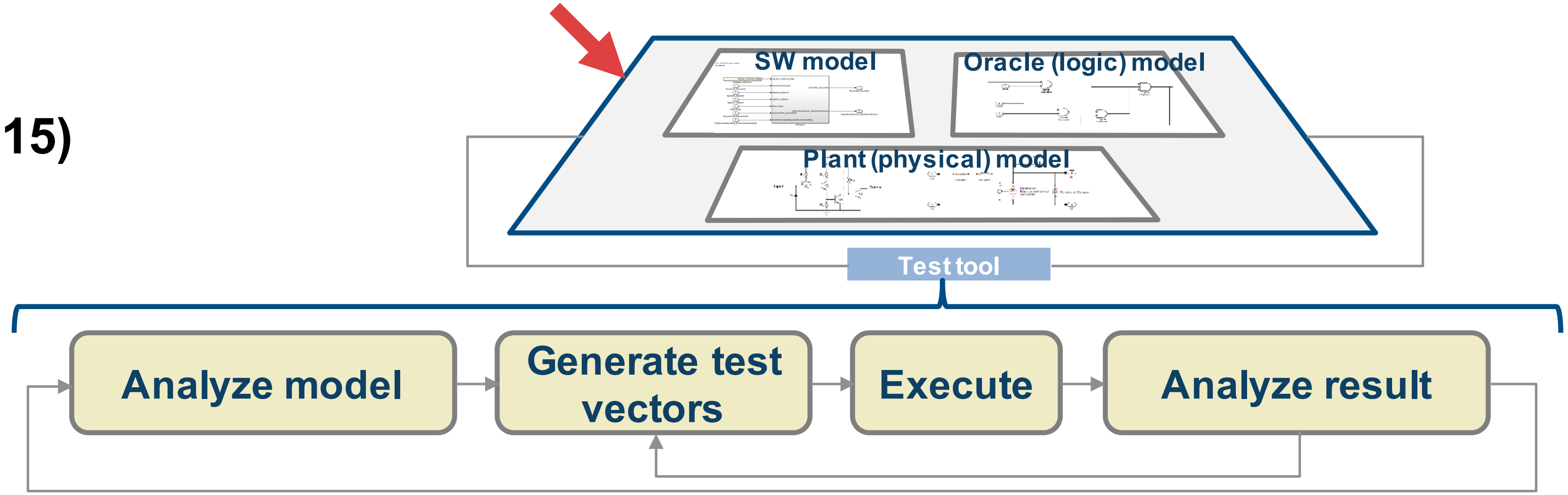
A numerical approach is required, but the model can still be analyzed.

Some tools exist (e.g. testWeaver from qTronic and Quickcheck from Quviq are tested)

This is an Executable Specification (if documented)

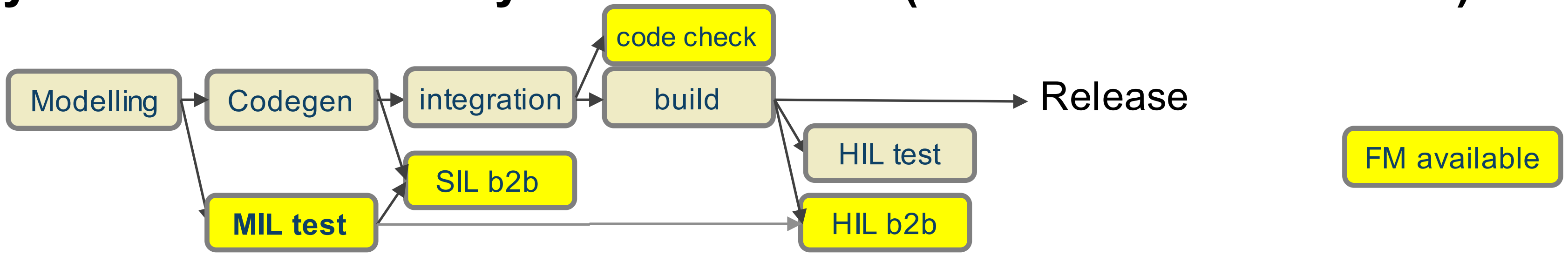
New research project (2015)

(VCG + Chalmers Univ.)

