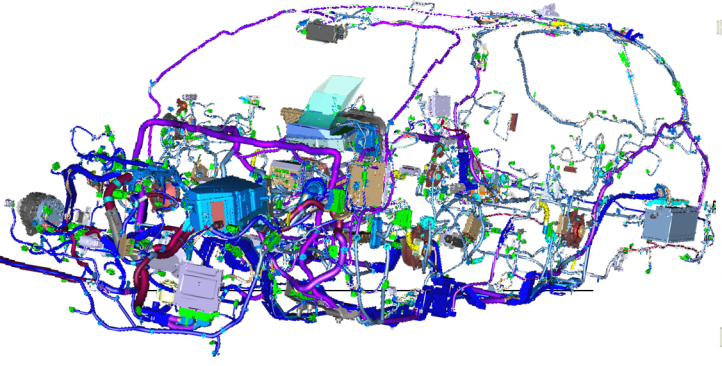
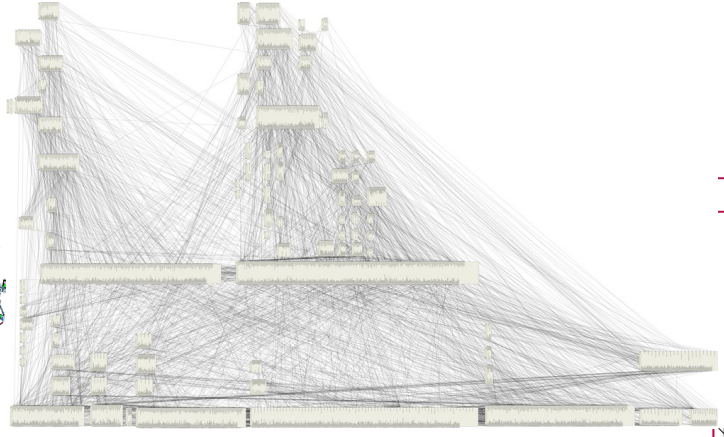
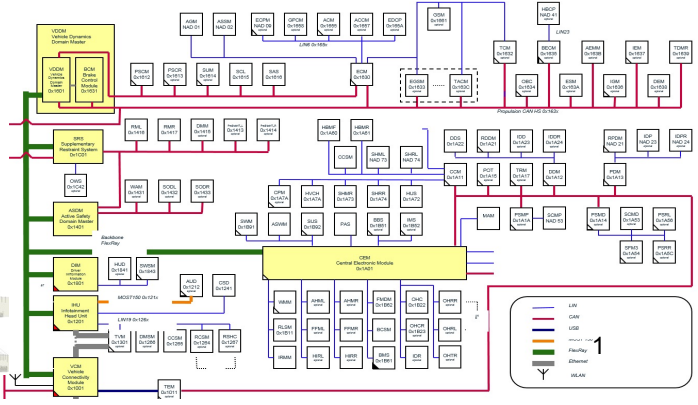
SOFTWARE COMPLEXITY AWARENESS

