Model Quality Objectives for Embedded Software Development with MATLAB/Simulink

Stéphane Louvet (Robert Bosch), François Guérin (MathWorks), Florian Levy (Renault)

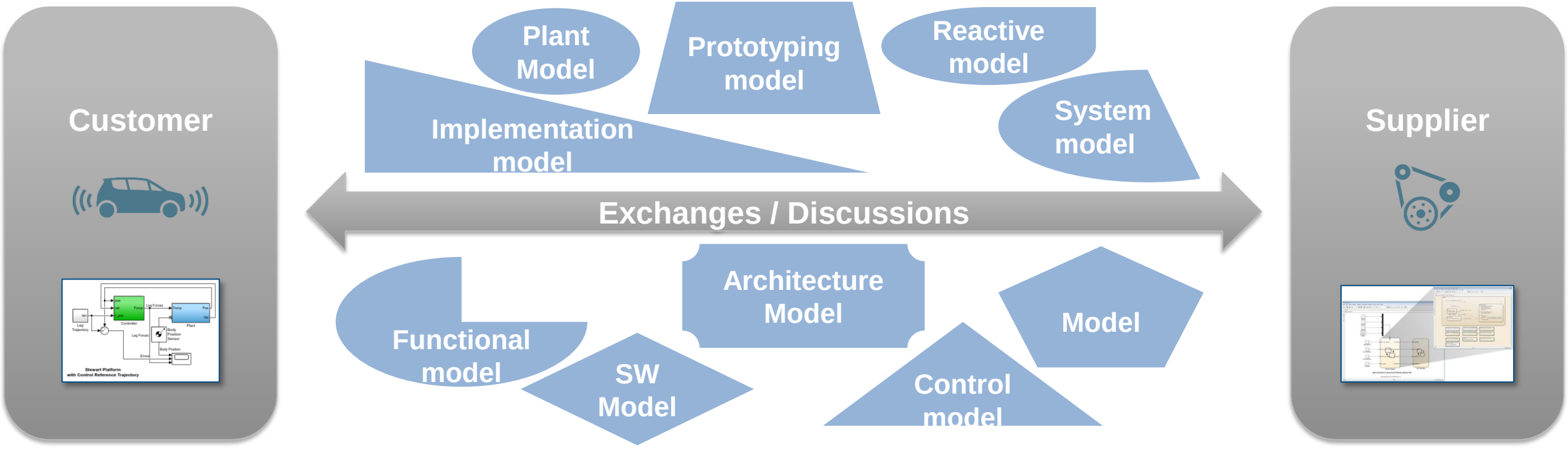
April 17th, 2018

Agenda

1. Background and Motivations
2. Introduction to Model Quality Objectives
3. Deployment and expected gains
4. Conclusion

Co-development in automotive

- System and Software co-development between teams and companies leads to increase “**Model Sharing**”. It should bring productivity gain…
- However difficulties are regularly faced.



Which kind of model is shared and what is its content ?

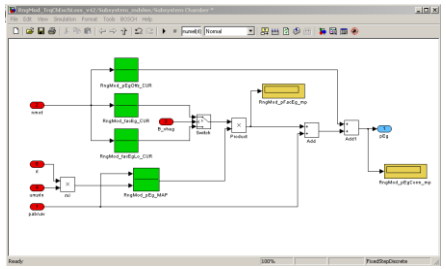
Model-Sharing to speed up software development

- Simulink models used as executable specification
  - Suppliers use Simulink models as base for code generation due to planning and cost pressure
  - Those Simulink models are initially not intended for code generation purpose

