MATLAB EXPO 2018

Model Quality Objectives to improve collaboration between manufacturers and suppliers

François Guérin

Model Quality Objective working group

- Idea emerged from SQO success few years ago

Software Quality Objectives for Source Code

A. *Patrick BRIAND*[5], B. *Martin BROCHET*[4], C. *Thierry CAMBOIS*[2], D. *Emmanuel COUTENCEAU*[5] E. *Olivier GUETTA*[3], F. *Daniel MAINBERTE*[2], G. *Frederic MONDOT*[3], H. *Patrick MUNIER*[4], I. *Loic NOURY*[4], J. *Philippe SPOZIO*[2], K. *Frederic RETAILLEAU*[1]

1. **Delphi Diesel System** France s.a.s, 9 bd de l'Industrie 41042 BLOIS France
2. **PSA Peugeot Citroën**, 75 avenue de la Grande-Armée, BP01, 75761 PARIS
3. **Renault s.a.s**, 13/15 Quai Alphonse Le Gallo, 92100 BOULOGNE-BILLANCOURT
4. **The MathWorks**, 2 rue de Paris 92196 MEUDON France
5. **Valeo**, 43 Rue Bayen, PARIS 75017

ERTS 2010 conference

- New working group focused on model (MQO)
  - extended to Robert Bosch
  - consensus-based decisions, ~100h of meetings over 2,5 years, many document reviews

Model Quality Objectives for Embedded Software Development with MATLAB and Simulink

Jérôme Bouquet (Renault), Stéphane Faure (Valeo), Florent Fève (Valeo), Matthieu Foucault (PSA Peugeot Citroën), Ursula Garcia (Robert Bosch), François Guérin (MathWorks), Thierry Hubert (PSA Peugeot Citroën), Florian Levy (Renault), Stéphane Louvet (Robert Bosch), Patrick Munier (MathWorks), Pierre-Nicolas Paton (Delphi Technologies), Alain Spiewek (Delphi Technologies)

