

Challenges and Opportunities in Design and Operation of Intelligent Cyber-Physical Systems

Invited Talk, 19th International Conference on Runtime Verification (RV)
Part of the 3rd World Congress on Formal Methods
Porto, Portugal. October 10, 2019

Akshay Rajhans, PhD
arajhans@mathworks.com
<https://arajhans.github.io>

Talk outline

- Cyber-physical systems: a feature classification
- “Runtime” verification at design time: simulation as a proxy for run time
- Runtime analysis at operation time: From CPS to IoT and Digital Twins
- Challenges and future outlook

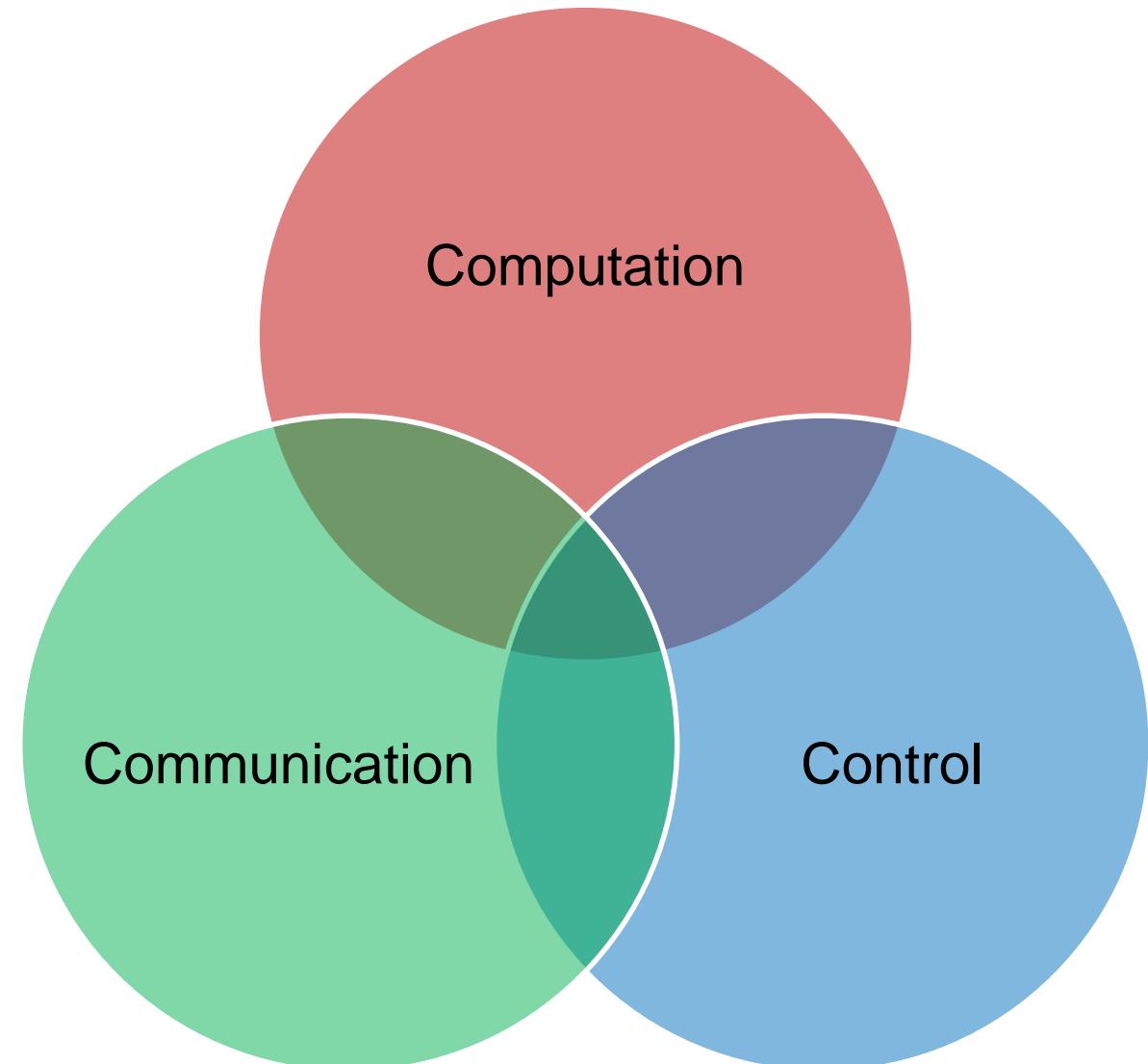
Cyber-physical systems: umbrella term, not a precise definition

“The term cyber-physical systems (CPS) refers to a new generation of systems with integrated computational and physical capabilities that can interact with humans through many new modalities.

