MES
MODEL ENGINEERING SOLUTIONS

COMPLEXITY MEASUREMENT AND REDUCTION

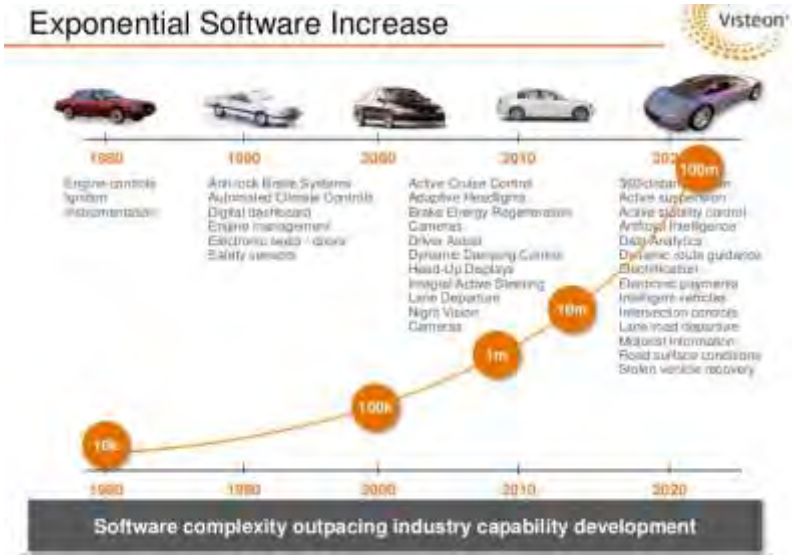
Heiko Doerr, Ferry Bachmann
MGI Group, January 23, 2018

SOFTWARE QUALITY.
MADE IN GERMANY.

SOLUTIONS FOR INTEGRATED QUALITY ASSURANCE
OF EMBEDDED SOFTWARE

SET THE SCENE

- Increase in software "size" (e.g. lines of code, parameters …)
- **Intuitively,** "complexity" also **increases**



Albert Faraj, "Driving Automotive R&D", Visteon Corporation, XIV Congreso Internacional de la Industria Automotriz en México on April 12, 2016.

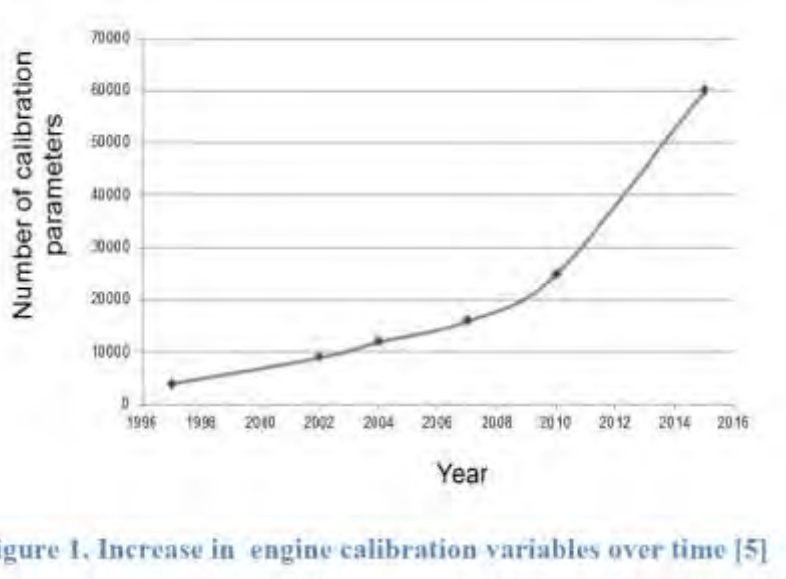


Figure 1. Increase in engine calibration variables over time [5]

Venkitachalam et al., "Metric-based evaluation of software architecture for an engine management system", SAE Technical Paper, 2016.

ISO 26262 RECOMMENDATIONS (1)

- Part 6: "Product development at the software level"

Table 1 — Topics to be covered by modelling and coding guidelines

	Topics	ASIL			
		A	B	C	D
1a	Enforcement of low complexity[a]	++	++	++	++
1b	Use of language subsets[b]	++	++	++	++
1c	Enforcement of strong typing[c]	++	++	++	++
1d	Use of defensive implementation techniques	o	+	++	++
1e	Use of established design principles	+	+	+	++
1f	Use of unambiguous graphical representation	+	++	++	++
1g	Use of style guides	+	++	++	++
1h	Use of naming conventions	++	++	++	++

- "++" is highly recommended

ISO RECOMMENDATIONS (2)

- Part 6-7: "Software Architectural Design"
  - "In order to avoid failures resulting from high complexity … by use of the principles"