- Different interpretations of what is a good model for code generation

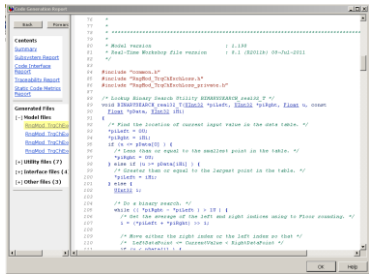
Management

≠



System Engineers

≠



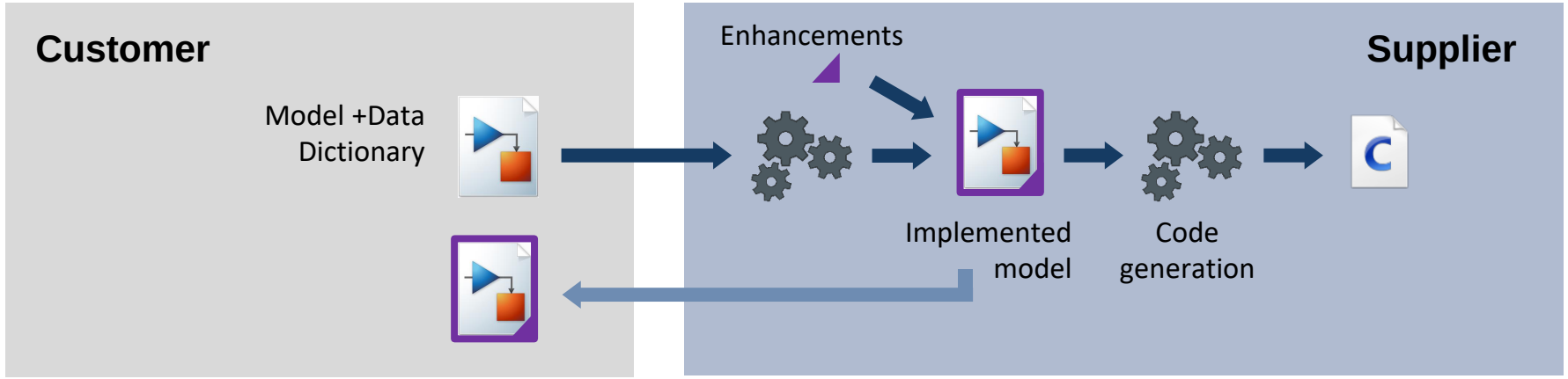
Software Engineers

Main issues faced for code generation with customer models

- Models cannot update !
- Usage of forbidden blocks
- Missing model data properties
- Too large models
- Hard to adapt for implementation
- No requirements nor test cases available for verification
- Not compliant with software standards (MDX or AUTOSAR)
- Not compliant with safety standard (ISO26262)

Model adaptation for code generation

- 1st solution : model exchange
  - The original models are enhanced / corrected manually by the supplier and delivered back to the customer



  - Drawbacks :
    - no easy exchange,
    - versioning issues, parallel development on customer and supplier side.

Model adaptation for code generation

- 2nd solution : automate some model corrections
  - The original models are automatically enhanced / corrected by the supplier with the help of scripts

customer model

Modified customer model

  - Drawbacks :
    - High investment in scripting + maintenance costs
    - 100% automation not achievable in practice.

Consequences of insufficient model quality

Developments are delayed

Lot of time spent in model issues reporting between customers and suppliers

Higher risk of bugs

Can lead to additional cost

Need to improve the customer / supplier cooperation with Model-Sharing :

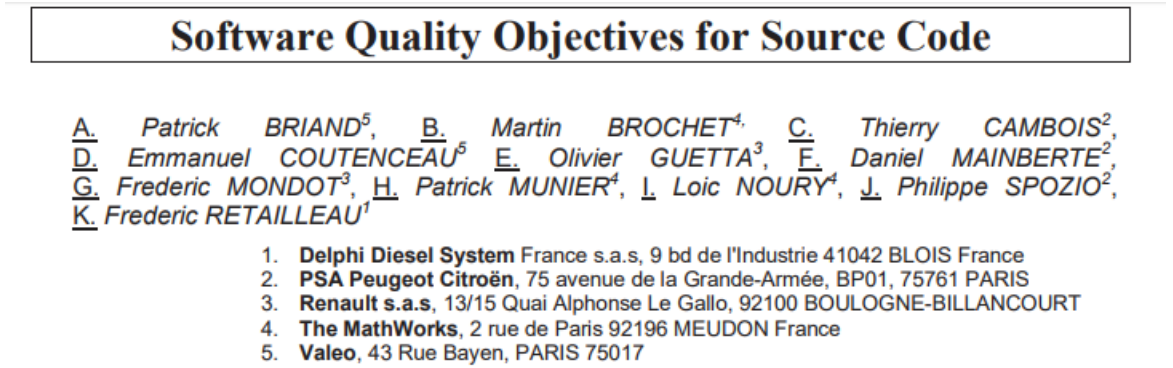
➔Model Quality Objectives (MQO)