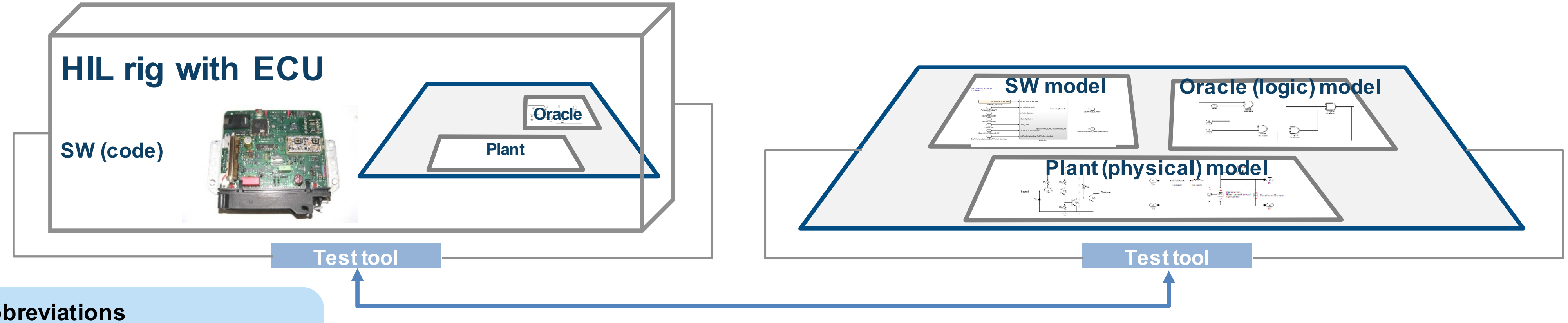


Another challenge: MIL – HIL back to back test!

Verify transformations of your SW model! (and follow ISO 26262-6)



- Verify that the control code running on target behaves as expected, as in the PC environment.
- Automated framework for functional (requirement based) HIL test.

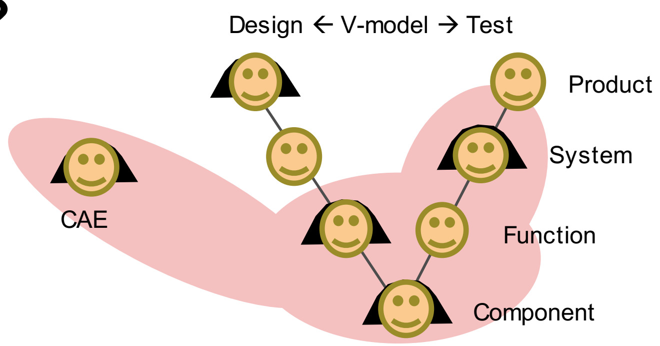


Abbreviations
MIL: test Model in the Loop
SIL: test (generated) Software in the Loop
HIL: test (code in ECU-) Hardware in the Loop

Verify identical (enough) behavior for all test cases

Let’s scale it even more!

- **Virtual System level test (manual HIL) has been used in SPA (XC90), for e.g. active safety functions.**
- What if we could have the vehicle – baseline – **”in the cloud”?**
- and move test from **HIL to MIL** (cheaper, faster, white box)



The library challenge (maintain **numerous models** and domains)

→ *The VVA, Virtual Vehicle Architecture, project is initiated*

The architecture challenge (extend system models to **include mechatronics**)

? In general tricky with system architectures. Now we need a mechatronic one…

The build challenge (integrate **external DSLs**, perhaps using FMIs)

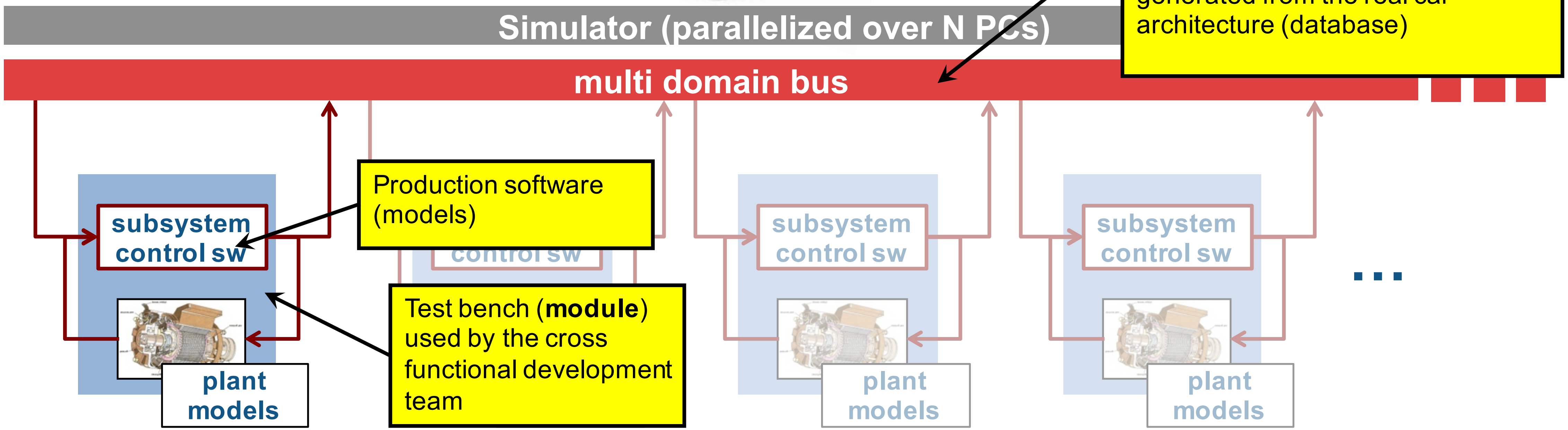
The simulation challenge (**large acausal models**(?), overall performance)

