MES
MODEL ENGINEERING SOLUTIONS

AGILE METHODS IN SAFETY-CRITICAL SOFTWARE DEVELOPMENT

MGI Group, September 04, 2018

SOFTWARE QUALITY. IN CONTROL.

SOLUTIONS FOR INTEGRATED QUALITY ASSURANCE OF EMBEDDED SOFTWARE

EMERGING COMPLEXITY

Software complexity trend
e.g. Body Control Modules (BCM)

Lines of

Problem

Avg. pro
inflow / r

* Average values per wee

Continental

Software complexity trend
e.g. Gateways (GW)

Lines o

Probler

Avg. pr
inflow /

* Average values per w

Continental

Software complexity trend
e.g. infotainment devices

	Device 1 2009-2012		Device 2 2011-2015
Lines of code	525,000	x6	3,000,000
Problems reported	13,000	x3.5	45,000
Avg. problems inflow / resolved*	150 / 200	x1.7	285 / 335

* Average values per week during active development

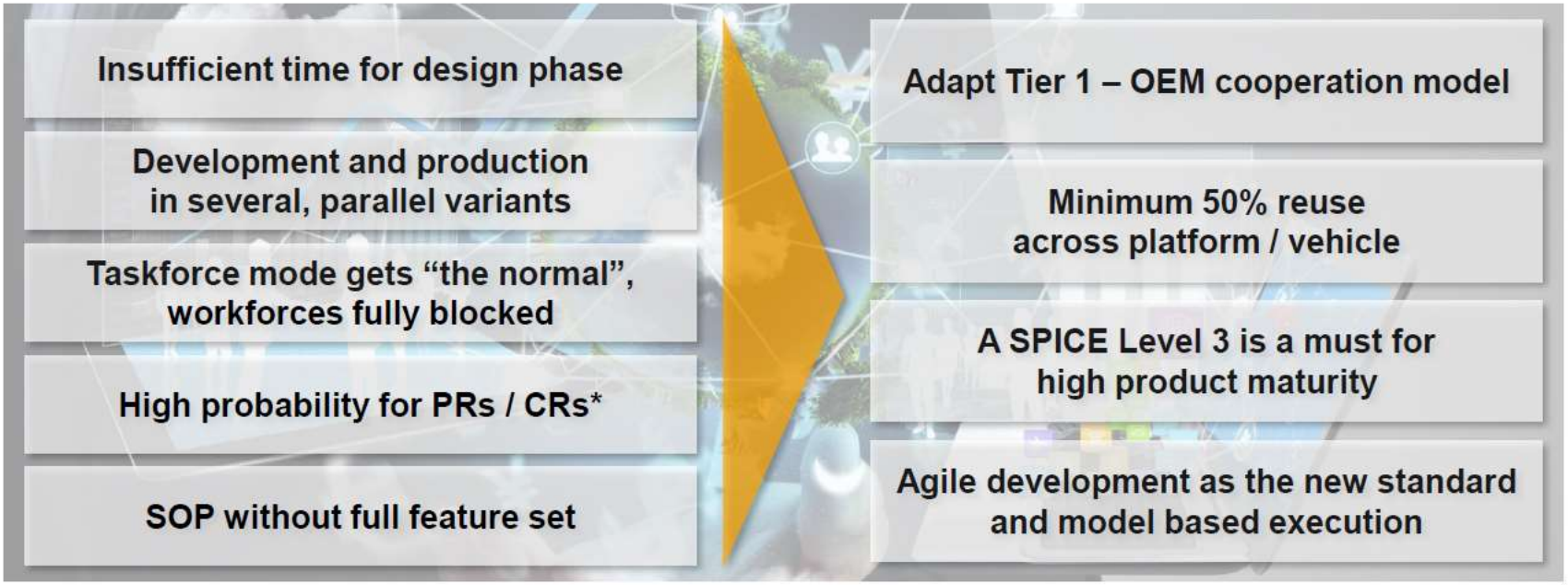
Continental

20th International Conference
"Advances in Automotive Electronics"
Public

June 15, 2016
H. Matschi © Continental AG

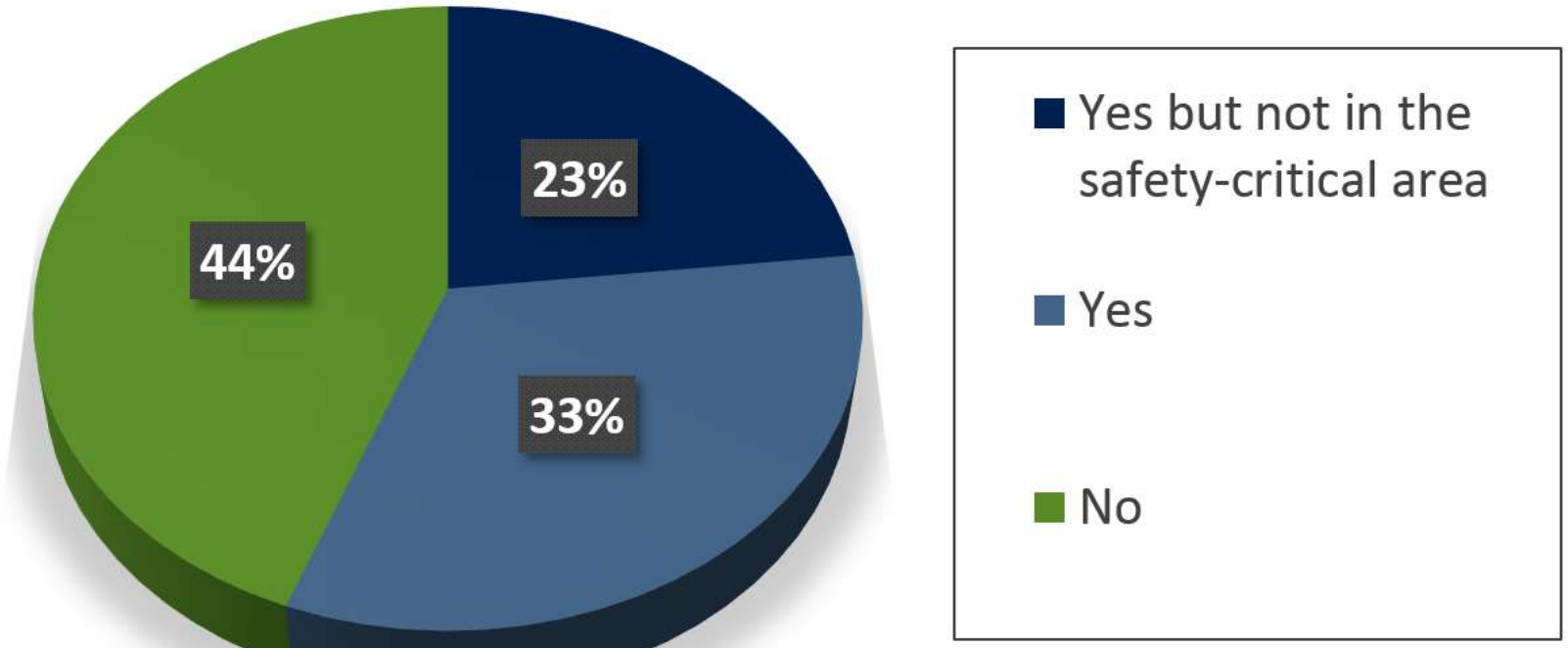
High number of requirements in less time