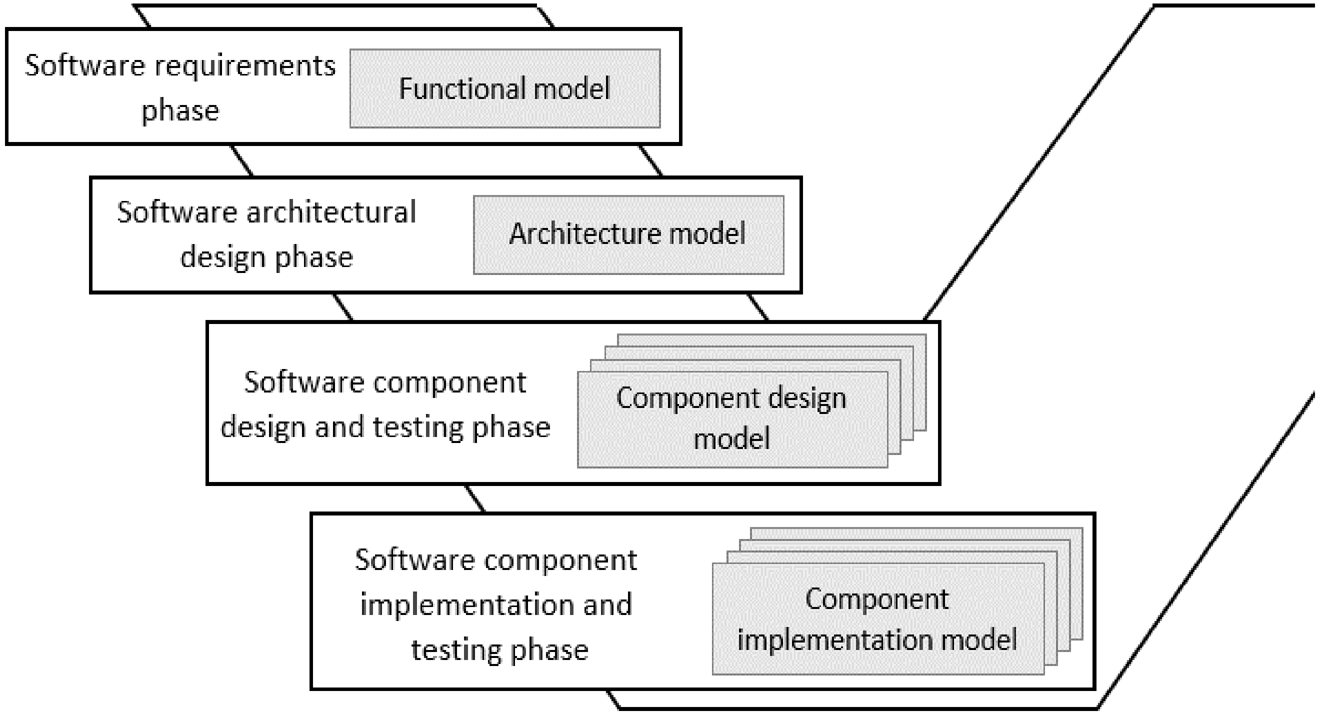
ERTS^2 2018 conference

Goals

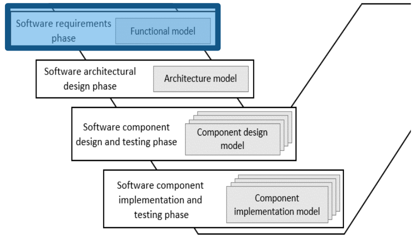
- Agree on a state-of-the art for model-based design in the context of software development.
- Establish common expectation on model quality when doing co-development between different parties.
- Help non-software developers to understand how they contribute to software development.
- Clarify impact of successive design stages with Simulink and how to transition from early prototyping to final design.

Model Quality Objectives – SW development process

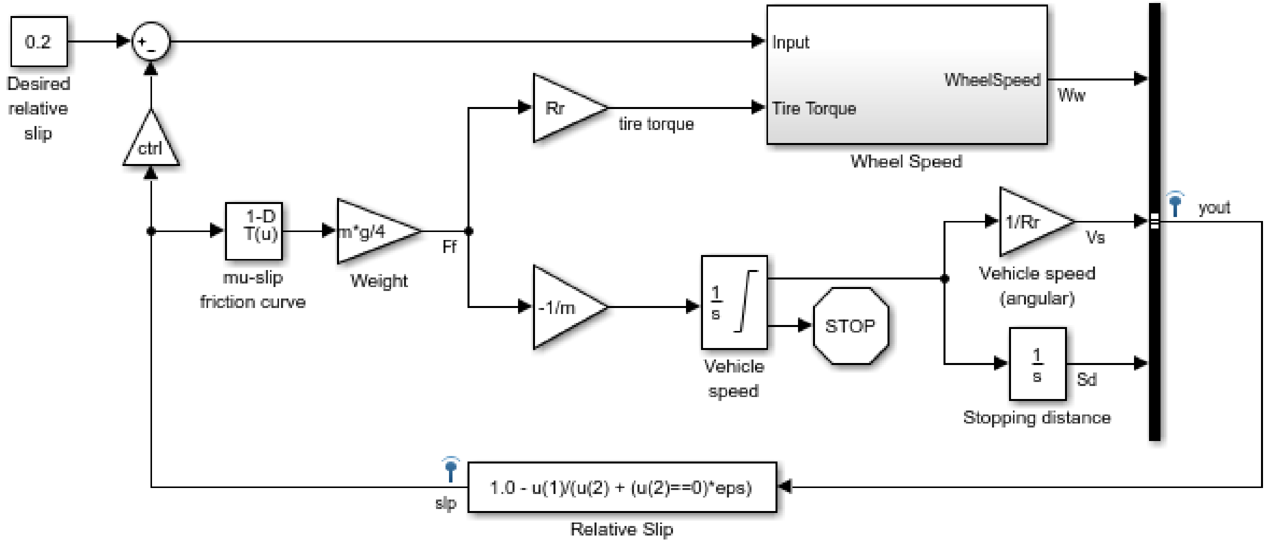
- A software development supported by <u>design models</u>



Model Quality Objectives – SW requirements phase



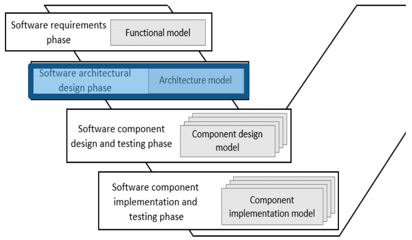
- **Functional model goals**
  - Accelerate the stabilization of the software functional requirements
  - Improve the quality of the software functional requirements