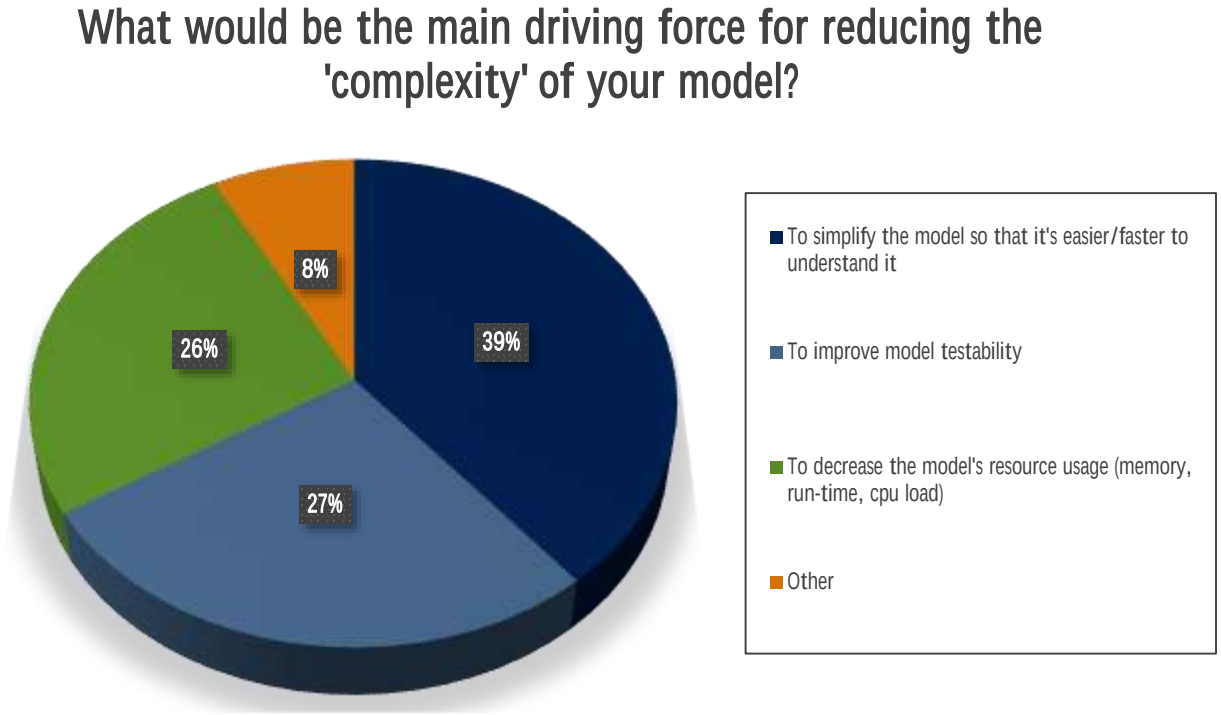
Table 3 — Principles for software architectural design

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components[a]	++	++	++	++
1c	Restricted size of interfaces[a]	+	+	+	+
1d	High cohesion within each software component[b]	+	++	++	++
1e	Restricted coupling between software components[a, b, c]	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts[a, d]	+	+	+	++

COMPLEXITY AMBIGUOUS

- Simplicity/Understandability?
- Testability?
- Resource usage (memory, run-time, cpu load)?