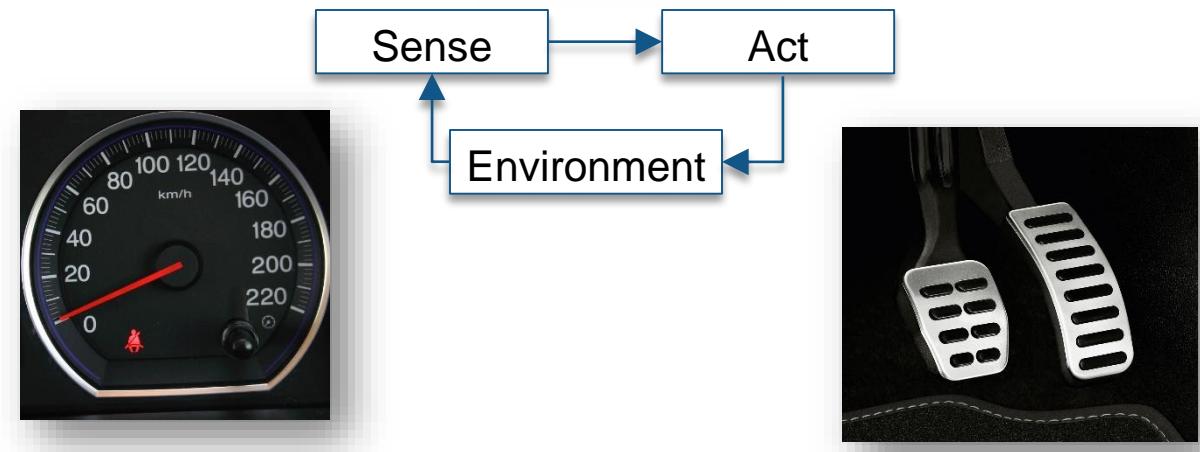
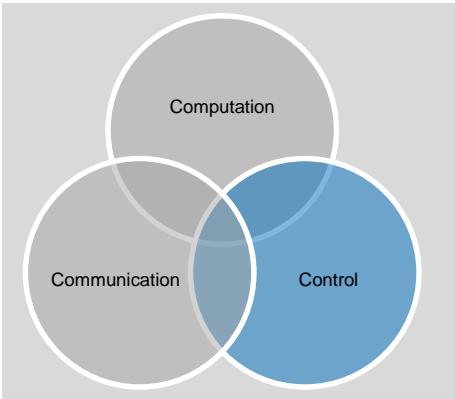
The ability to interact with, and expand the capabilities of, the physical world through **computation**, **communication**, and **control** is a key enabler for future technology developments.”

- Helen Gill and Kisan Baheti, NSF

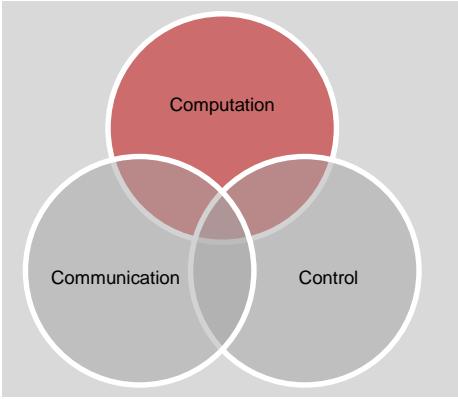
IEEE Impact of Control Technology, T. Samad and A.M. Annaswamy (eds.), 2011. Available at www.ieeecss.org.