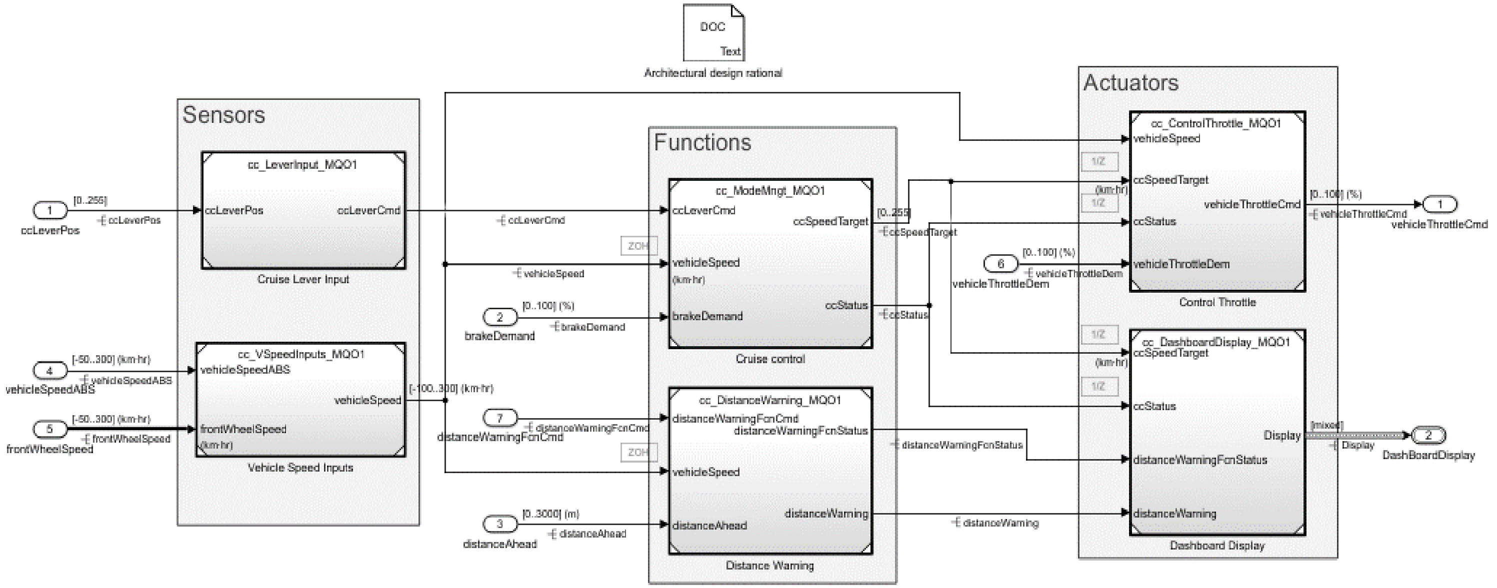
- No data typing
- Continuous or discrete
- Focus on complex requirements

Model does not replace the software functional requirements !

Model Quality Objectives – SW architectural design phase



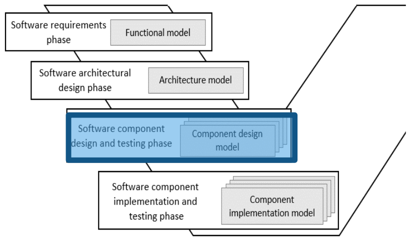
- **Architecture model goals**
  - Model and verify the component interfaces, connections and scheduling
  - Comply with modeling and architecture standard (e.g. AUTOSAR)



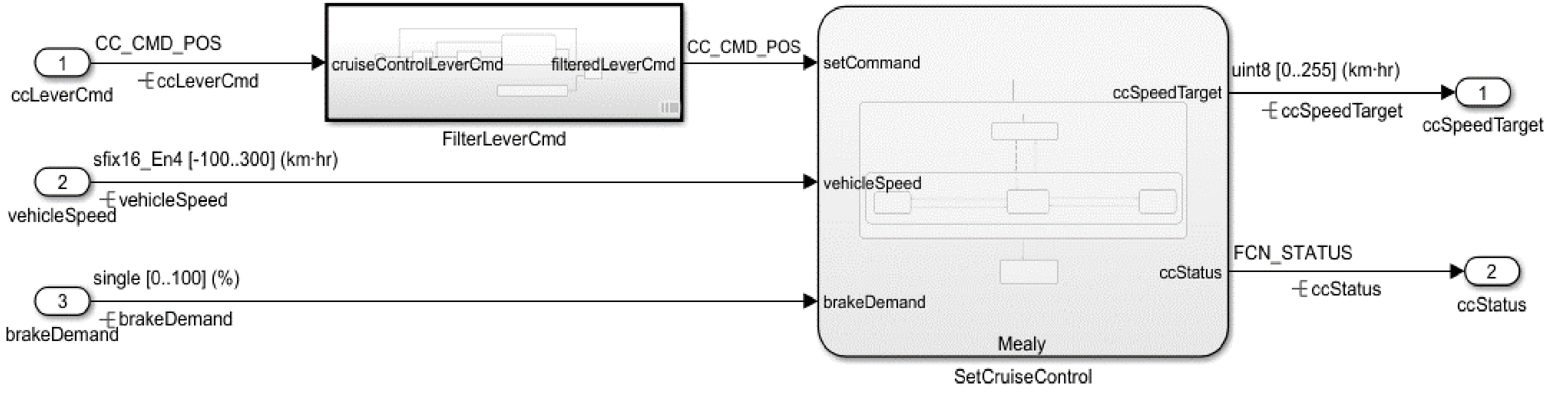
- Fully typed
- Discrete
- Assembly of model references