Agenda

1. Background and Motivations
2. Introduction to Model Quality Objectives
3. Deployment and expected gains
4. Conclusion

Model Quality Objective working group

- Idea emerged from SQO success few years ago



Software Quality Objectives for Source Code

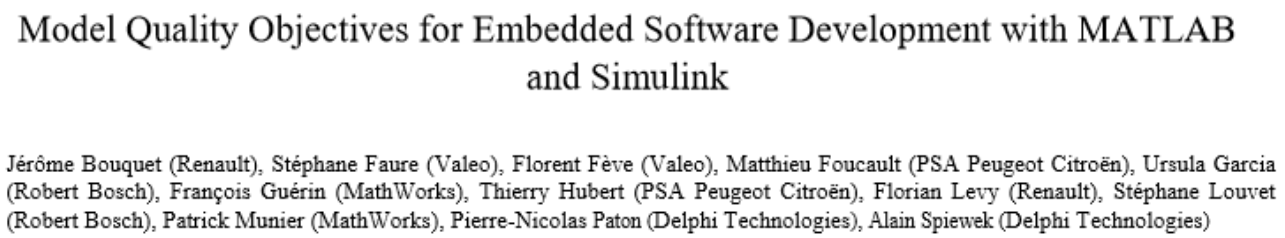
A. Patrick BRIAND[5], B. Martin BROCHET[4], C. Thierry CAMBOIS[2], D. Emmanuel COUTENCEAU[5] E. Olivier GUETTA[3], F. Daniel MAINBERTE[2], G. Frederic MONDOT[3], H. Patrick MUNIER[4], I. Loic NOURY[4], J. Philippe SPOZIO[2], K. Frederic RETAILLEAU[1]

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

→ https://www.mathworks.com/discovery/software-quality-objectives.html

- New working group focused on model (MQO)
  - extended to 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)