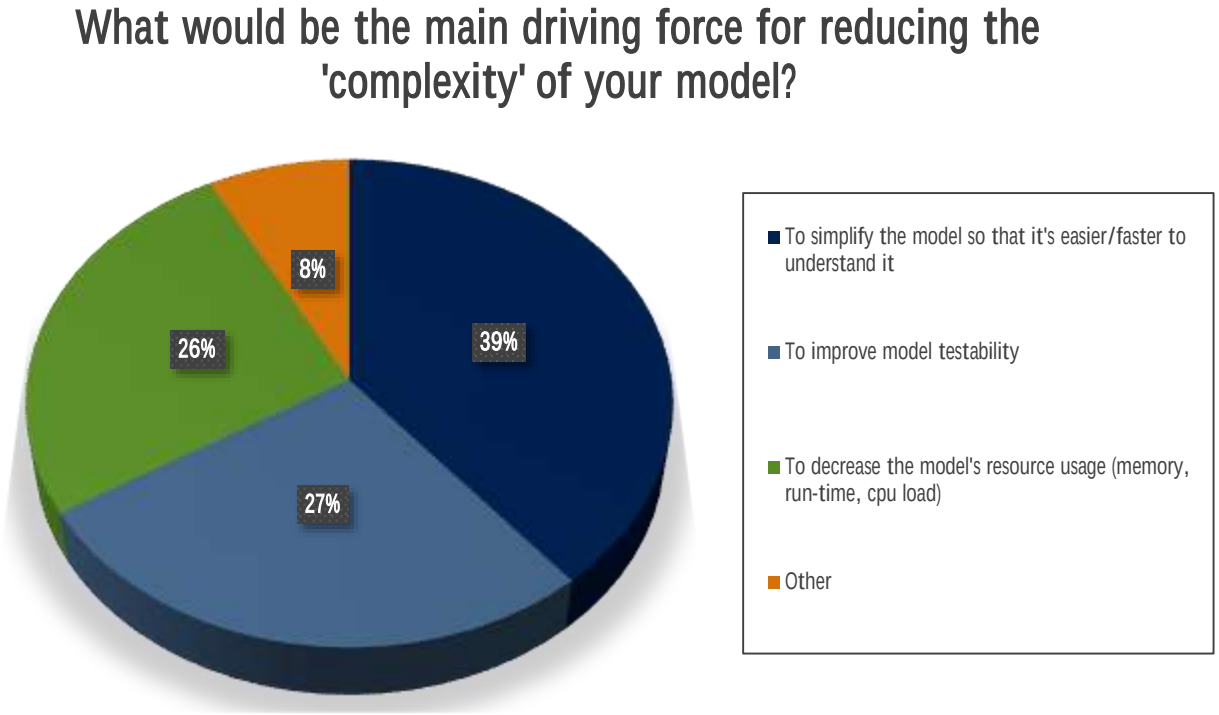
COMPLEXITY AMBIGUOUS



- 76 participants
- Only 5 voted 'Other', where 2 noted 'All three aspects combined'
- => Only 4% have different driving force
  - 'Simplify so that formal methods can be applied'
  - 'Measure and reduce complexity to satisfy safety standards'

COMPLEXITY AMBIGUOUS

- Simplicity/Understandability
- Testability
- Resource usage (memory, run-time, cpu load)



SIMPLICITY/UNDERSTANDABILITY

Table 3 — Principles for software architectural design

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Hierarchical structure of software components	++	++	++	++

- MAAB db_0144

5.2.2. db_0144: Use of Subsystems

ID: Title	db_0144: Use of Subsystems
Priority	strongly recommended
Scope	MAAB
MATLAB Version	All
Prerequisites	
	Blocks in a Simulink diagram should be grouped together into subsystems based on functional decomposition of the algorithm, or portion thereof, represented in the diagram.

SIMPLICITY/UNDERSTANDABILITY

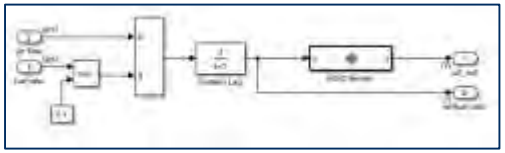
- How to detect missing use of subsystems?
- Best practice: Metrics

5.2.2. db_0144: Use of Subsystems

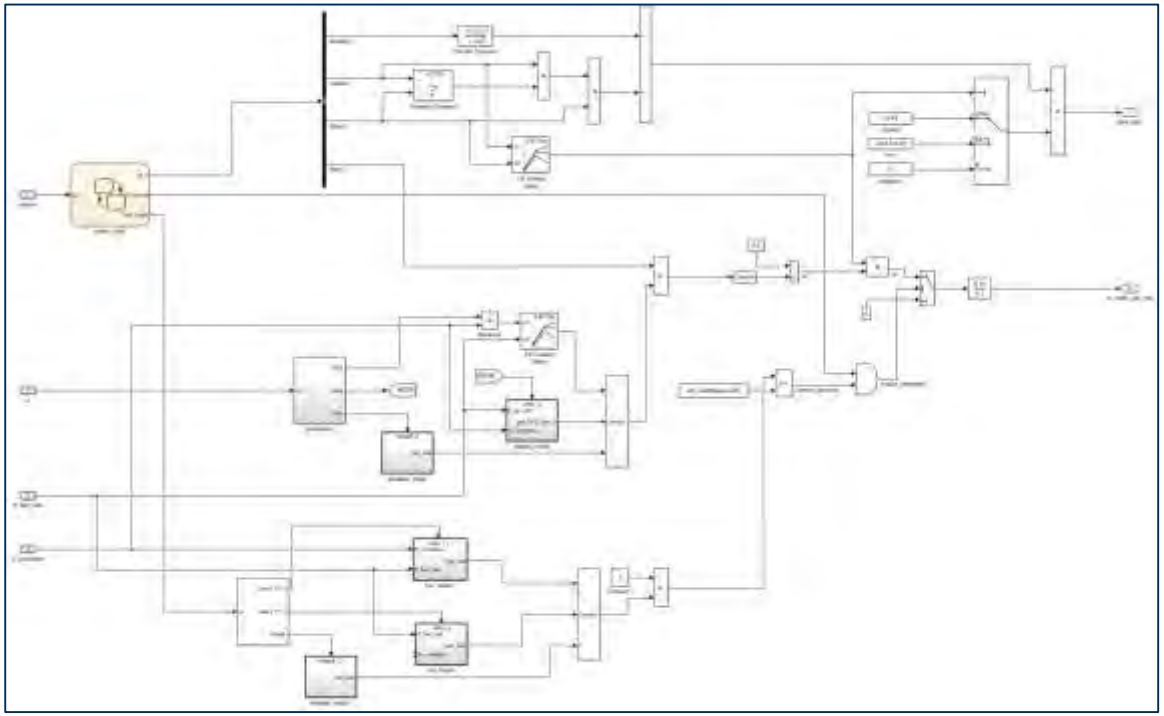
ID: Title	**db_0144: Use of Subsystems**
Priority	strongly recommended
Scope	MAAB
MATLAB Version	All
Prerequisites	
	Blocks in a Simulink diagram should be grouped together into subsystems based on functional decomposition of the algorithm, or portion thereof, represented in the diagram.