Need to be traced to requirements, commented, and linked with a data dictionary

Model Quality Objectives – SW component design phase



- **Component design model**
  - Model precisely the algorithms with a rich modelling language
  - Is functionally correct, robust and compliant with modelling standard



- Fully typed
- Discrete
- Mix of libraries and native blocks

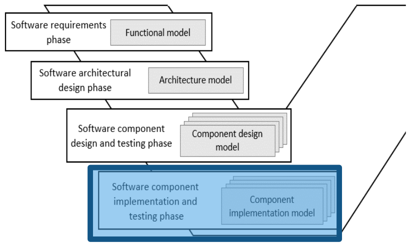
Need to be fully tested with 100% of requirement and design coverage

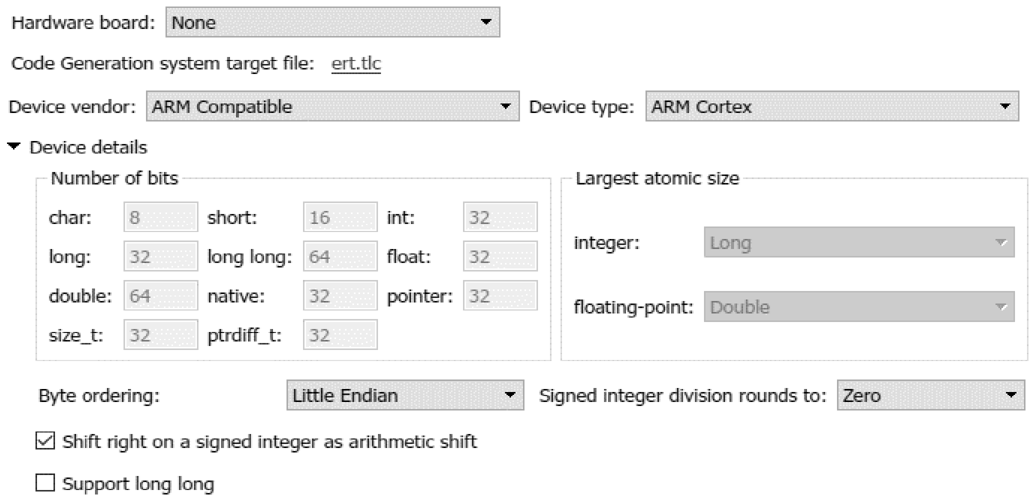
Model Quality Objectives – SW component implementation phase

- **Component implement model**

Generate production code

Generates code that is correct, robust, performant and compliant with coding standard