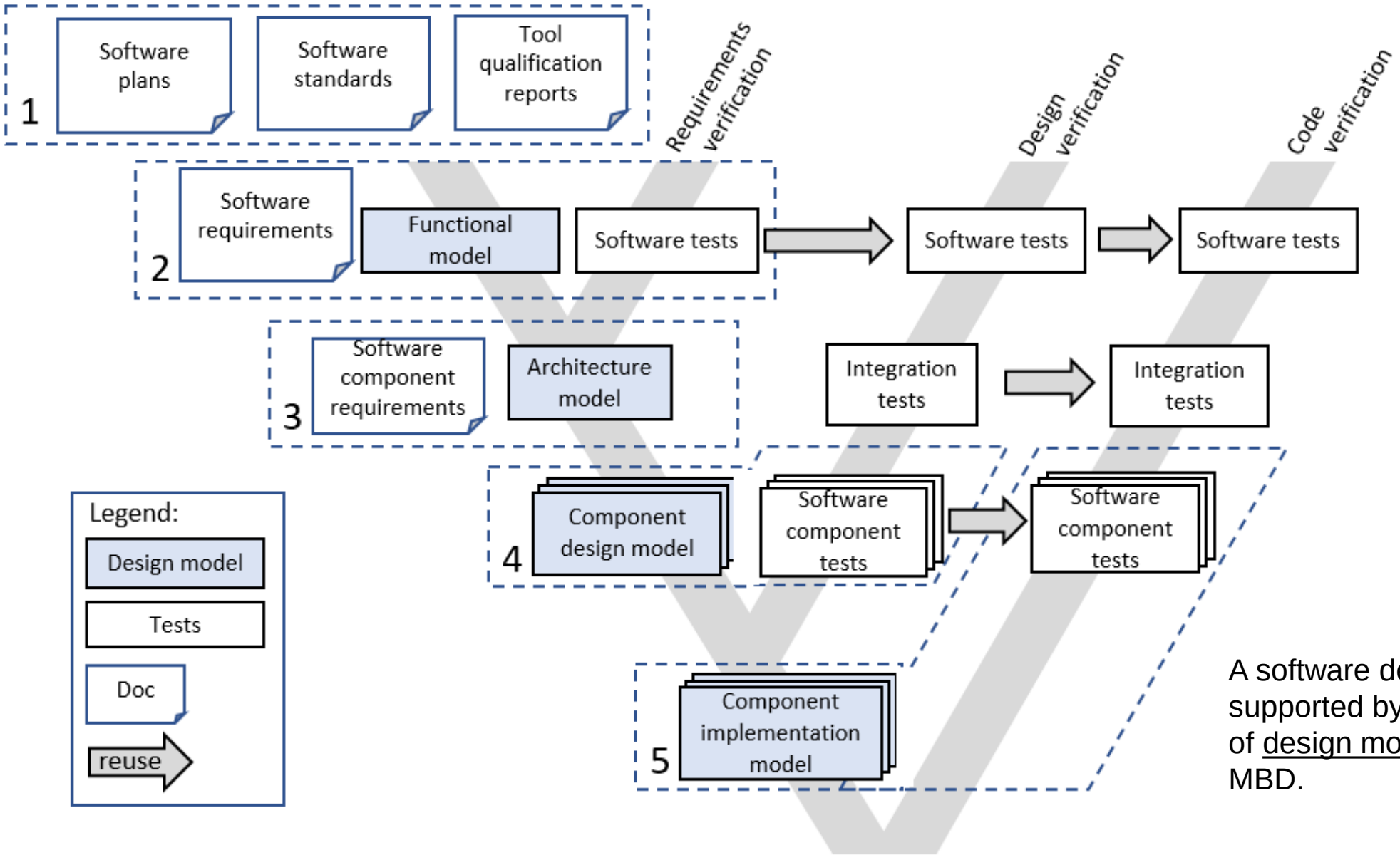
ERTS^2 2018 conference

→ Please contact MathWorks to access the paper

Goals

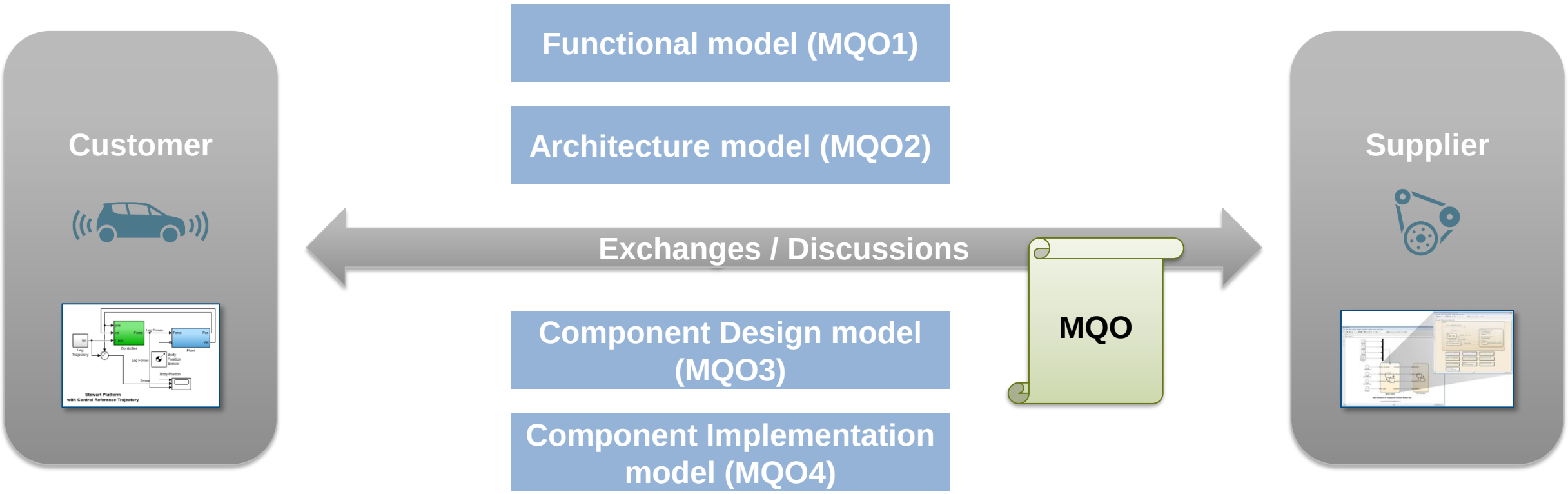
- Agree on a state-of-the art for model-based design in the context of software development.
- Establish common expectation on model quality when doing co-development between different parties.
- Help non-software developers to understand how they contribute to software development.
- Clarify impact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

Types of Models



A software development life cycle supported by different types/levels of design models to fully support MBD.

MQO clarifies exchanges and discussions



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