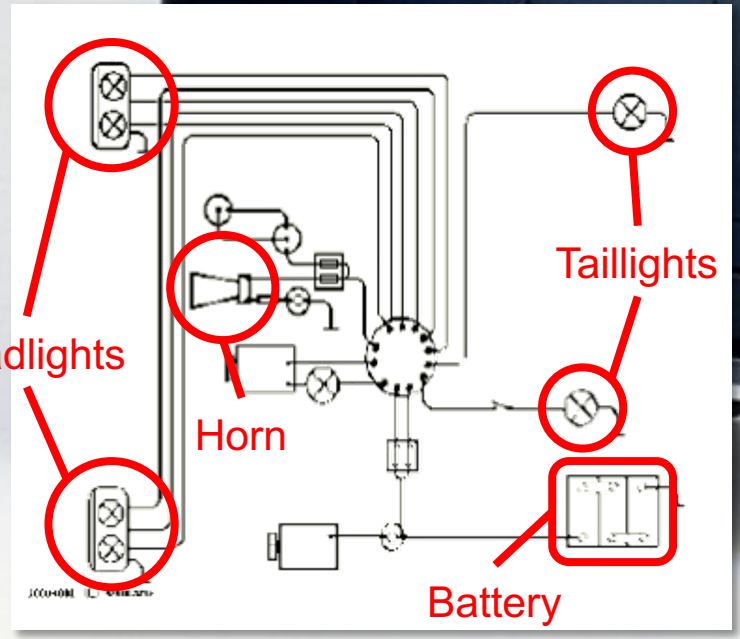
VARD ANTINYAN



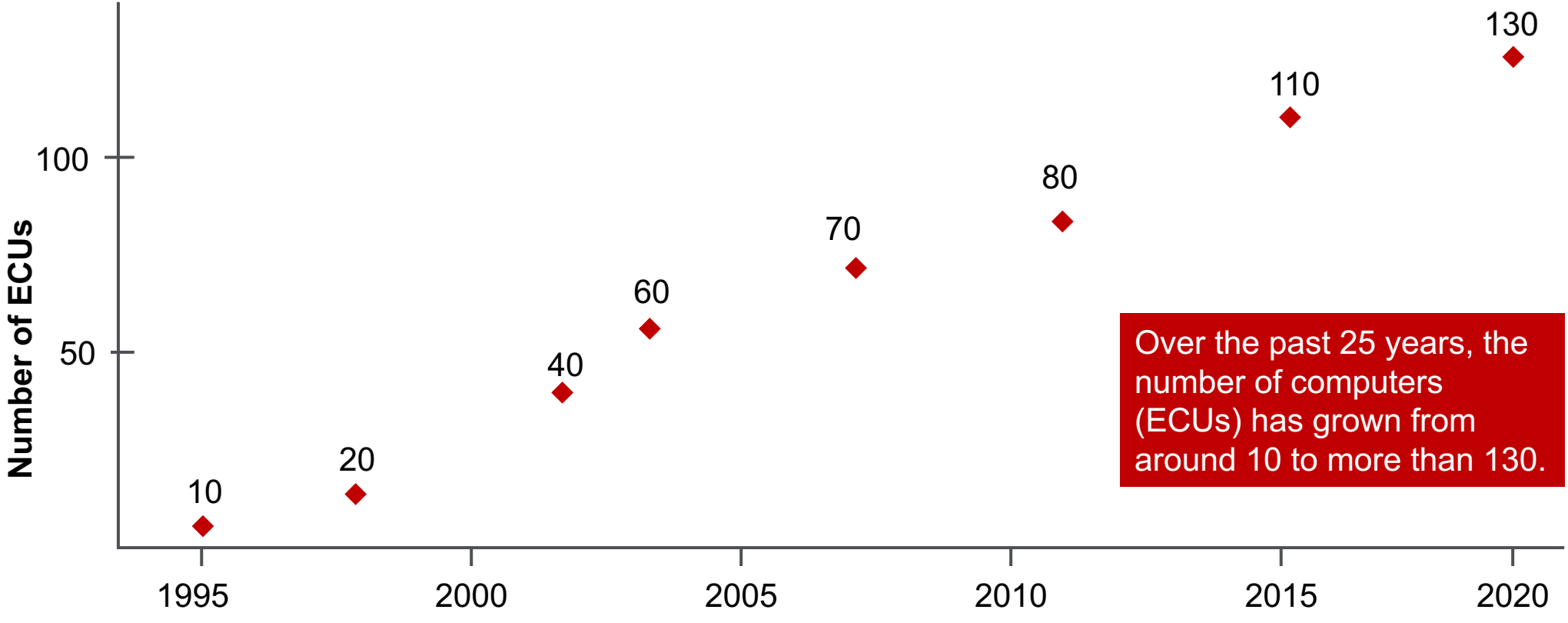




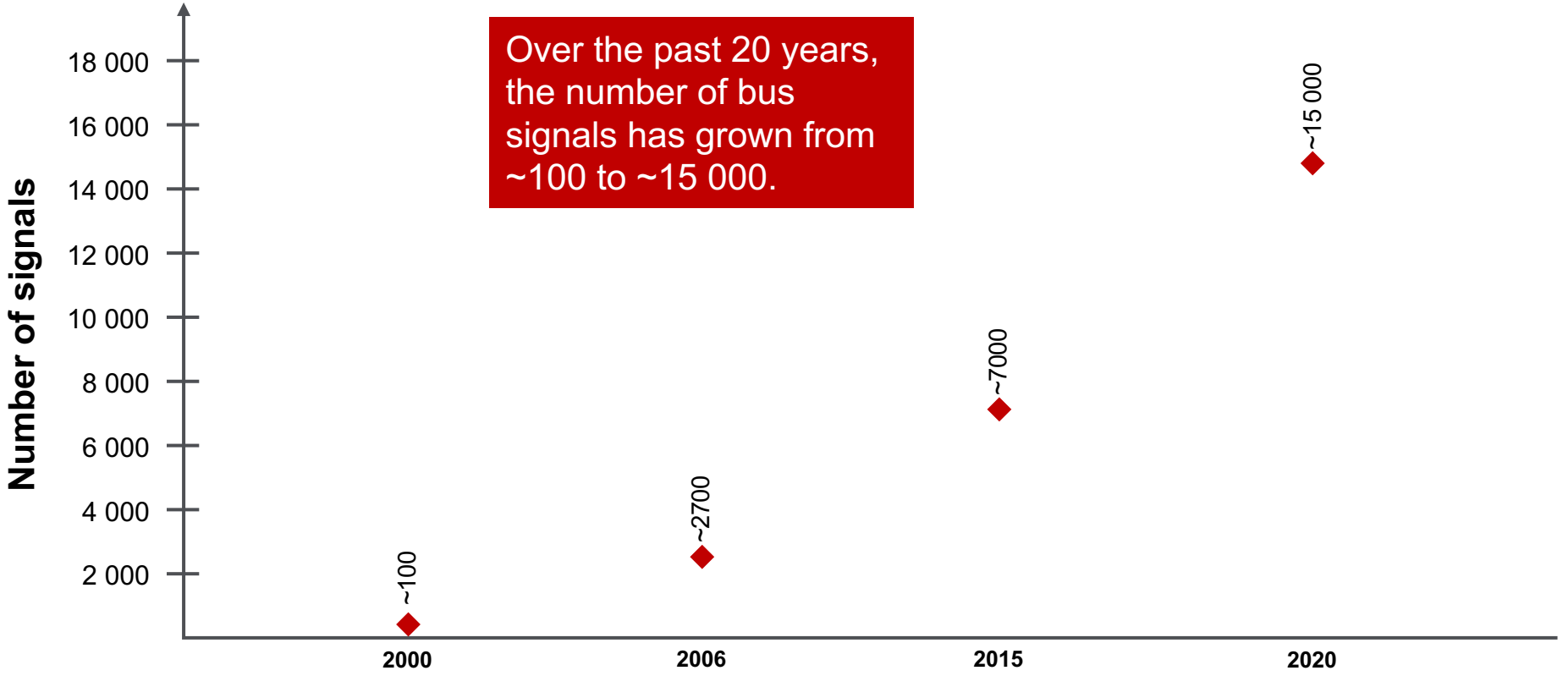
THE BEGINNING: 1927 VOLVO



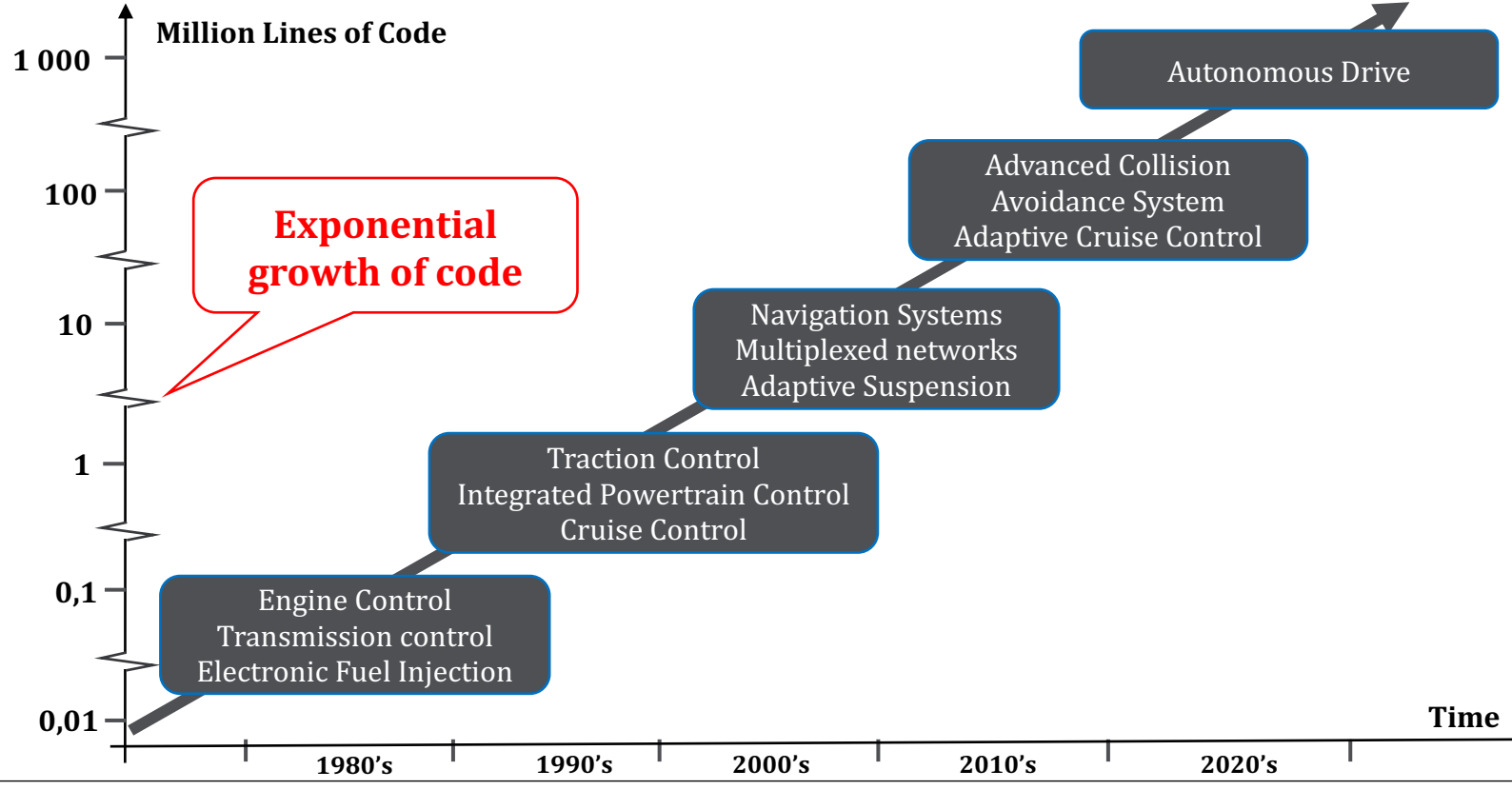
ECU COUNT EVOLUTION AT VOLVO



BUS SIGNAL COUNT EVOLUTION AT VOLVO



COMPUTERIZATION OF VEHICLE



WHAT DOES MANY LINES OF CODE MEAN?

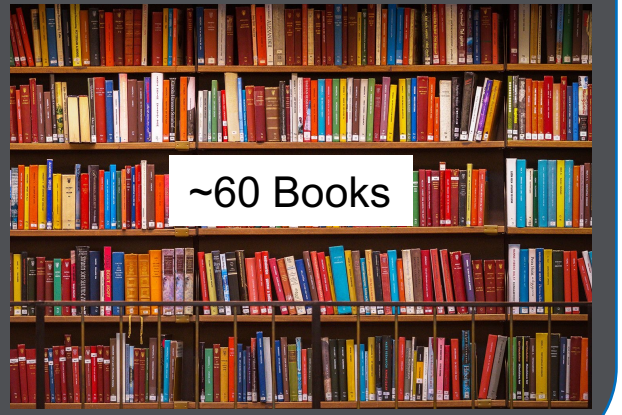
```
SpinDerivative = SignalConvatActSpeed * AntiSpinDW;
```

1 line of code (from propulsion code, Volvo)