Agile development, close with the customer



* PRs: problem reports, CRs: changes requests

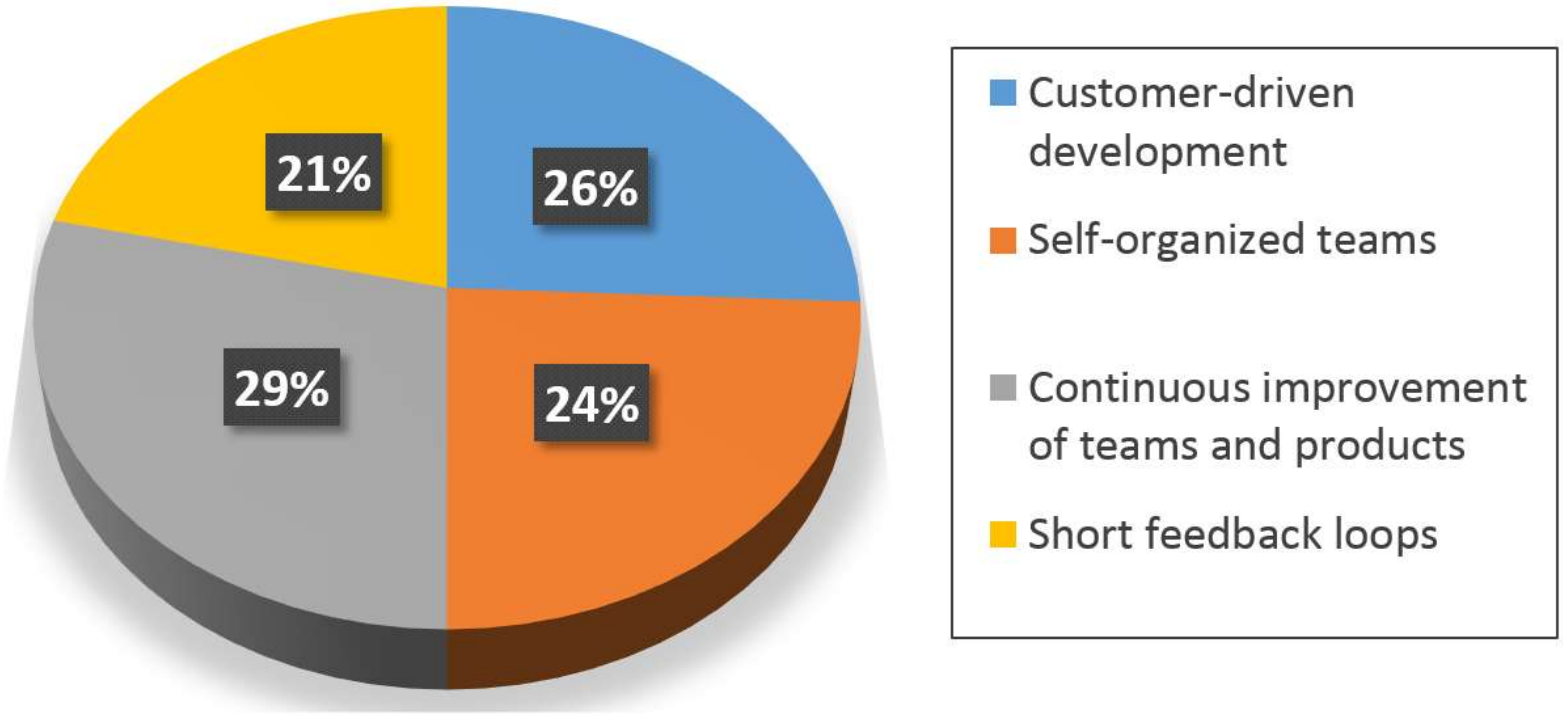
ARE AGILE METHODS USED AT YOUR WORKPLACE?