- Measure **Complexity** on subsystem scope (without sub-subsystems)!

Low complexity



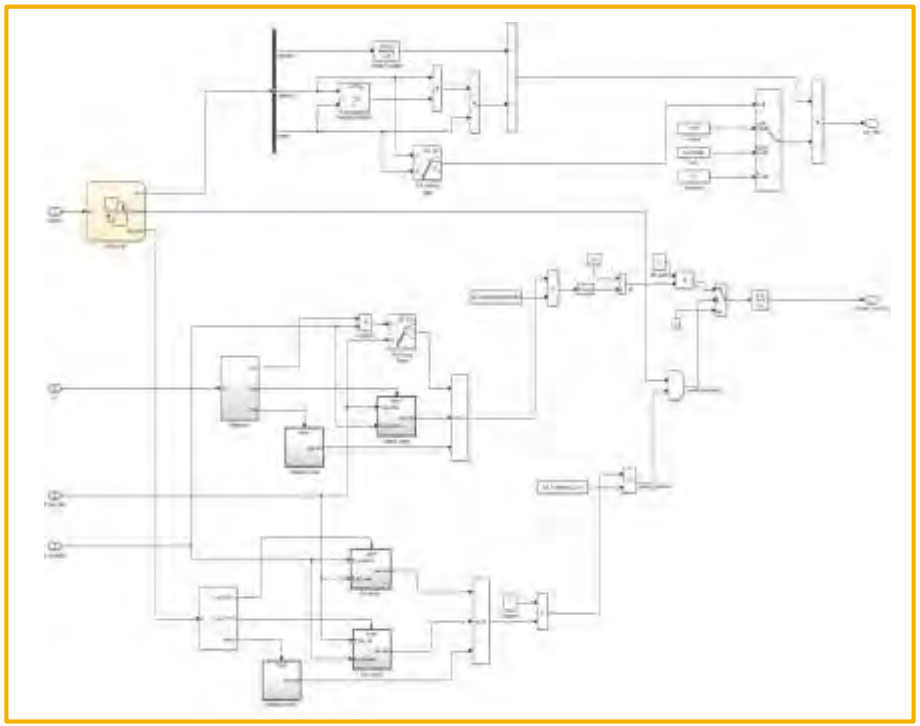
High complexity



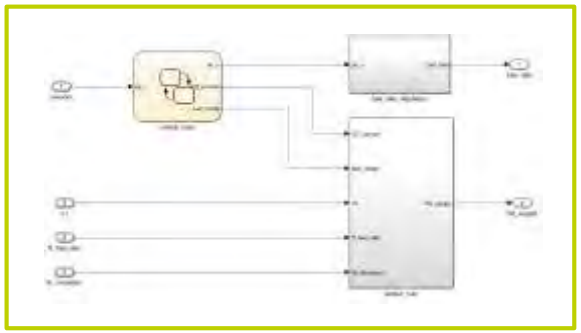
LIMIT COMPLEXITY => ENFORCE DECOMPOSITION

MES
MODEL ENGINEERING SOLUTIONS

High complexity



Low complexity



Simplicity

- Easier to understand for developer -> reduced probability of bugs
- Easier and faster to review and maintain (e.g. navigation speed: time to locate functionality or bug)

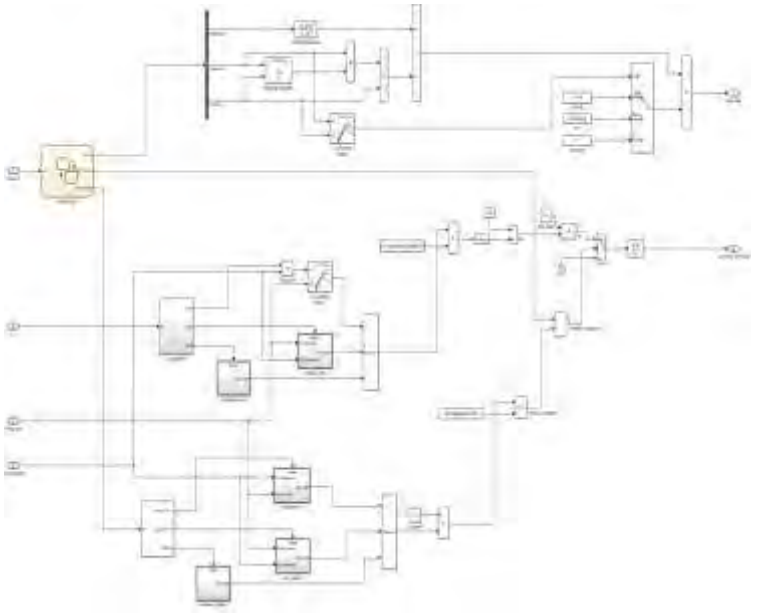
COMPLEXITY METRICS ON SUBSYSTEM SCOPE

- Number of Blocks (e.g. Model Adviser®, MES M-XRAY®)
- Halstead Complexity (MES M-XRAY®) < 750

Which measures do you use to assess simplicity of a subsystem?