Buck did no read the newspapers, or he would have known that ...

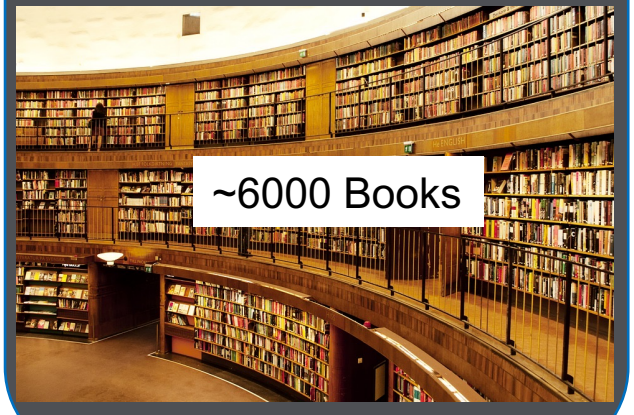
1 line of text (from “the call of the wild”, Jack London)

Volvo 2000: 1 Million LOC



Volvo 2010: 10 Million LOC

Volvo 2020: 100 Million LOC



WHAT DOES MANY LINES OF CODE MEAN?

```
SpinDerivative = SignalConvatActSpeed * AntiSpinDW;
```

1 line of code (from propulsion code, Volvo)

Buck did no read the newspapers, or he would have known that ...