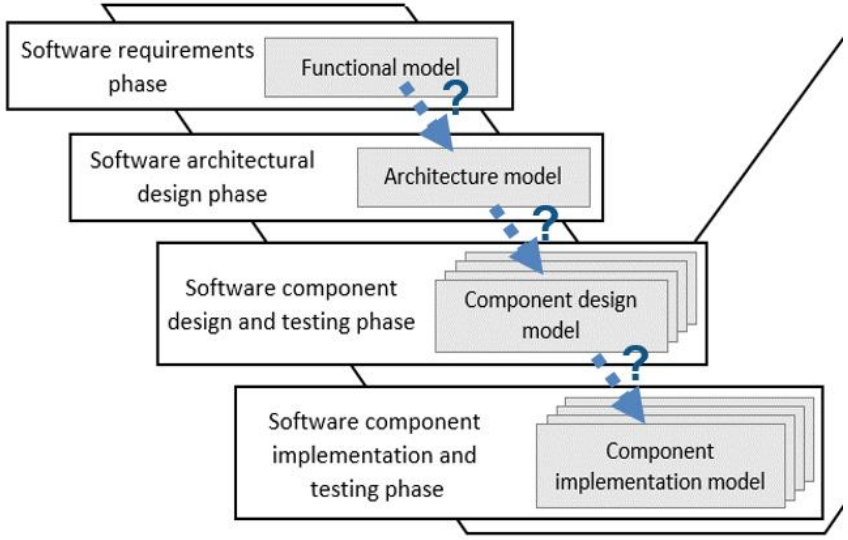
MQR ID	MQR Title	MQO-1	MQO-2	MQO-3	MQO-4
MQR-01	Model layout	M	M	M	M
MQR-02	Model comments	M	M	M	M
MQR-03	Model links to requirements	M	M	M	M
MQR-04	Model testing against requirements	M	R	M	M
MQR-05	Model compliance with modeling standard		M	M	M
MQR-06	Model data		M	M	M
MQR-07	Model size			M	M
MQR-08	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

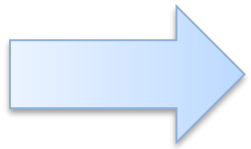
Example of Model Quality Requirement

Example of a 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	<table><tr><th>MQO-1</th><th>MQO-2</th><th>MQO-3</th><th>MQO-4</th></tr><tr><td></td><td></td><td>**Mandatory**</td><td>**Mandatory**</td></tr></table>
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