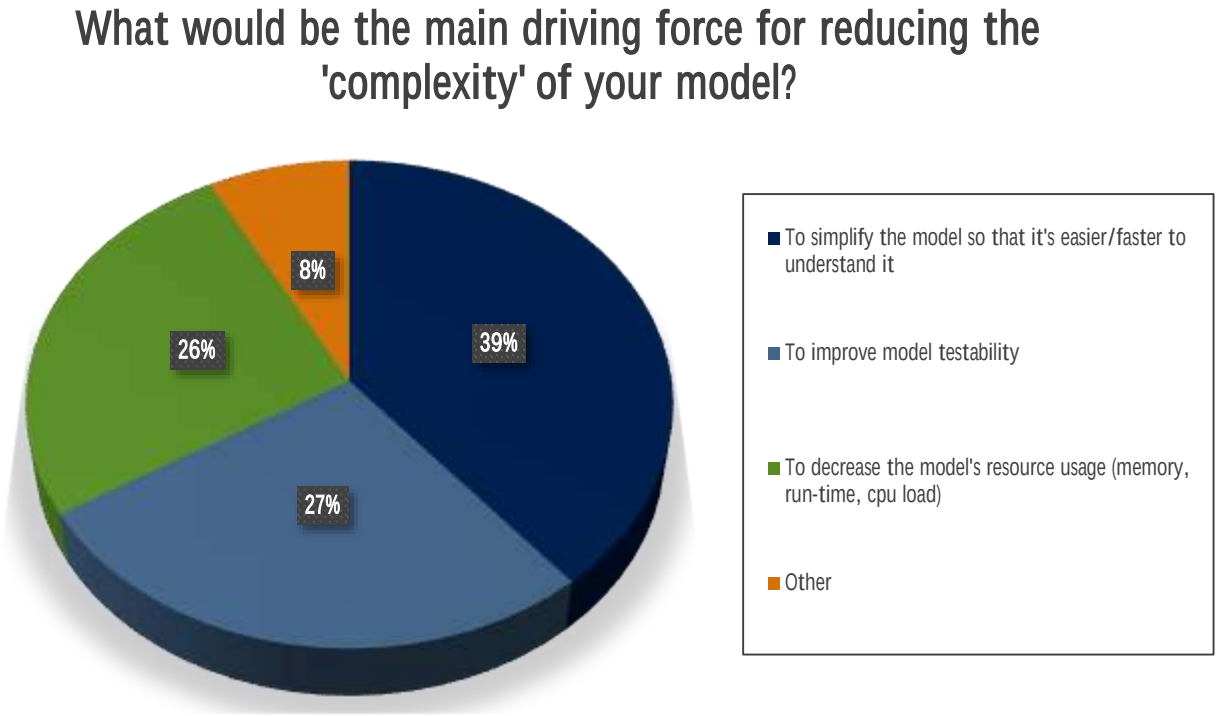
Which complexity metrics do you use for objective and well-accepted assessment for simplicity?

How do you derive upper bounds for metrics?



COMPLEXITY AMBIGUOUS

- Simplicity/Understandability
- Testability
- Resource usage (memory, run-time, cpu load)



TESTABILITY

- Principles related to testability

Table 3 — Principles for software architectural design

Methods		ASIL			
		A	B	C	D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components[a]	++	++	++	++
1c	Restricted size of interfaces[a]	+	+	+	+
1d	High cohesion within each software component[b]	+	++	++	++
1e	Restricted coupling between software components[a, b, c]	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts[a, d]	+	+	+	++

- 'Restricted size of interfaces' already discussed in MGI Group Meeting 03/2017

TESTABILITY

Table 3 — Principles for software architectural design

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components[a]	++	++	++	++
1c	Restricted size of interfaces[a]	+	+	+	+

5.2.2. db_0144: Use of Subsystems

ID: Title	db_0144: Use of Subsystems
Priority	strongly recommended
Scope	MAAB
MATLAB Version	All
Prerequisites	
	Blocks in a Simulink diagram should be grouped together into subsystems based on functional decomposition of the algorithm, or portion thereof, represented in the diagram.

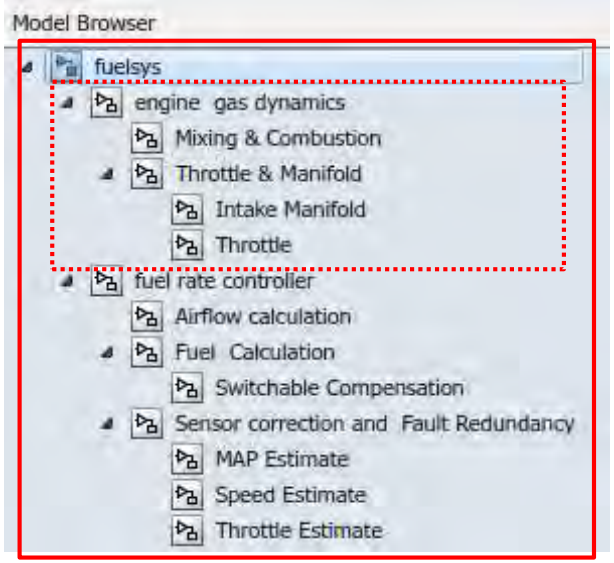
- Good functional decomposition => Better testability
- Metrics/Guidelines for complexity on subsystem scope