1 line of text (from “the call of the wild”, Jack London)

Volvo 2030: 1 Billion LOC

How many logical and grammatical mistakes do you think there are in these books?

Equivalent to MIT Science Fiction Society Library

WHAT IS SOFTWARE COMPLEXITY?

- An emergent property of software due to increasing number of software **elements** and **interconnections**

- **Elements and Interconnections**
  - Structural
    - Operator
    - Control statement
    - Variable
    - ...
  - Representational
    - Line length
    - Indentation
    - Variable name
    - ...
  - Evolutional
    - Modified lines of code over time
    - ....

Reference
https://www.researchgate.net/publication/339377604_Evaluating_Essential_and_Accidental_Code_Complexity_Triggers_by_Practitioners'_Perception

VOLVO SOFTWARE COMPLEXITY

Some Aspects of Volvo Software complexity

- 10 million control statements
- 3 million function definitions called 30 million times

Volvo Software vs. human brain

- $3 \text{ x } 10^{6}$ functions vs. $1 \text{ x } 10^{11}$ neurons

Each function vs. Each neuron connection

- $3 \text{ x } 10^{1}$ called vs. 10^{4} connected

A PROPULSION CASE

Code written by Volvo for Propulsion:
Electric Drive control: **160 000 Loc**
Combustion Drive control: **800 000 Loc**

Total **1 MLoc**

Note: 1 MLoc is about 1% of the complete software

A PROPULSION CASE

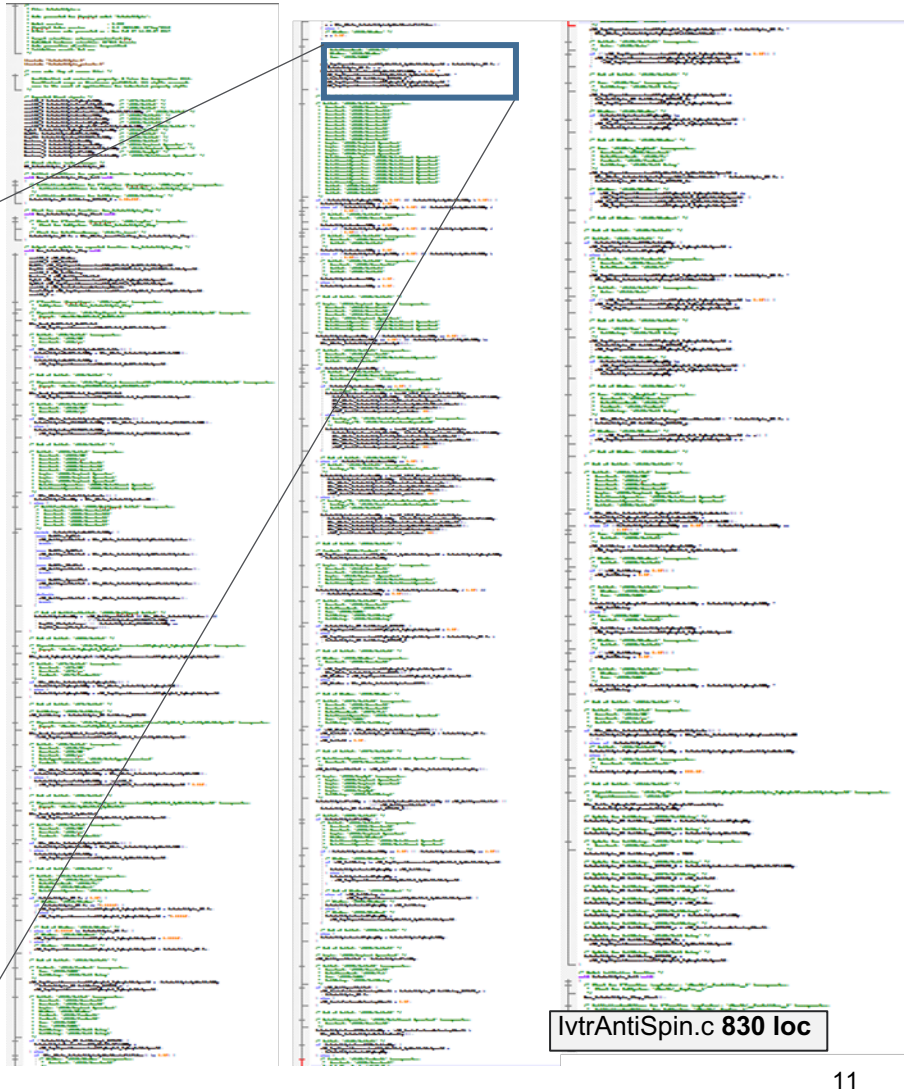
In Electric Drive Control:

1 Software component, IvtrAntiSpin has **830 Loc**

```
/* Product: '<S53>/Divide' incorporates:
 *  DataStoreRead: '<S53>/Ts'
 *  MinMax: '<S53>/MinMax'
 *  Sum: '<S53>/Add2'
 */
rtb_TmpSignalConversionAtSpdActInt_SpdActIntOutport2 = IvtrAntiSpin_DW.Ts /
  (IvtrAntiSpin_DW.Ts + u);
IvtrAntiSpinDerivativeC
  rtb_TmpSignalConversion
  IvtrAntiSpin_DW.UnitDelay_
  rtb_TmpSignalConversionAtTq
  rtb_TmpSignalConversionAtSp
}
```