→ *This is a potential tool market, and business opportunity. Present demos are built in (causal) Simulink, using ad-hoc PC to PC-communication.*

Experimenting with Virtual Integration

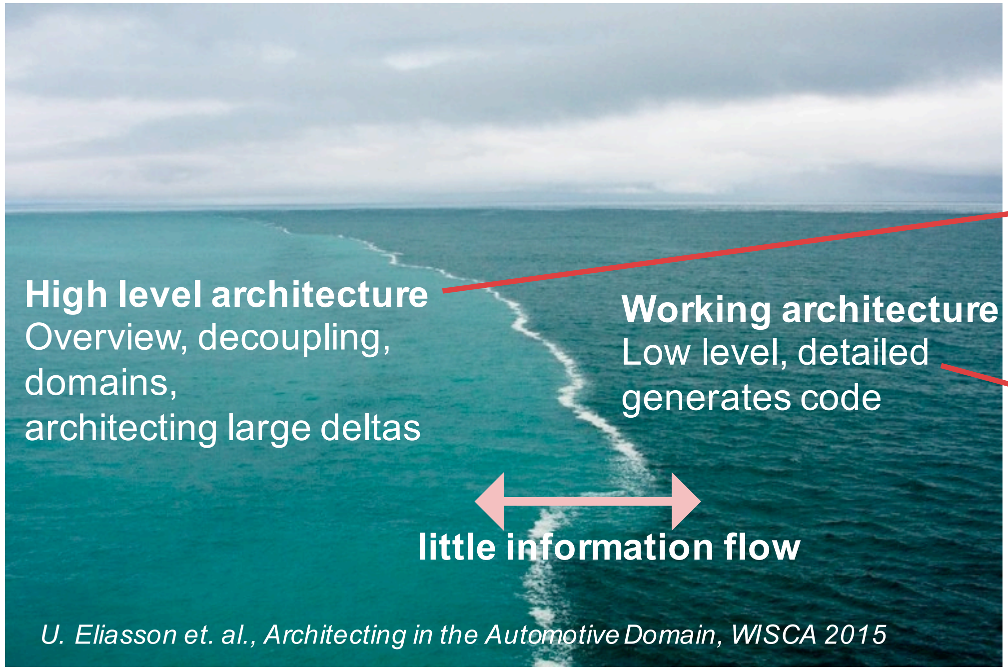
Full product simulation; complete vehicle MIL in Simulink (demo 2015)

- A complement to real vehicle integration (fast, cheap)
- Can be used for variant coverage in a broad product line