TESTABILITY

Table 3 — Principles for software architectural design

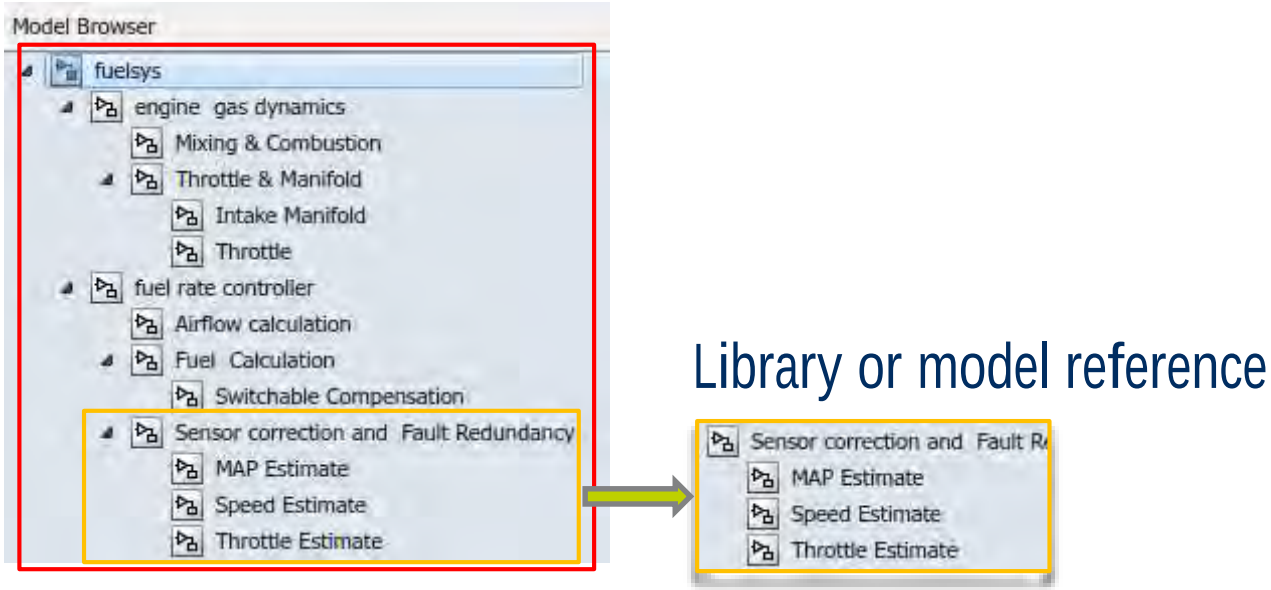
	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components[a]	++	++	++	++
1c	Restricted size of interfaces[a]	+	+	+	+

- How to limit size of software components in Simulink®?
- Limit overall size of model or subtree



LIMIT OVERALL SIZE/COMPLEXITY OF MODEL

=> Enforce usage of libraries + model references



- Testability gain?
  - Focused requirements + tests
  - Flexible loading + compilation + simulation time
  - Reusability: Test only once

Modularity

LIMIT OVERALL SIZE/COMPLEXITY OF MODEL

- How to measure overall size/complexity of model?
  - File size (inaccurate)
  - Number of Blocks (e.g. Model Adviser®, MES M-XRAY®)
  - Halstead Complexity (MES M-XRAY®)
  - Cyclomatic Complexity (e.g. Model Adviser®, MES M-XRAY®)