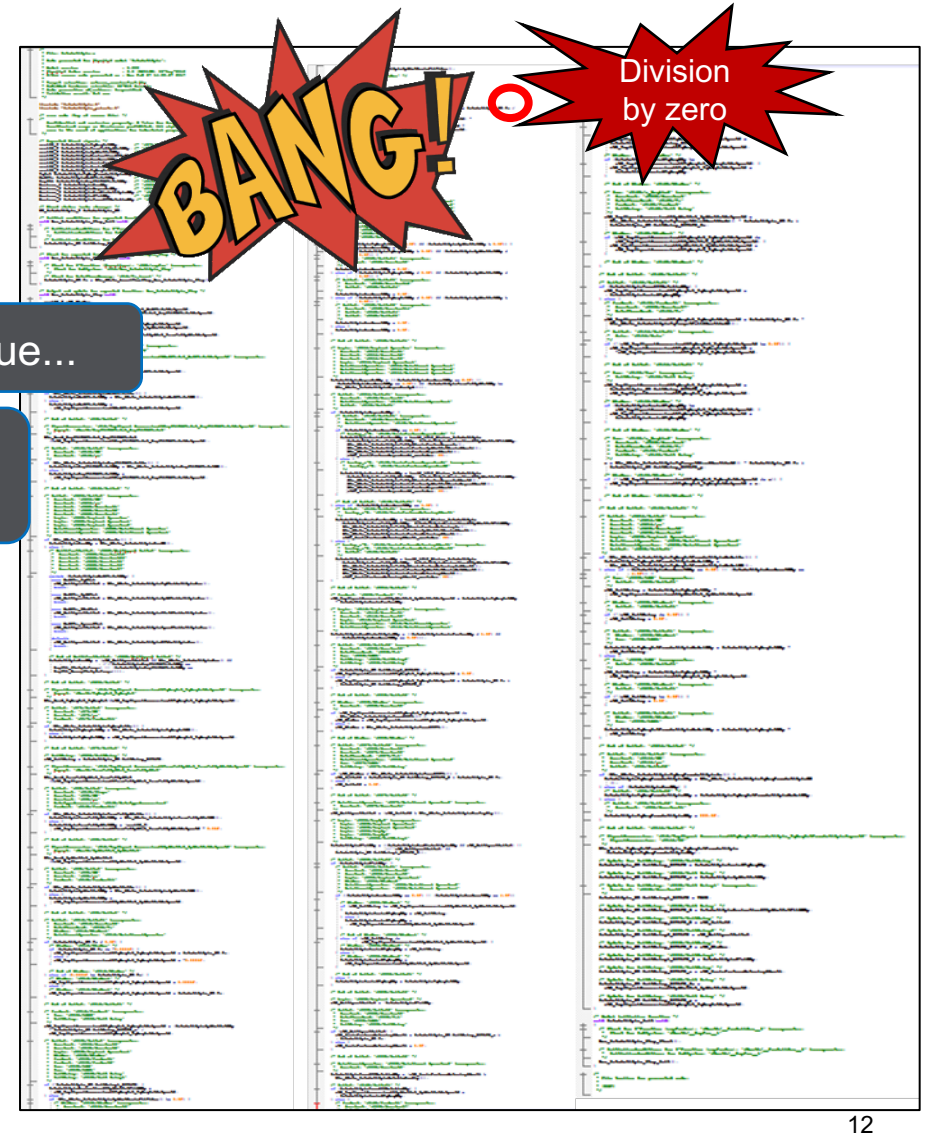
At line 303 there is a division sign.

Careful testing is required!
Division by zero is fatal



IvtrAntiSpin.c **830 loc**

A BUG IS TRIGGED...



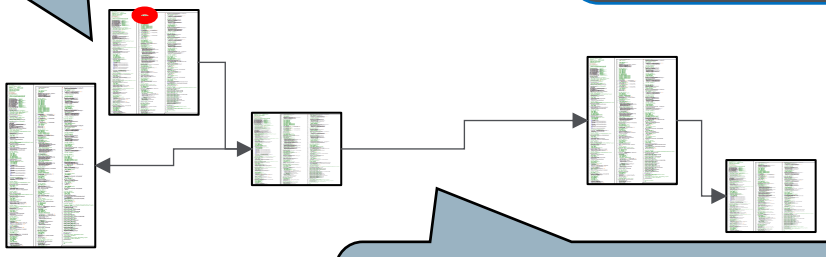
Issue...

Bug fixing...

Software update

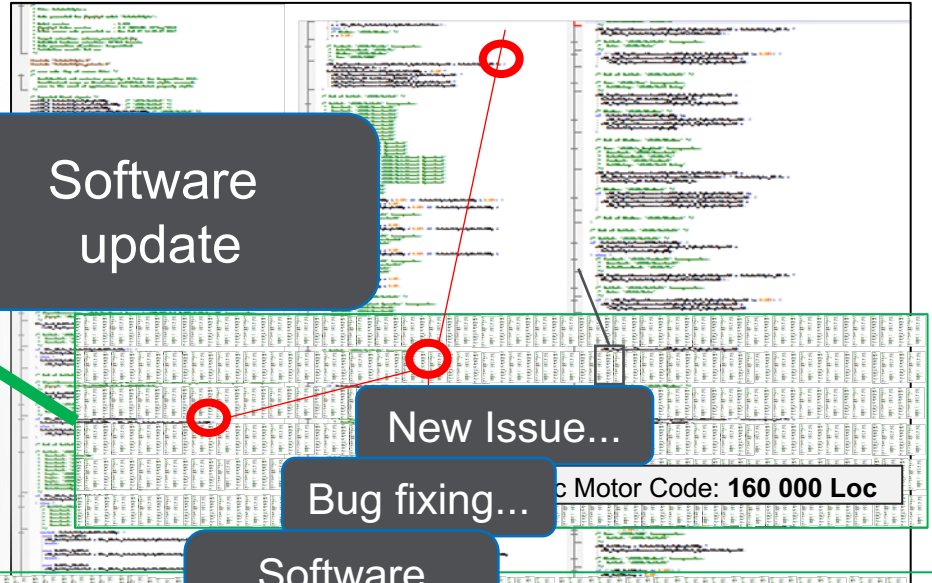
The required software update is quite large!