Your feedback

44 answers as of Sept. 04, 2018

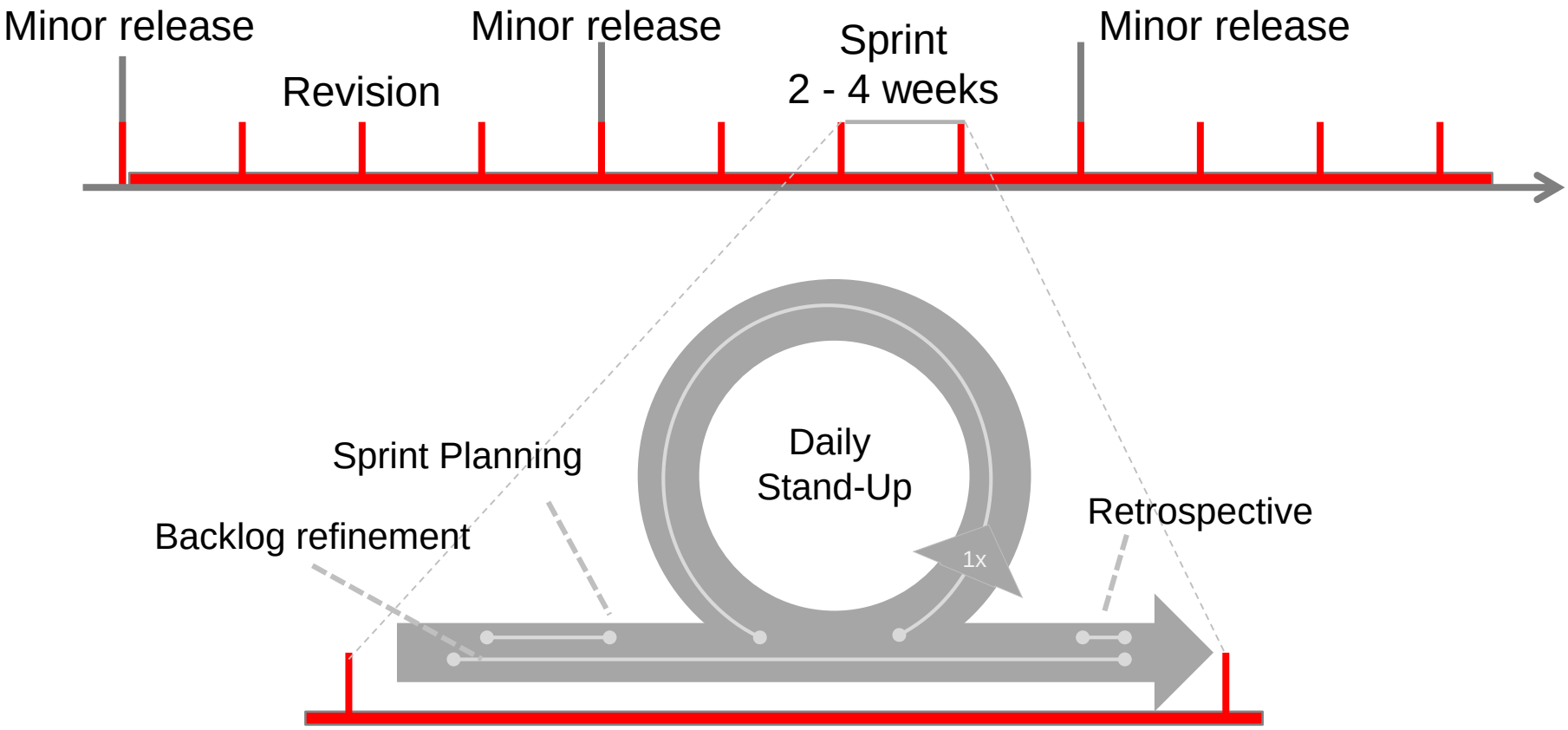
BIGGEST ADVANTAGES OF AGILE DEVELOPMENT?