Guidelines on model reuse

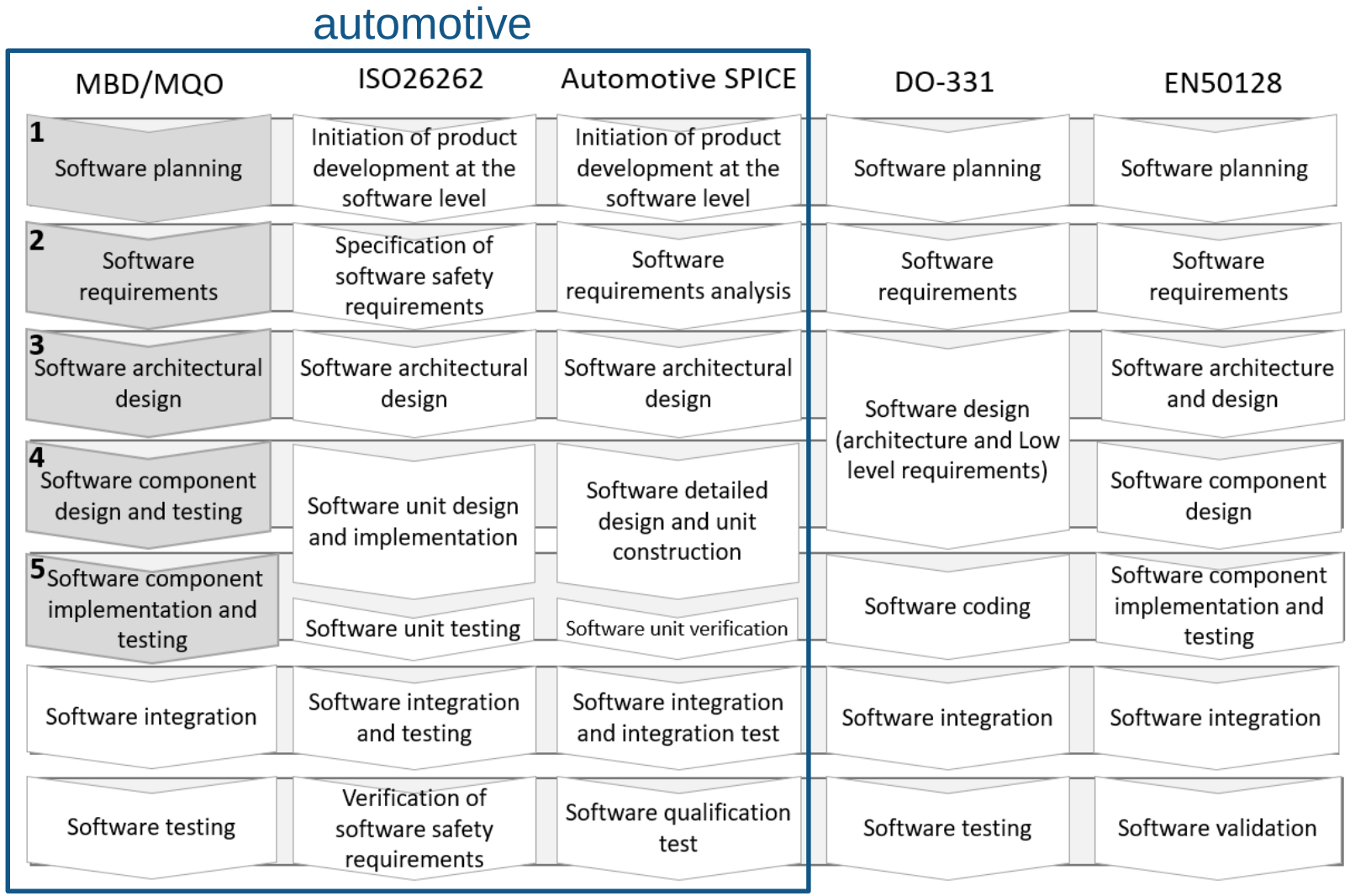




Type of design model	Design model aspect: Architectural	Algorithmic	Code generation
Functional model	Prototyping	Prototyping	Prototyping
	refine (1a)	reuse (2a)	
Architecture model	Production	Prototyping	Prototyping
	reuse (1b)	refine (2b)	
Component design model	Production	Production	Prototyping
	reuse (1c)	reuse (2c)	refine (3c)
Component implementation model	Production	Production	Production