Several software components require changes



... and affect signaling to other ECUs.

WHICH CAUSED A CHAIN REACTION...



Software update

New Issue...

Bug fixing...

Software update

Issue...

Bug fixing...

Software update

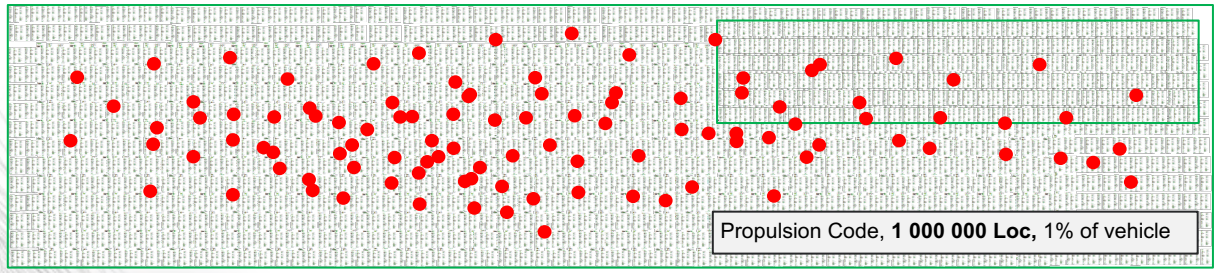
Issue...

Bug fixing...

Software update

c Motor Code: **160 000 Loc**

Combustion motor Code: **800 000 Loc**



Complex software in average
- Developed SW: 1 defect / 200 – 3500 loc.
- Shipped SW: 1 defect / 10000 loc.

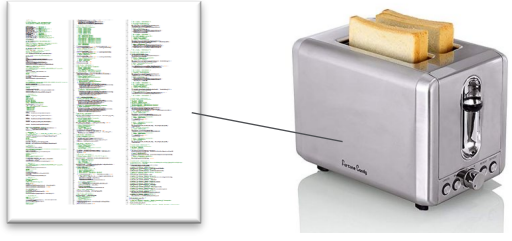
Unmanaged Complexity
generates Technical Debt
for which we pay interest rate regularly

Volvo 2020 **100 000 000 Loc**

TECHNICAL DEBT EXPLAINED

Small product
1000 LOC

No technical debt!

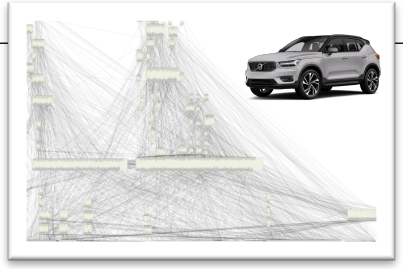


Total Cost = 50 KSEK

Invested money for new feature development	Cost of Fixing Bugs

TECHNICAL DEBT EXPLAINED

Large product
100 000 000 LOC



Total Cost

Invested money for new feature development

Essential **Complexity** **Accidental**

Cost of mastering the existing complexity to enable new feature addition

Cost of fixing bugs

Visible

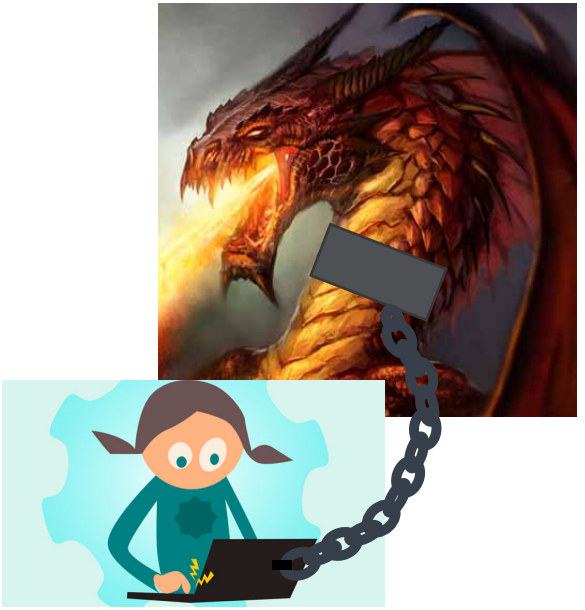
Invisible

Visible

Technical debt - "not quite right software which we postpone making it right"

WHAT TO DO?

How?