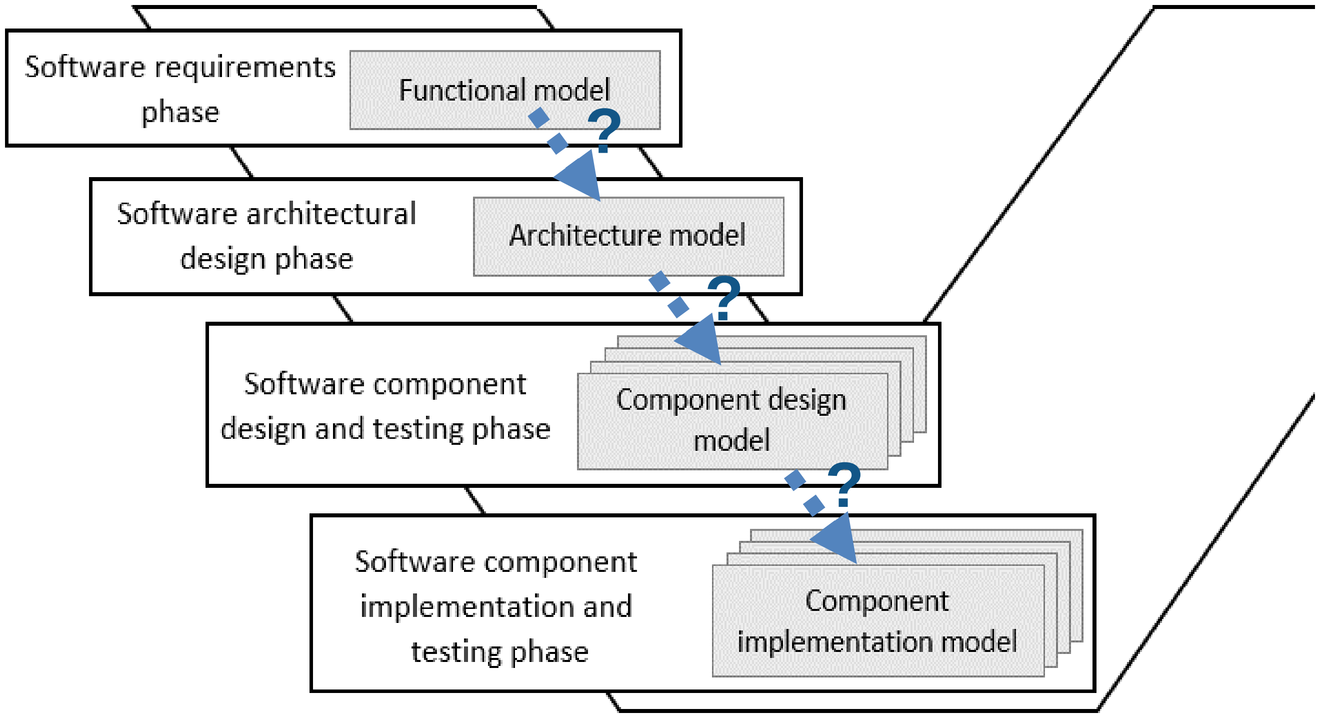




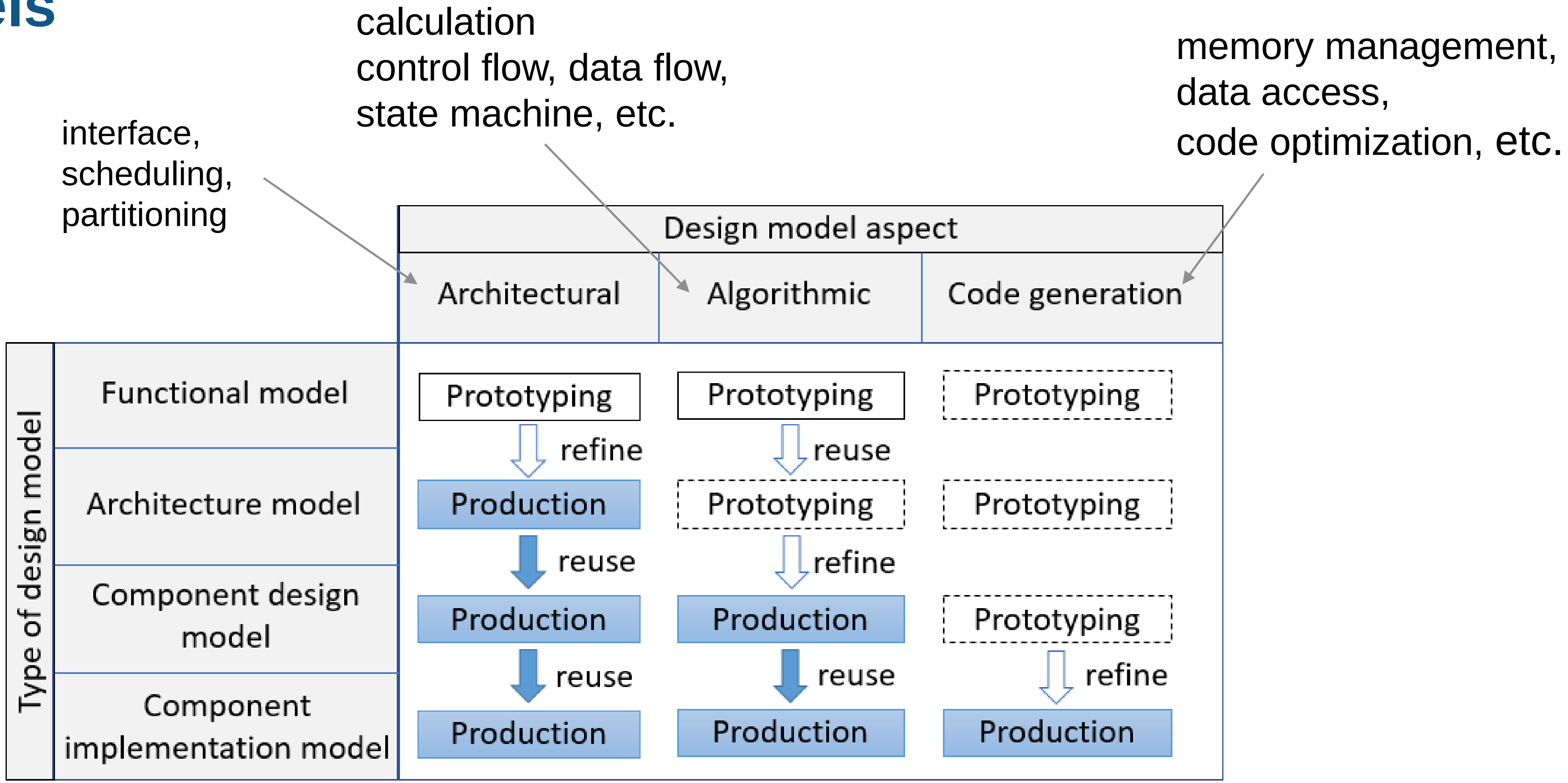
- Fully typed
- Mix of libraries, native blocks.
- Code generation configured for target hardware