Your feedback

44 answers as of Sept. 04, 2018

SCRUM – CONTINUOUS DEVELOPMENT



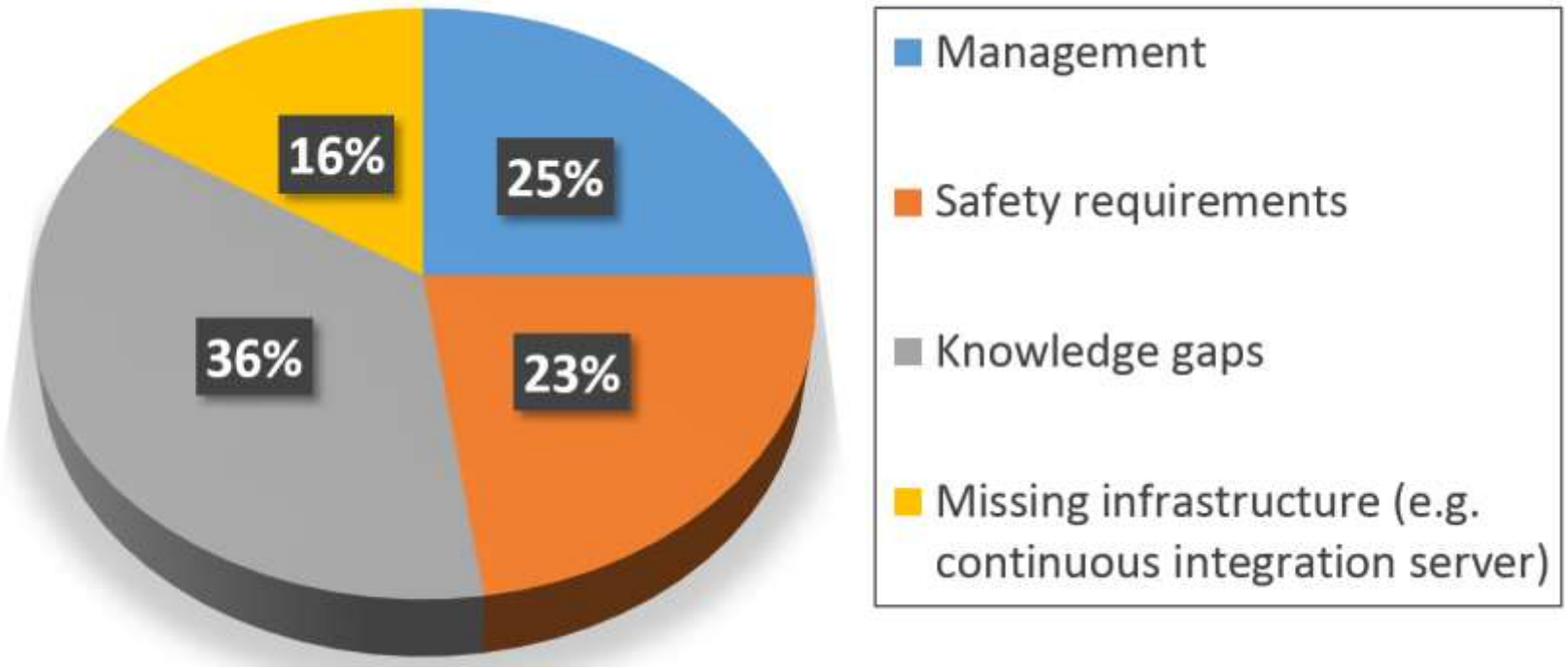
based on: AUTOSAR – Development of Adaptive Platform

SCRUM – MAIN ARTIFACTS

- Product Backlog maintained by Product Owner
  - List of all work items at functional level
  - Living document capturing new requirements during development
  - High-level epics, detailed user stories
- Sprint Backlog
  - Selection of user stories is "drawn for implementation" by team
  - Size of user story is such that implementation can be achieved in 1-2 days
  - Burn-down chart as leading indicator for progress
- Definition of Ready / Definition of Done
  - Main vehicle to enforce quality assurance
  - Joint agreement in team on completion criteria for user stories
  - Fine grain quality assurance mostly supported by continuous integration

see: Hundermark "Do better Scrum"

BIGGEST OBSTACLES TO WORKING IN AN AGILE WAY?



Which are the biggest obstacles in detail?

44 answers as of Sept. 04, 2018

BIGGEST OBSTACLES IN DETAIL

Obstacle	Deg	Details
Knowledge Gaps	33%	Engineers did not fully understand what agile principles meant nor the tools applied in the process (in contrast to supervisors) Knowledge transfer from Scrum Master to engineers + understanding the objectives
Management	23%	Procedures involved in creating ISO work products: Breaking work down into user stories is too time-consuming and risks deadlines being missed Selection of user stories / tasks by team will not ensure deadlines set by customers are met (=> Including customer in selection / prioritization of user stories counters this risk) Low interaction with / involvement of customer
Safety Requirements	19%	Safety requirements were continuously incorporated into the code => benefit Safety requirements shall be determined upfront and completely (before determining further functional requirements) Safety requirements should be defined without interference, with functional requirements reflected in an appropriate SW architecture Validation of safety requirements requires dedicated procedures leading to more extensive DoDs
Missing Infrastructure	15%	Tools are very important to do things like monitor and define the agile process + continuous integration servers for SW validation and build Continuous integration is central + support of reviews via automatic creation of review baselines (e.g. here particular style guide needed in order to have all information captured in printed documents)

CORE CONCEPTS – SAFETY-RELATED SOFTWARE

- Essential work products for safety-related software
  - Safety goals and safety requirements
  - System and software architectural designs
  - Software
  - Validation reports contributing to safety case
- Agile development must not focus on software only but treat essential work products equally
- Handling proposals:
  - a) have individual user stories for each work product => challenge consistency
  - b) develop such small increments that all work products are touched in the same user story => very small increments
  - c) introduce dedicated sprints for a posteriori synchronization of essential work products => gaps in safety case might be discovered late