Control – closing the loop over the physical environment



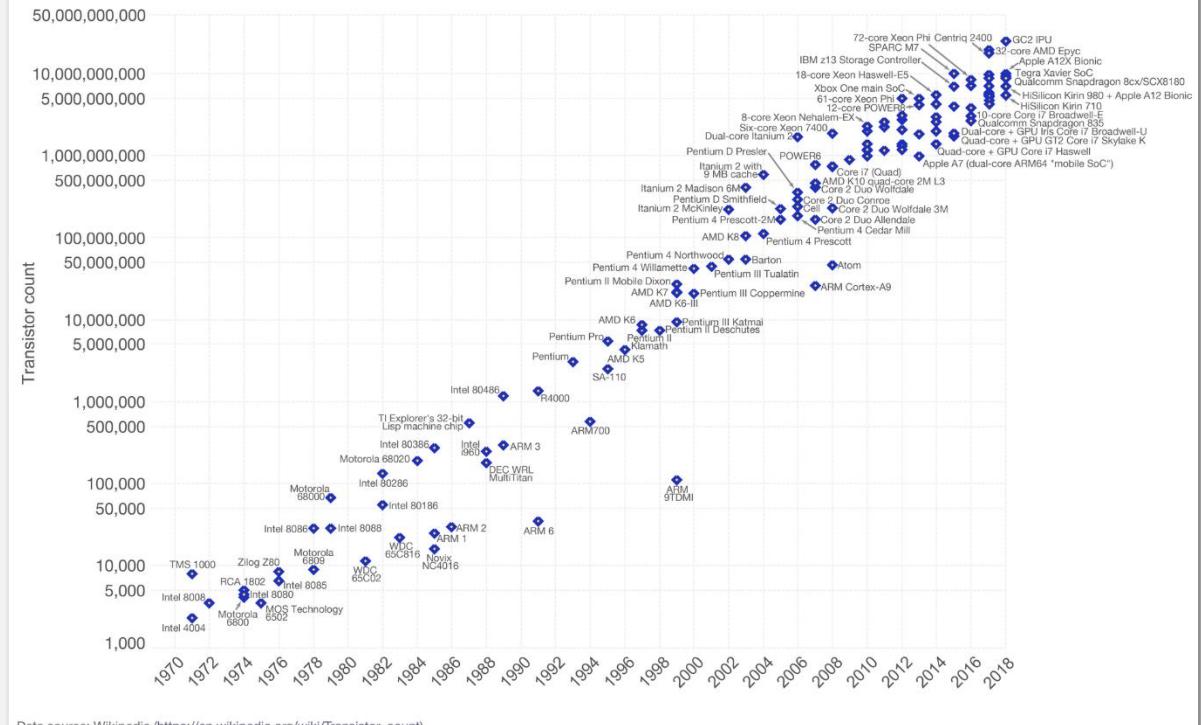
Computation – fueled by Moore's law



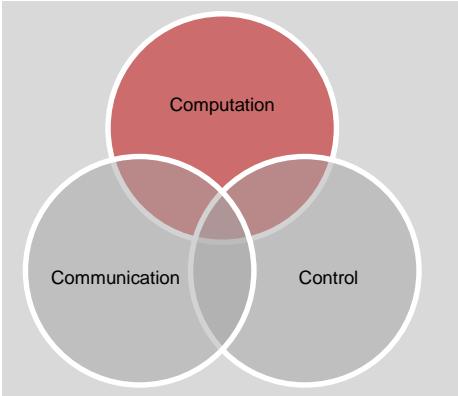
Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.