Generated code needs to be fully tested with 100% of requirement and code coverage

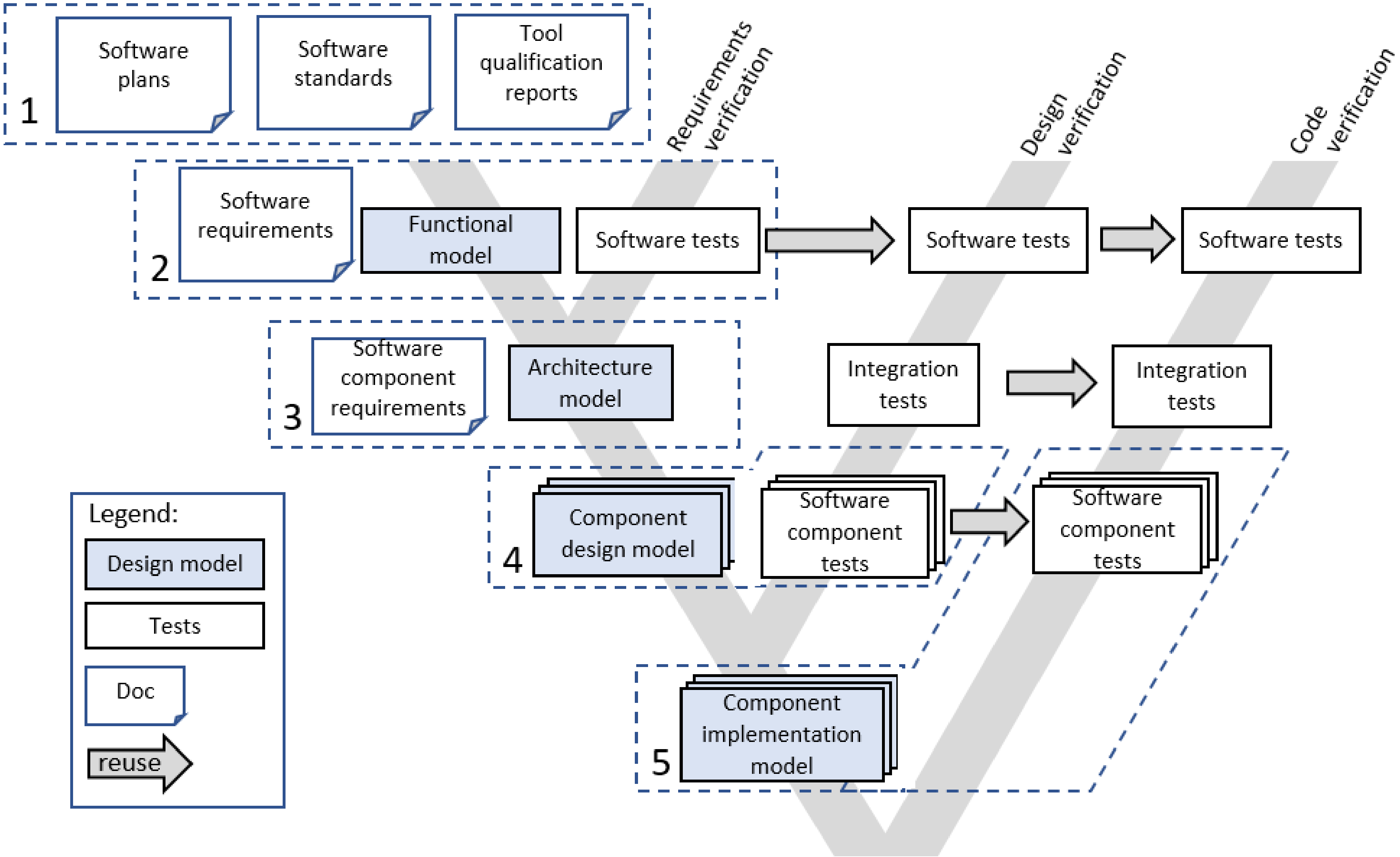
Model Quality Objectives – Relationship between design models



Model Quality Objectives – Relationship between design models



MQO/ Model-Based Design Workflow



Compatible with existing industry standards

- Complementary and compatible with existing standards
- Provide metrics and threshold to address quality requirements referred in standards
- Additional guidelines on planning phase to define responsibilities, and ensure workflow compatibility.