CORE CONCEPTS – QM SOFTWARE

- Essential work products for QM software
  - System and software architectural designs
  - Functional requirements
  - Software
- Agile development may be applied to develop software in order to implement functional requirements
- Handling proposals:
  - Apply regular concepts of Scrum method
  - Ensure that boundary conditions stated in the software architecture are met by implementation => Definition of Done
  - Define appropriate quality assurance methods as Definition of Done

CORE CONCEPTS – OVERVIEW

	Work Products	(Elements of …) Definition of Done
QM Unit / Component	Requirements (In) Product Backlog (In/Out) Software (Out) *Further WPs*	*Sample definitions?*
ASIL Unit / Component	Safety Requirements (In) Product Backlog (In/Out) Software (Out) Validation Report (Out) Functional model for early unit validation (e.g. simulation results at model level) Safety analysis at SW level	*Sample definitions?* Different types of sprint depending on the stage of development (e.g. early functional evaluation, HW integration, in order to cope with specific constraints of embedded system development of mechatronic systems)

CORE CONCEPTS – EXAMPLE

	Work Products	(Elements of …) Definition of Done
QM Unit / Component	Requirements (In) Product Backlog (In/Out) Software (Out)	Quality assurance (at QM level) applied Constraints of SW architecture met Validation suite incremented to capture new requirements
ASIL Unit / Component	Safety Requirements (In) Product Backlog (In/Out) Software (Out) Validation Report (Out)	Quality assurance (for respective ASIL) applied Constraints of SW architecture met Validation suite incremented to capture new requirements Contribution to safety case updated

DEFINITION(S) OF DONE – SAMPLES

	Modeling Guidelines	Metrics	Dynamic tests
QM Unit	• MAAB • Starter Set • Code Generator	• Max local complexity < 750	• Requirements Coverage > 75% • Code Coverage > 75%
ASIL Unit	Above + • MISRA SL/SF • Functional Safety	• Max local complexity < 500 • Max global complexity of unit < 1500 • No ineffective interfaces • No clones	• Requirements Coverage = 100% • Code Coverage > 98%

- Detailed analysis of project needs required for Definition of Done
- Metrics:
  - Model coverage
  - Documentation of safety-parts present (as being part of the models and in addition to that (e.g. traceability))

SAVE THE DATE!

MES USER CONFERENCE 2018 – "MBD TOOL CHAIN FOR AGILE DEVELOPMENT" OCTOBER 11 - 12, 2018

- Venue: palisa.de at Umspannwerk Ost, Palisadenstraße 48, 10243 Berlin (Germany)
- Registration fee: € 220 plus VAT
- Limited number of participants, please register in advance by September 20, 2018

We look forward to welcoming you!

MGIGROUP – NEXT ONLINE MEETING

- Tuesday, December 4, 2018
  3:00 pm CET (Berlin)
  9:00 am EST (Detroit)
  7:30 pm IST (Bangalore)
  10:00 pm CST (Beijing)
  11:00 pm JST (Tokyo)

- Link to Event:
  https://model-engineers-event-en.webex.com/model-engineers-event-en/onstage/g.php?MTID=e72c2b2930e3302aa98bb4744cf12d665

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

ESSENTIAL REQUIREMENTS

- Demonstrate that system will cause no harm

- Transparent and well-documented line of argument that
  - a) any safety goal has been comprehensively refined to SW / HW safety requirements
  - b) all safety requirements by SW or HW units have been verified for full implementation
  - c) all design decisions, refinements, verifications have been confirmed by reviews

… is needed

- and collected in Safety Case

ROLE OF ARCHITECTURES

- System and software architectural designs are crucial
  - Identify components contribution to safety goals
  - Implementing safety requirements
- Note: Major design principle “Enforce low complexity”
  - Keep the perimeter of units implementing safety requirements small, and
  - Try to enlarge the scope of components and units without safety requirements => development according to QM feasible
- Proper system and software architectures will limit amount of safety-related components