OurWorld
in Data



Computation – fueled by Moore's law



**Cost of
Printing**

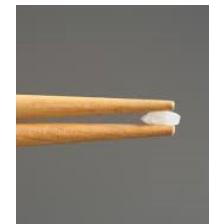


>



**Annual
Production**

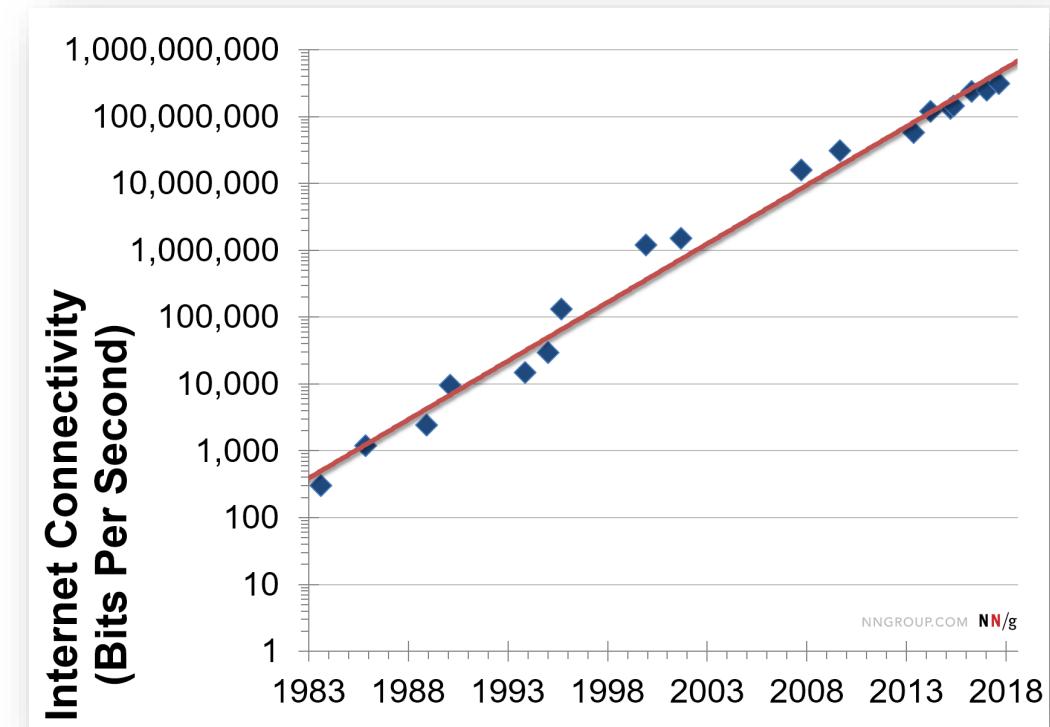
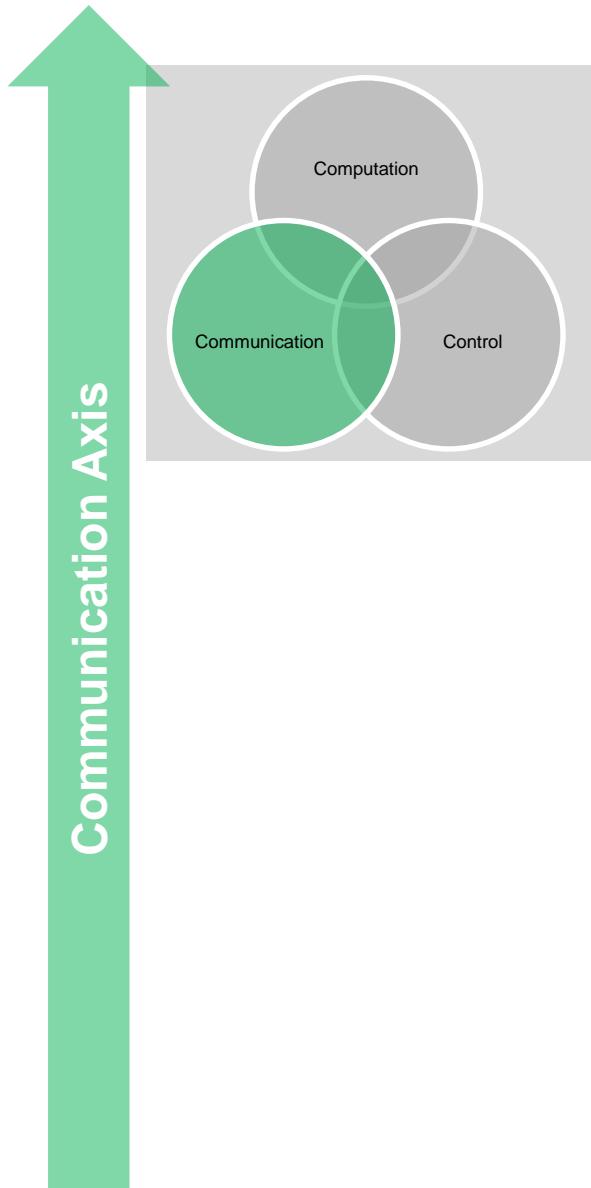
>