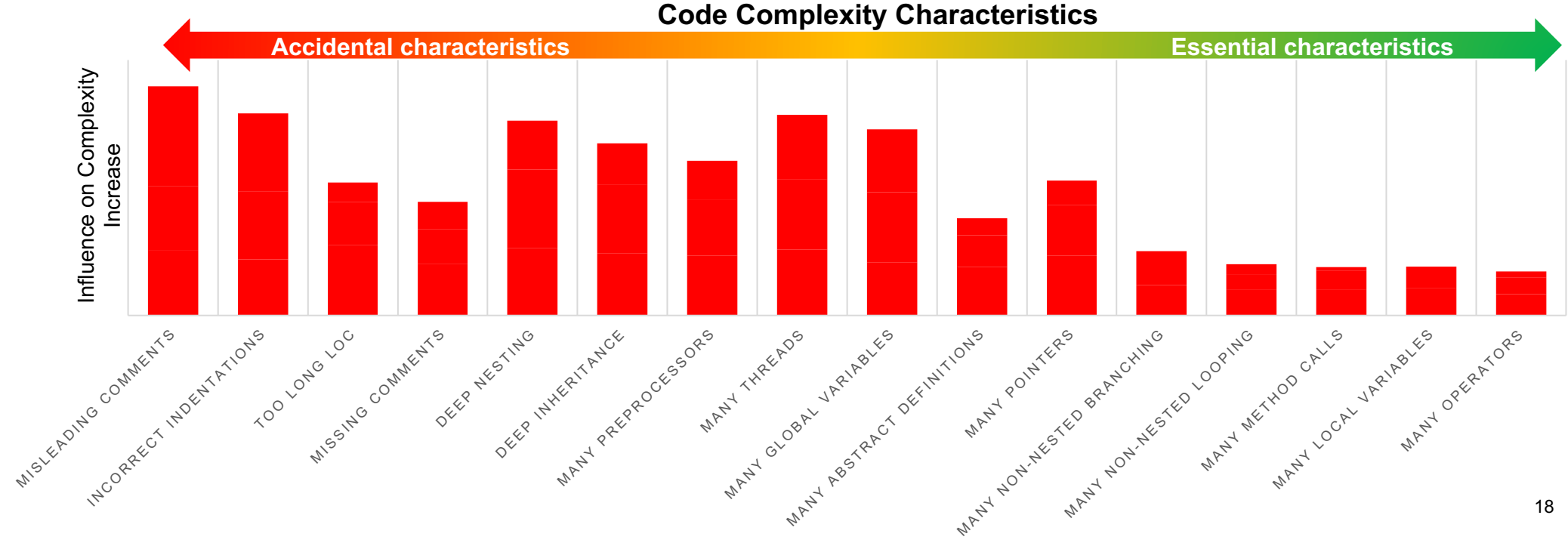


Today:
Significant Technical Debt!

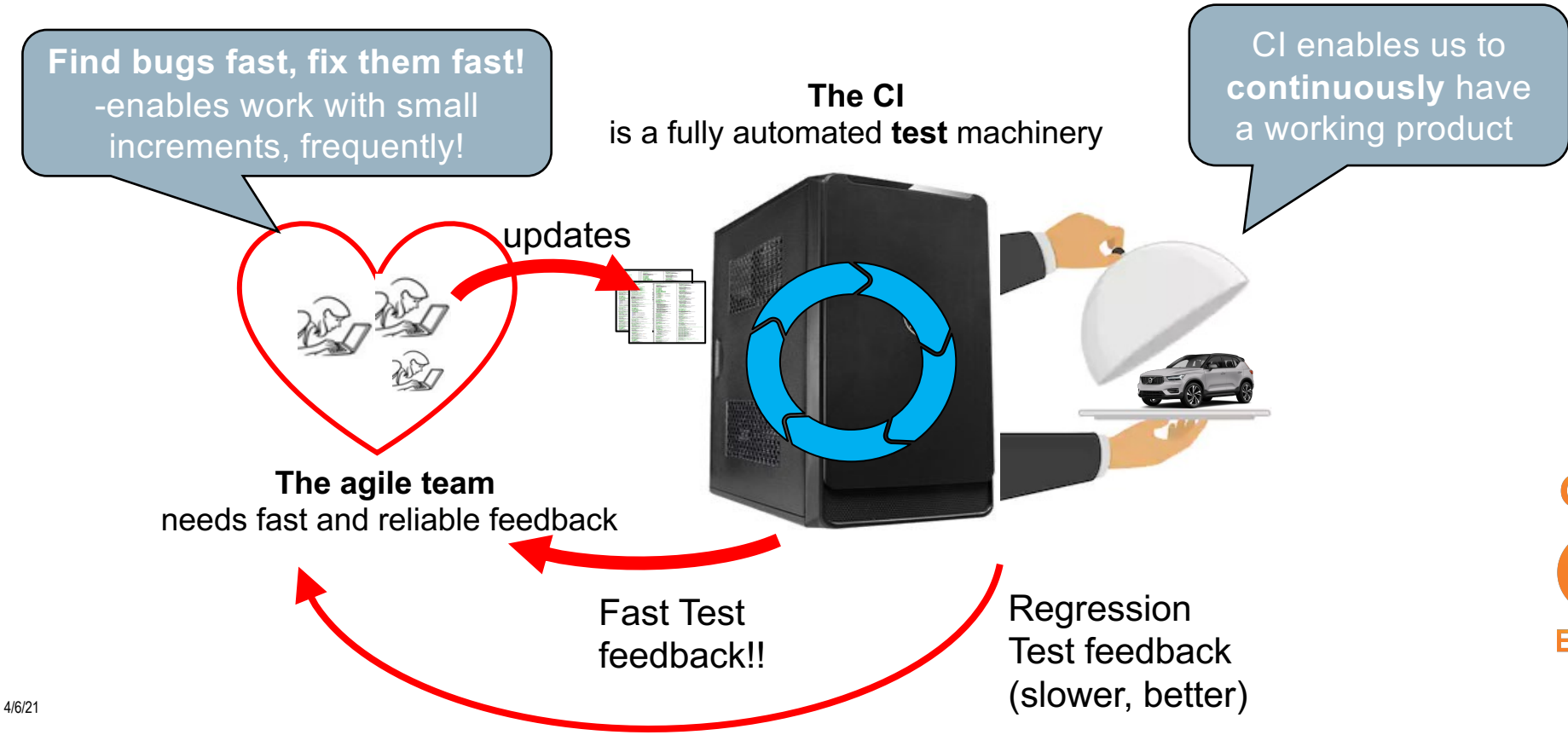
Tomorrow:
Insignificant Technical Debt!