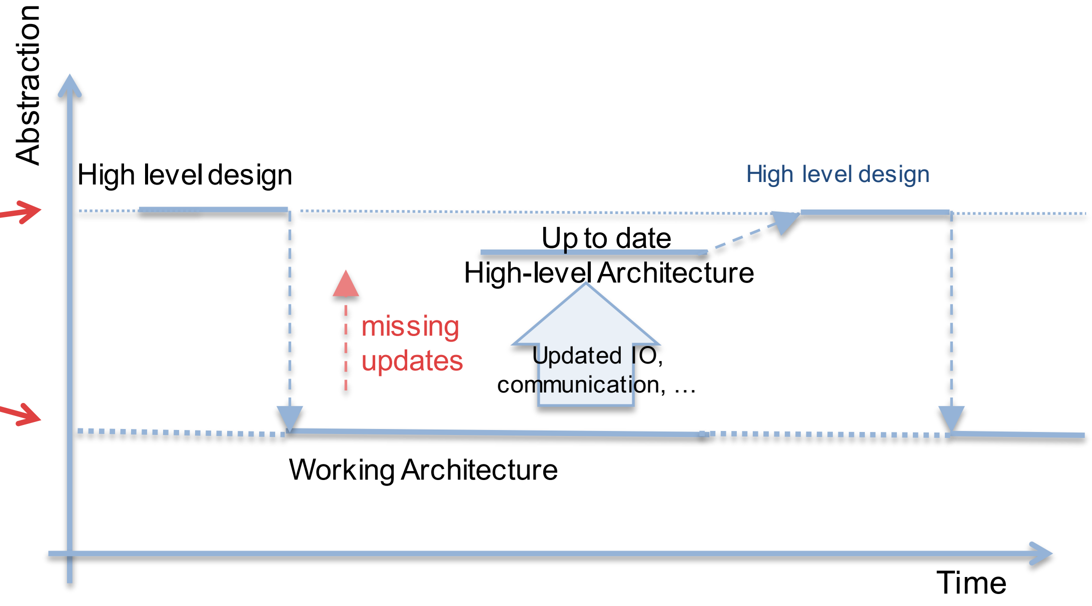


The Gap – system architecture vs. component development

- A huge challenge to keep **High level (descriptive) architecture** and **low level (prescriptive) architecture** up to date with each other
- A common problem within (automotive) organizations developing **complex mechatronic products**!? *The overview is lost when the details explode.*
- This may become a problem when the speed is increased and continuous integration enables frequent updates on ECU level.

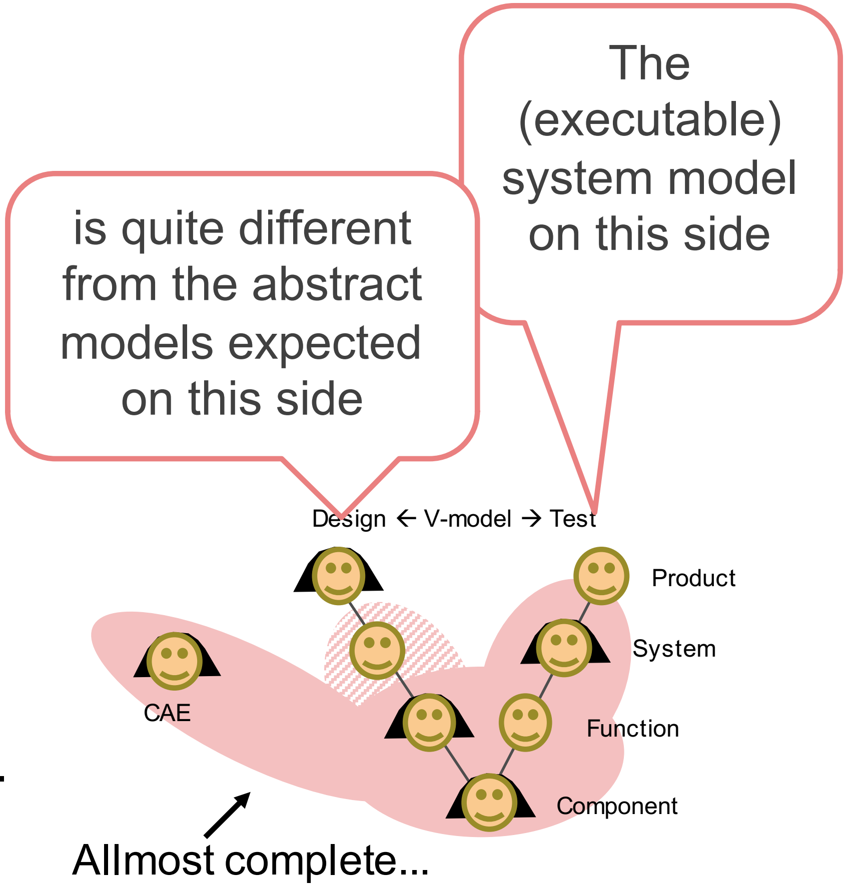


U. Eliasson et. al., Architecting in the Automotive Domain, WISCA 2015



The Gap – can models help us to extend the cross functional team to system design?

- **An updated system model**, generated from the current low level architecture could be used for system level design (and fast feedback)
- Has to be combined with a proper architectural design modeling language to be useful?
- Development of architectural deltas must be based on the present, true, architecture!
- An executable model of the present arch will be very useful.
- A challenge for development ecosystems (as MATLAB)!



*This **is** agile:*
Less (no) handovers, Teamwork, Transparency...

Conclusions

- The automotive business is transforming into agile mechatronics, at least for key functions, “VCG-DNA”
- This while the software complexity explodes
- Agile methods, Continuous Integration, etc. are introduced and spread rapidly

- The automotive business is **different, but not special**!
  - The mechatronics domain is nontrivial. Agile methods from sw business will not work right away.
  - The car is more of a cluster of closely interacting sub products, in-house made and external.
  - Time scales for mechanics, in-house sw, externally developed sw are very different.

- Executable Modelling is an enabler for agile mechatronics. Proven on component level, and growing
- Continuous integration to product combined with continuous virtual integration seems to be a good approach, but several challenges remain.

- The Volvo *will* be better and better for every day. *At least vital parts of it.*

Thanks for listening!