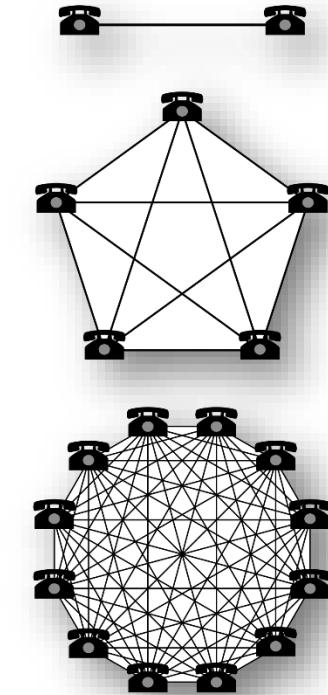
Source: Intel, IEEE

Computation Axis

Communication – Nielsen's law, Metcalfe's law

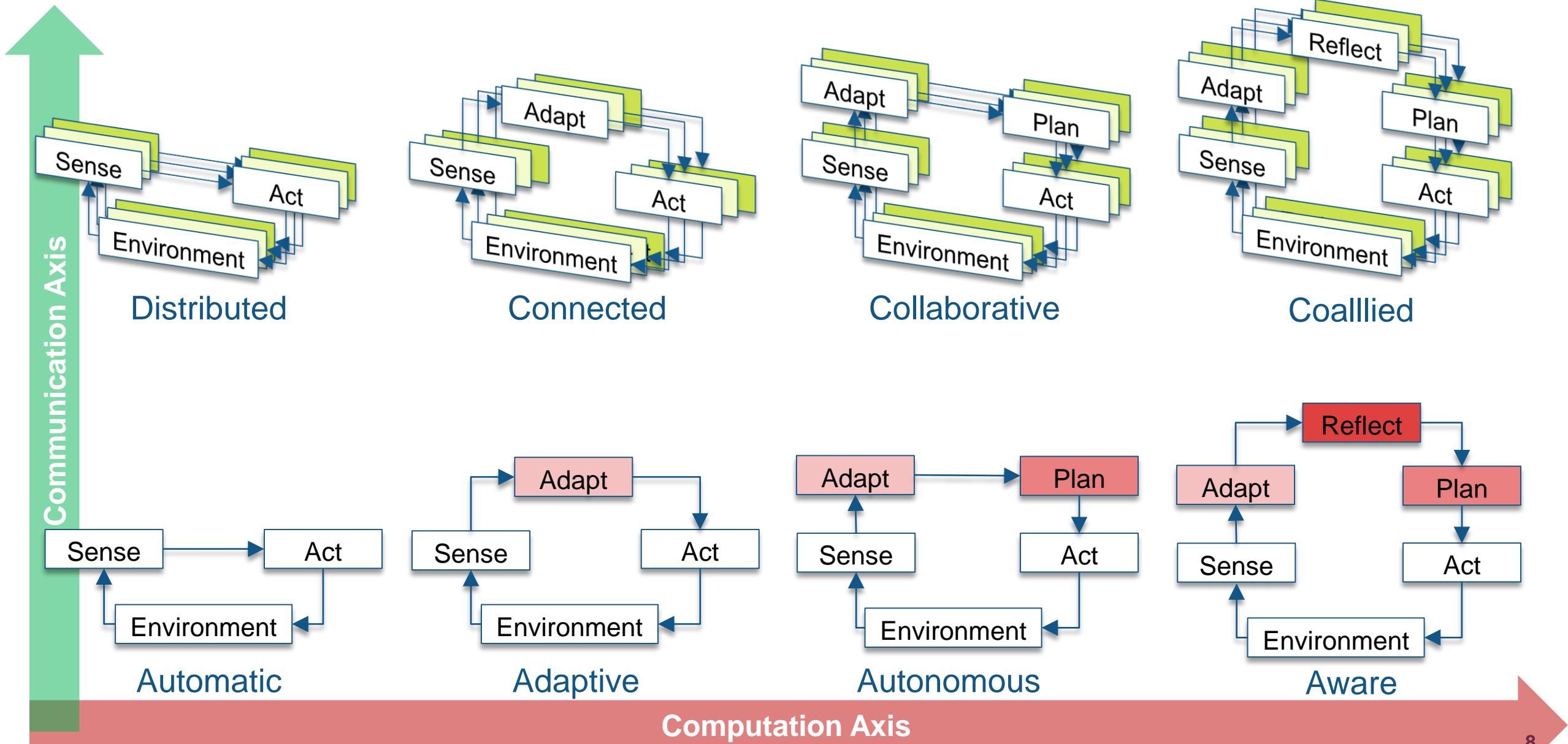


Source: nngroup.com



Source: Wikipedia

CPS feature classification



References

Configuration	Behavior			
	Reflexive	Reactive	Reasoned	Reflective
Individual	Automatic	Adaptive	Autonomous	Aware
Ensemble	Distributed	Connected	Collaborative	Coallied

[CMR⁺20] S. Castro, P. Mosterman, A. Rajhans, R. Valenti, “*Challenges in the Operation and Design of Intelligent Cyber-Physical Systems*”, to appear.