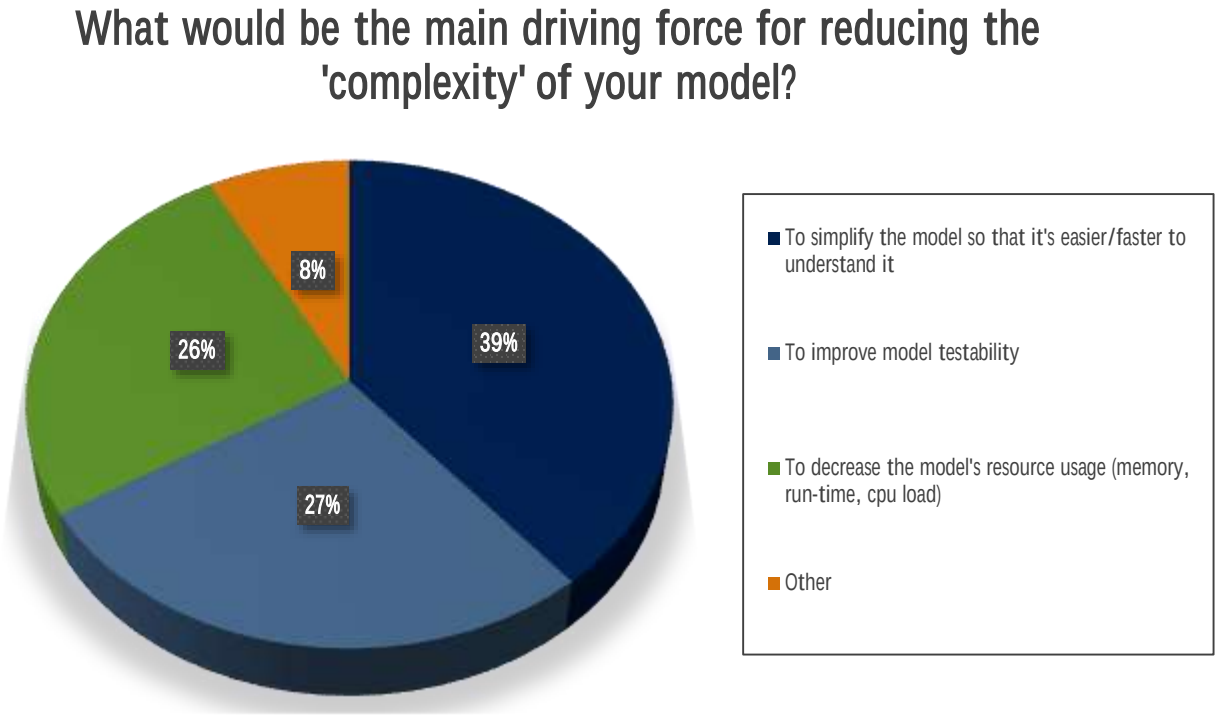
Which measures do you use to assess testability of a model/subsystem?

Which complexity metrics do you use for objective and well-accepted assessment for testability?

How do you derive upper bounds for metrics?

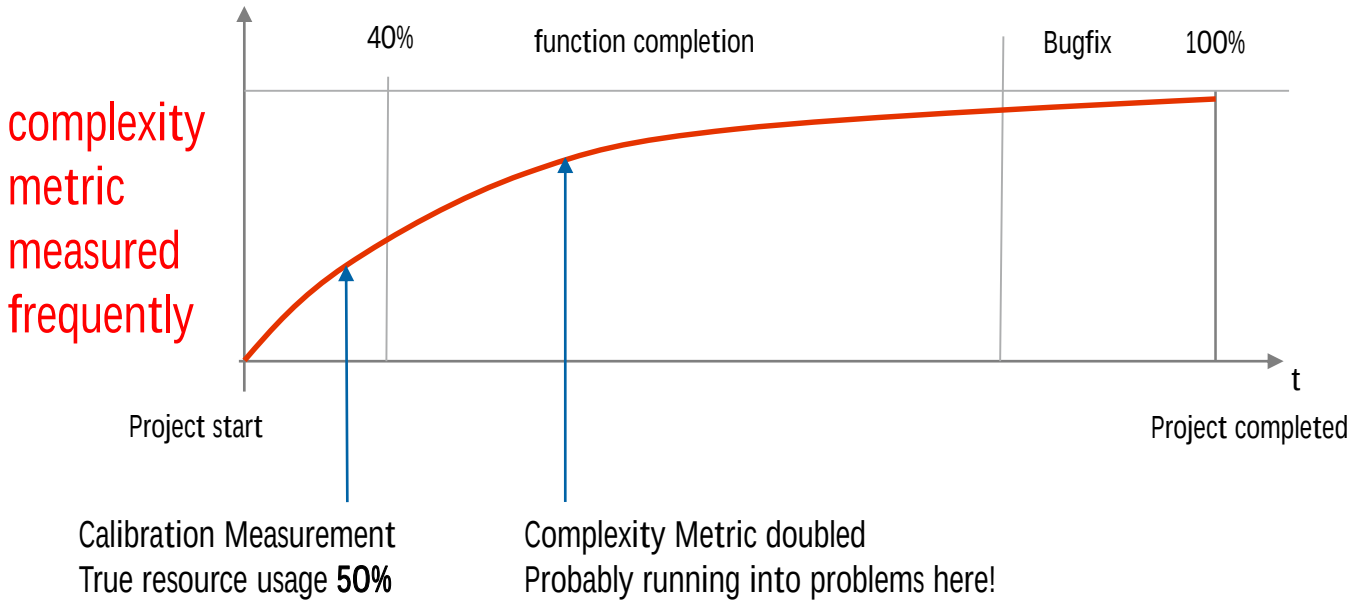
COMPLEXITY AMBIGUOUS

- Simplicity/Understandability
- Testability
- Resource usage (memory, run-time, cpu load)



RESOURCE USAGE

- Measuring true resource usage is best metric
- But: Measuring is sometimes costly or impossible in premature versions
  => Model metrics used to estimate resource usage



RESOURCE USAGE

- Estimation based on correlation between metrics and resource usage (run-time, binary size, memory usage)
- But: No systematic investigation of dependency yet!