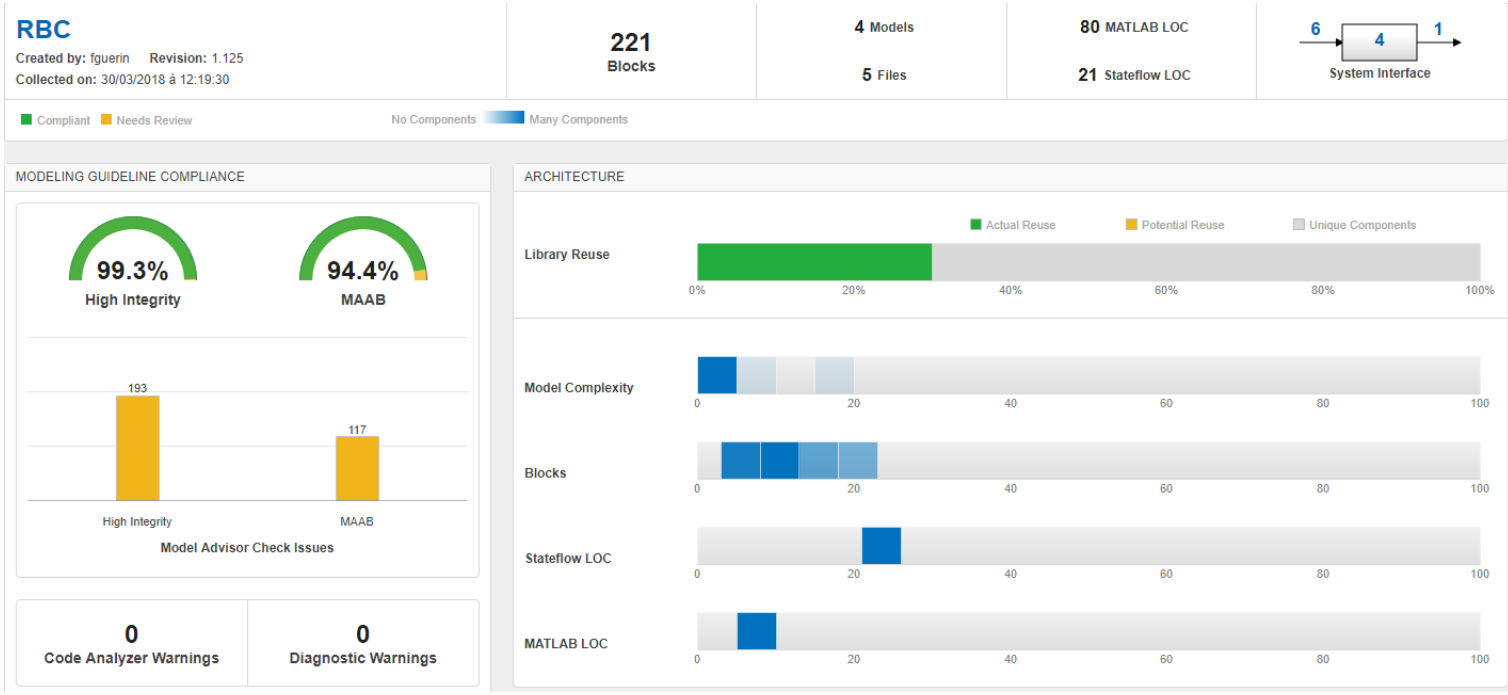
Compatible with existing industry standards