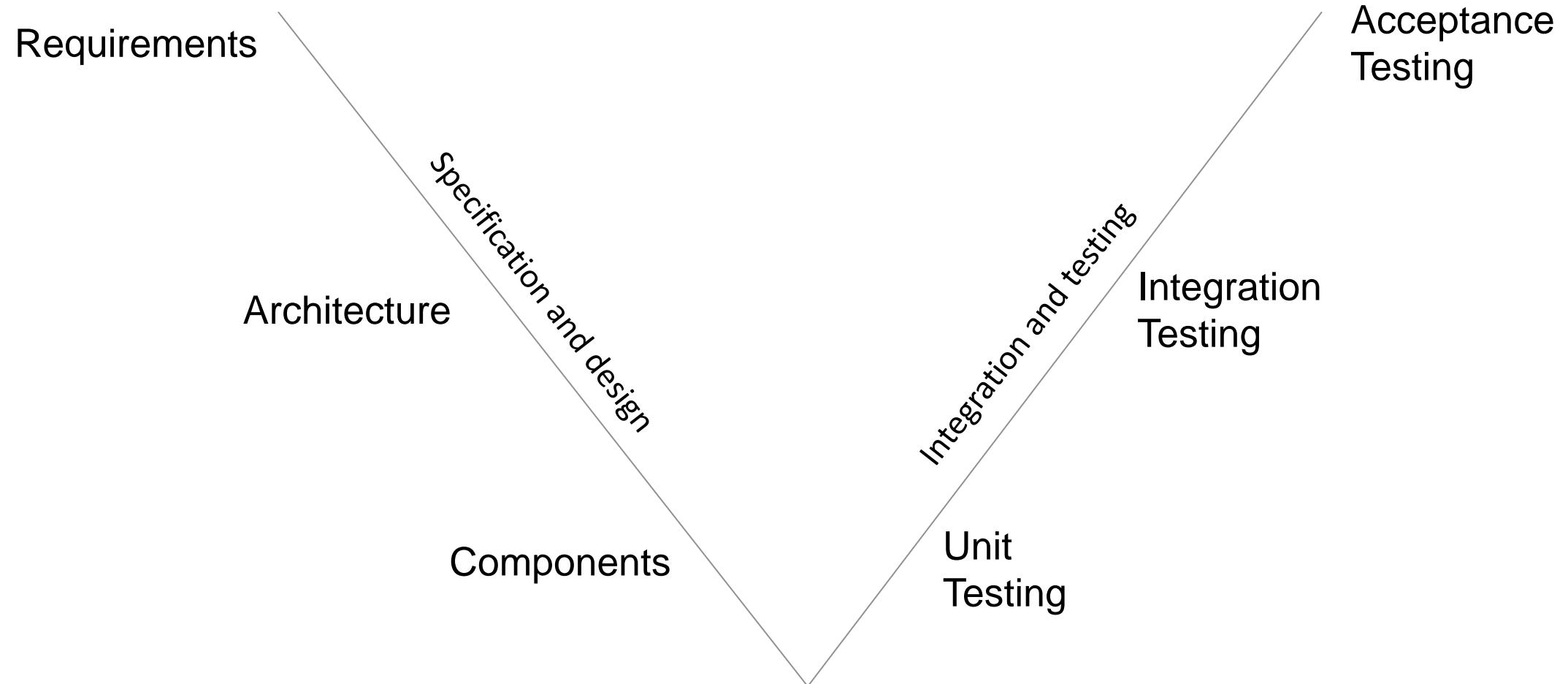
[ABK⁺19] F. Allgöwer, J. Borges de Sousa, J. Kapinski, P. Mosterman, J. Oehlerking, P. Panciatichi, M. Prandini, A. Rajhans, P. Tabuada, P. Wenzelburger, “*Position paper on the challenges posed by modern applications to cyber-physical systems theory*”, Nonlinear Analysis: Hybrid Systems, Volume 34, Pages 147-165, November 2019.