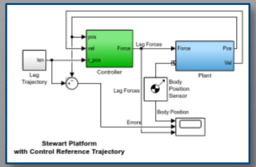
MBD/MQO	ISO26262	Automotive SPICE	DO-331	EN50128
1 Software planning	Initiation of product development at the software level	Initiation of product development at the software level	Software planning	Software planning
2 Software requirements	Specification of software safety requirements	Software requirements analysis	Software requirements	Software requirements
3 Software architectural design	Software architectural design	Software architectural design	Software design (architecture and Low level requirements)	Software architecture and design
4 Software component design and testing	Software unit design and implementation	Software detailed design and unit construction		Software component design
5 Software component implementation and testing	Software unit testing	Software unit verification	Software coding	Software component implementation and testing
Software integration	Software integration and testing	Software integration and integration test	Software integration	Software integration
Software testing	Verification of software safety requirements	Software qualification test	Software testing	Software validation

MQO clarifies exchanges and discussions

Functional model (MQO1)

Architecture model (MQO2)

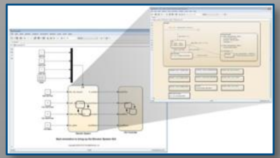
Customer



Exchanges / Discussions

MQO

Supplier



Component Design model (MQO3)

Component Implementation model (MQO4)

Model Quality Objectives / Requirements

Design model name	Quality Objective
Functional model	MQO-1
Architecture model	MQO-2
Component design model	MQO-3
Component implementation model	MQO-4

Set of requirements (MQO) to be able to assess the quality of each type of model

MQR ID	MQR Title	MQO-1	MQO-2	MQO-3	MQO-4
MQR-01	Model layout	M	M	M	M
MQR-02	Model comments	M	M	M	M
MQR-03	Model links to requirements	M	M	M	M
MQR-04	Model testing against requirements	M	R	M	M
MQR-05	Model compliance with modeling standard		M	M	M
MQR-06	Model data		M	M	M
MQR-07	Model size			M	M
MQR-08	Model complexity			M	M
MQR-09	Model coverage			M	M
MQR-10	Model robustness			M	M
MQR-11	Generated code testing against requirements			R	M
MQR-12	Generated code compliance with coding standard			R	M
MQR-13	Generated code coverage			R	M
MQR-14	Generated code robustness			R	M
MQR-15	Generated code execution time				M
MQR-16	Generated code memory footprint				M