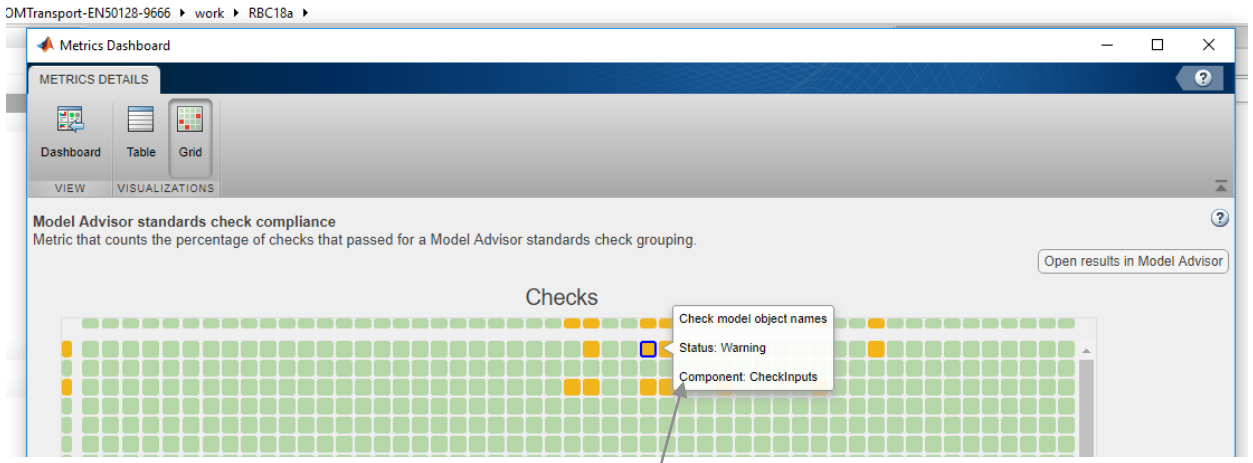
- Complementary and compatible with existing standards
- Provide metrics and threshold to address quality requirements referred in standards
- Additional guidelines on planning phase to define responsibilities, and ensure workflow compatibility.



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

Agenda

1. Background and Motivations
2. Introduction to Model Quality Objectives
3. Deployment and expected gains
4. Conclusion

Communication

- Internal company communication
  - Communication to engineering teams for feedback (engineering, quality, safety)
  - Presentation to management for approval

- Public communication
  - ERTS^2 2018 (Toulouse, February 2018)
  - MathWorks Automotive Conference (Stuttgart, April 2018)
  - MATLAB Expo (Paris, June 2018)
  - MathWorks website (to be scheduled)

Example of deployment at Renault / Bosch / Valeo

- Training
  - MQO will be part of some standard MBD training for new users. (Renault / Valeo)
  - Large internal communication on MQO is planned for the coming months. (Renault / Valeo / Bosch)
- Projects
  - MQO will be integrated in a future version of our Request For Quotation package as an additional informative reference. (Renault)
  - Discuss with customer at the start of new projects (Bosch)
- Process
  - MQO will be taken into account as an input for our future software development process. (Renault)

Expected gains

- The organizations that apply MQO should experience the following benefits:
  - Shared understanding of Model-Based Design within the organization
  - Application of a quality model adapted to MBD projects and compatible with industry software quality and safety standards
  - Assessment of model quality at different phases of projects

- The organizations that also collaborate with partners to execute MBD projects should experience the following benefits:
  - Clear split of responsibility between parties at the beginning of projects
  - Common understanding of model quality
  - Common expectation on model quality when sharing models

Agenda

1. Background and Motivations
2. Introduction to Model Quality Objectives
3. Deployment and expected gains
4. Conclusion

Conclusion

- We expect MQO to improve MBD co-development between customers and suppliers.

- We look for feedback to improve MQO:
  - Additional Model Quality Requirements ?
  - Additional types of models (e.g. system level) ?
  - Applicability in other industries ?

Q/A ?