ENHANCE CODE CRAFTSMANSHIP

- Learn to write simple, modular, and readable code
- Maintain full traceability to reduce recall time



EMPLOY CONTINUOUS INTEGRATION AND FEEDBACK



ONE CI EFFORT

REMOVE UNWORTHY VARIANTS

Vehicle Current Variants - Example

Vehicle variant	Engine variant	Emission variant	Gearbox variant
XC90	HP	Europe	Forward wheel drive
XC60	MP	China	All wheel drive
V90		Russia	
		USA	
		South America	

Every variant is a complete car

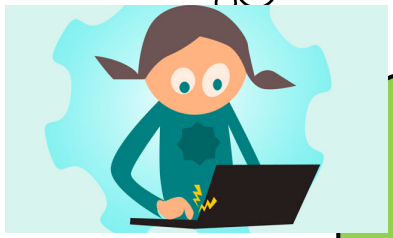
Every variant needs to be crafted continuously to perfection!

Increasing variants linearly increases complexity exponentially

DEVELO DECOUPLED MODULES AND INTERFACES

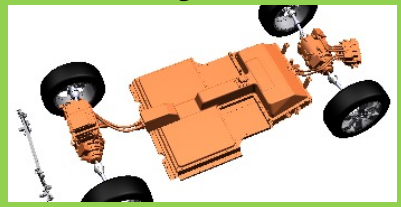
Enable:

- As **independent development** as possible
- Minimal **accidental complexity**
- Understandable **software architecture**



software | software | software

Battery Module



Hardware & sensors

Engine control module

Charging module

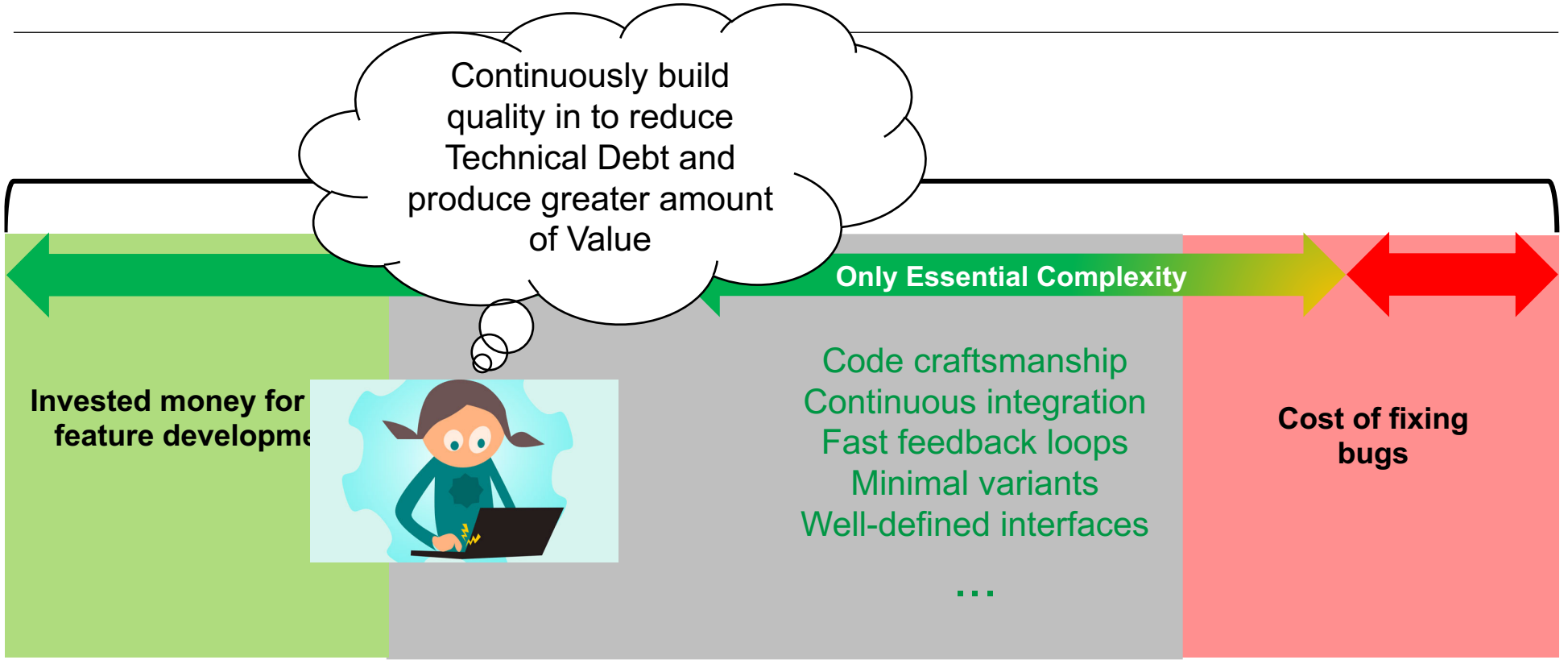
Used services

Electric Drive Module

Autonomous Drive Module

Modules using this service

WHAT DO WE NEED? - MANAGE ACCIDENTAL COMPLEXITY



What generates technical debt? - Not quite right software which we postpone making right

REFLECTIONS

- Discuss with your neighbor
  - Have you developed software project that had significant technical debt?
  - What actions could you take?

RECOMMENDED READING

MISRA 2012 clean coding guidelines
https://www.academia.edu/40301277/MISRA_C_2_012_Guidelines_for_the_use_of_the_C_language_in_critical_systems

Future Automotive Architecture and the Impact of IT Trends
https://assets.vector.com/cms/content/consulting/publications/BMW_Future_Automotive_Architectures.pdf
https://ieeexplore.ieee.org/abstract/document/7927914

Evaluating Essential and Accidental Code Complexity Triggers by Practitioners' Perception
https://www.researchgate.net/publication/339377604_Evaluating_Essential_and_Accidental_Code_Complexity_Triggers_by_Practitioners'_Perception
https://ieeexplore.ieee.org/abstract/document/9007382

Books:
Clean Code, Robert C. Martin