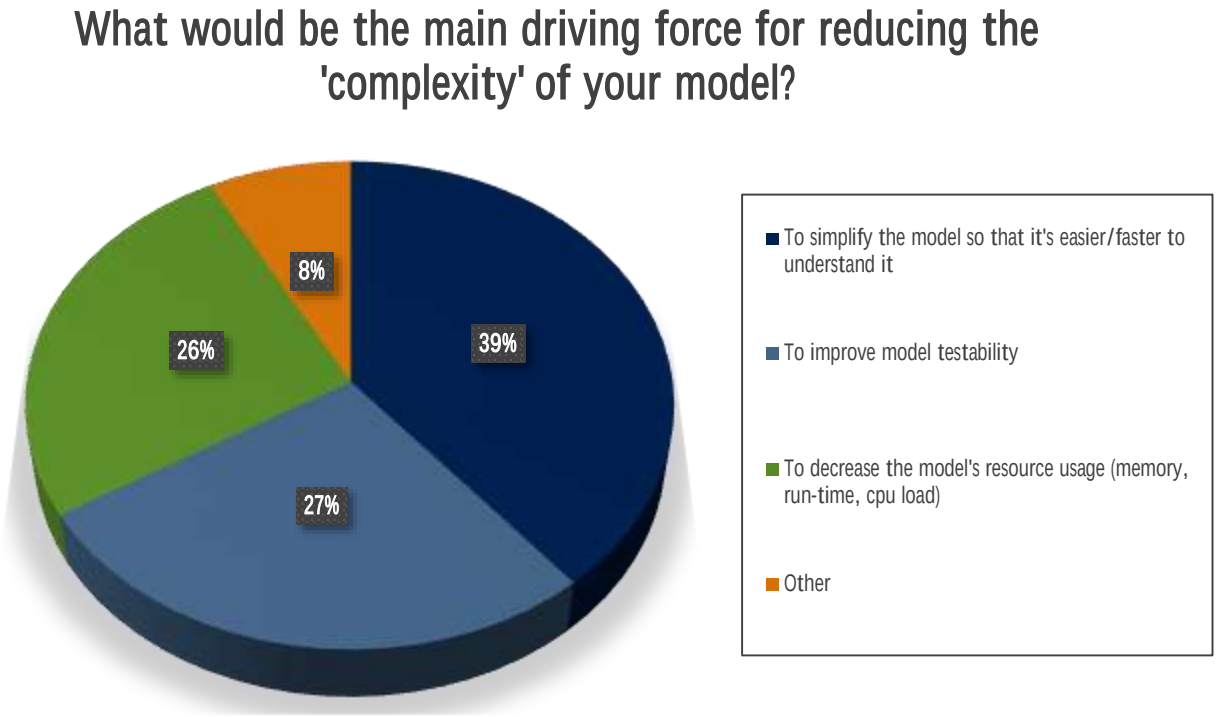
Which measures do you use to assess resource usage of software generated from a model/subsystem?

Which complexity metrics do you use for objective and well-accepted assessment for resource usage?

How do you derive upper bounds for metrics?

Have you identified a correlation between metrics on software complexity and resource consumption?

If so, which resources are highly correlated?



Which further properties which can be subsumed under the notion of model complexity should be considered?

MGIGROUP TIMELINE

MES USER CONFERENCE 2018

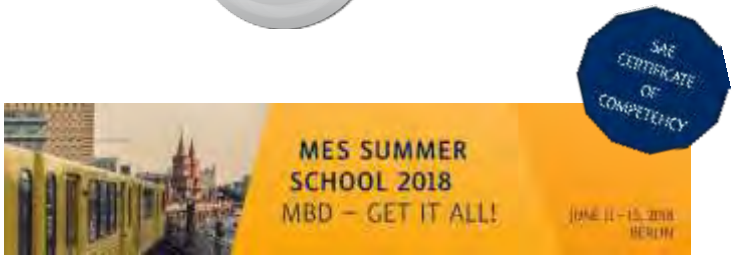
- Venue: Umspannwerk Ost, Palisadenstraße 48, 10243 Berlin
- Registration fee: € 220
- Limited number of participants, please register in advance by September 20, 2018

- **MES Summer School on Introduction to MBD**

May 29, 2018

- **Next online meeting via WebEx of the MGIGroup**

June 11-15, 2018



MES SUMMER SCHOOL 2018

- Venue: Michelberger Hotel, Warschauer Str. 39-40, 10243 Berlin
- Registration fee: € 2,950 incl. full board, hotel, and leisure program in Berlin
- Limited number of participants, please register in advance by May 29, 2018

October 11-12, 2018

- **Next MES User Conference to meet face-to-face, share experiences, discuss…**

MES
MODEL ENGINEERING SOLUTIONS

MODEL ENGINEERING SOLUTIONS GMBH

Waldenserstraße 2 - 4
10551 Berlin
Germany

T: +49 30 2091 6463-0
F: +49 30 2091 6463-33
info@model-engineers.com
www.model-engineers.com