M: Mandatory
R: Recommended

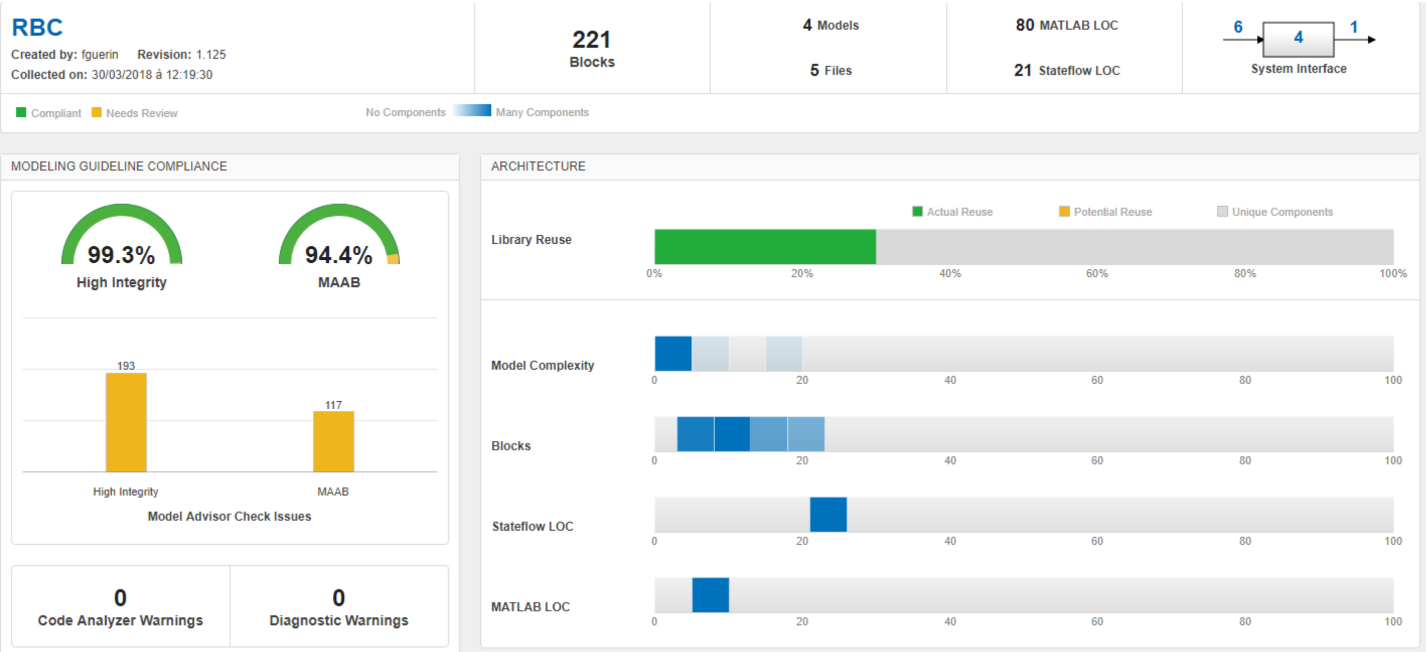
Example of Model Quality Requirement

MQR-08	Model complexity
Description	The model and its subsystems, Stateflow charts and MATLAB functions shall have a local cyclomatic complexity lower or equal to "30".
Recommendation level	MQO-1: – ; MQO-2: – ; MQO-3: **Mandatory** ; MQO-4: **Mandatory**
Notes	Local complexity is the cyclomatic complexity for objects at their hierarchical level. Aggregated cyclomatic complexity is the cyclomatic complexity of an object and its descendants. The threshold of 30 for local cyclomatic complexity is a recommendation and can be adapted on a project basis. The number 30 for Cyclomatic complexity has been derived from the HIS code metric (value of 10) and adapted to Model-Based Design.
References / Examples of techniques	Cyclomatic complexity is a measure of the structural complexity of a model. It approximates the McCabe complexity measure for code generated from the model. The McCabe complexity measure is slightly higher on the generated code due to error checks that the model coverage analysis does not consider. To compute the cyclomatic complexity of an object (such as a block, chart, or state), model coverage uses the following formula: $c = \sum_{1}^{N} (o_n - 1)$ N is the number of decision points that the object represents and o_n is the number of outcomes for the nth decision point. The tool adds 1 to the complexity number for atomic subsystems and Stateflow charts.
Rational	Cyclomatic complexity is a leading testability metric. Test harness can be created for simulation at model, subsystem, chart and MATLAB Function level.
Last update	1.0

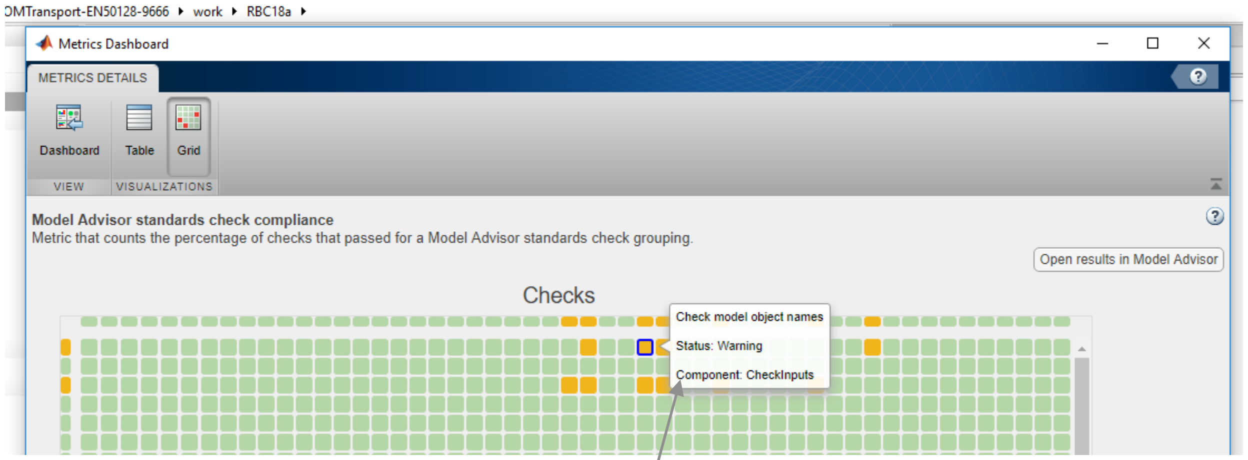
Recommendation level detail:

MQO-1	MQO-2	MQO-3	MQO-4
		Mandatory	**Mandatory**

MathWorks metric support



- All MQR can be measured with MathWorks tools
- Simulink Check provides additional metrics for size, architecture, compliance and readability
- Metric dashboard introduced in R2017b is rapidly involving to display and navigate from metrics results to models



navigate to model block