[TBD⁺18] A. Tolk, F. Barros, A. D’Ambrogio, A. Rajhans, P. J. Mosterman, S. S. Shetty, M. K. Traoré, H. Vanheluwe, and L. Yilmaz, “*Hybrid Simulation for Cyber Physical Systems – A Panel on Where are we Going Regarding Complexity, Intelligence, and Adaptability of CPS Using Simulation*,” Spring Simulation Multi-Conference, 2018.

Outline

- CPS feature classification
- “Runtime” verification at design time: simulation as a proxy for run time
- From CPS to IoT and Digital Twins: runtime analysis
- Challenges and future outlook

Model-Based Design: Models as a proxy for the real system



Model-Based Design:
Use of computational models throughout

Implementation

Simulink

Stateflow

Simscape

CPS

SimEvents

MATLAB

```

function [residual, xhatOut] = extkalman(meas, de
persistent P, xhat;
Phi = [1 deltat 0 0; 0 1 0 0 ; 0 0 1 deltat; 0 0
Q = diag([0 .005 0 .005]); R = diag([300^2 0.
P = Phi*P*Phi' + Q;
xhat = Phi*xhat;
Rhat = sqrt(Q);
Bhat = atan2(Rhat, P);
yhat = [Rhat;
M = [cos(Bhat) 0 -sin(Bhat);
-sin(Bhat)/Rhat 0 cos(Bhat)/Rhat
0];
residual = meas - yhat;
W = P*M'*inv(M*P*M'+ R);
xhat = xhat + W*residual;

```

Simulations for increasingly faithful proxies of runtime behavior

- Model-in-the-loop simulation
- Software-in-the-loop simulation
- Processor-in-the-loop simulation
- Hardware-in-the-loop simulation
- Gaming-engine-in-the-loop

simulate / test the model
the generated code
code on the processor
plant on real-time h/w
visualization, physics



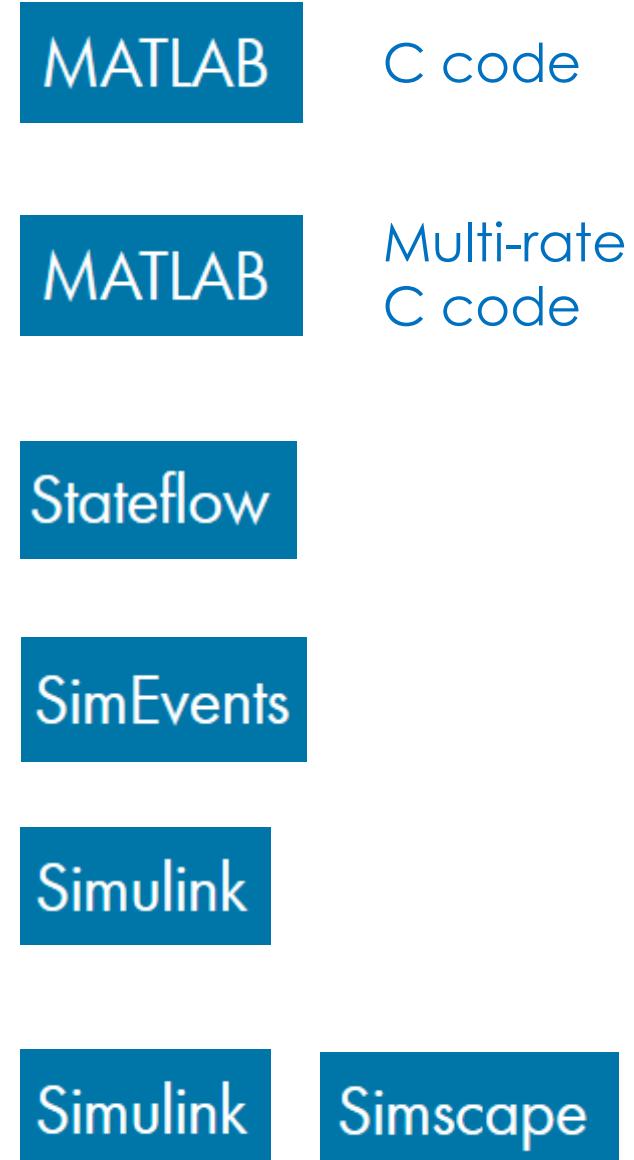
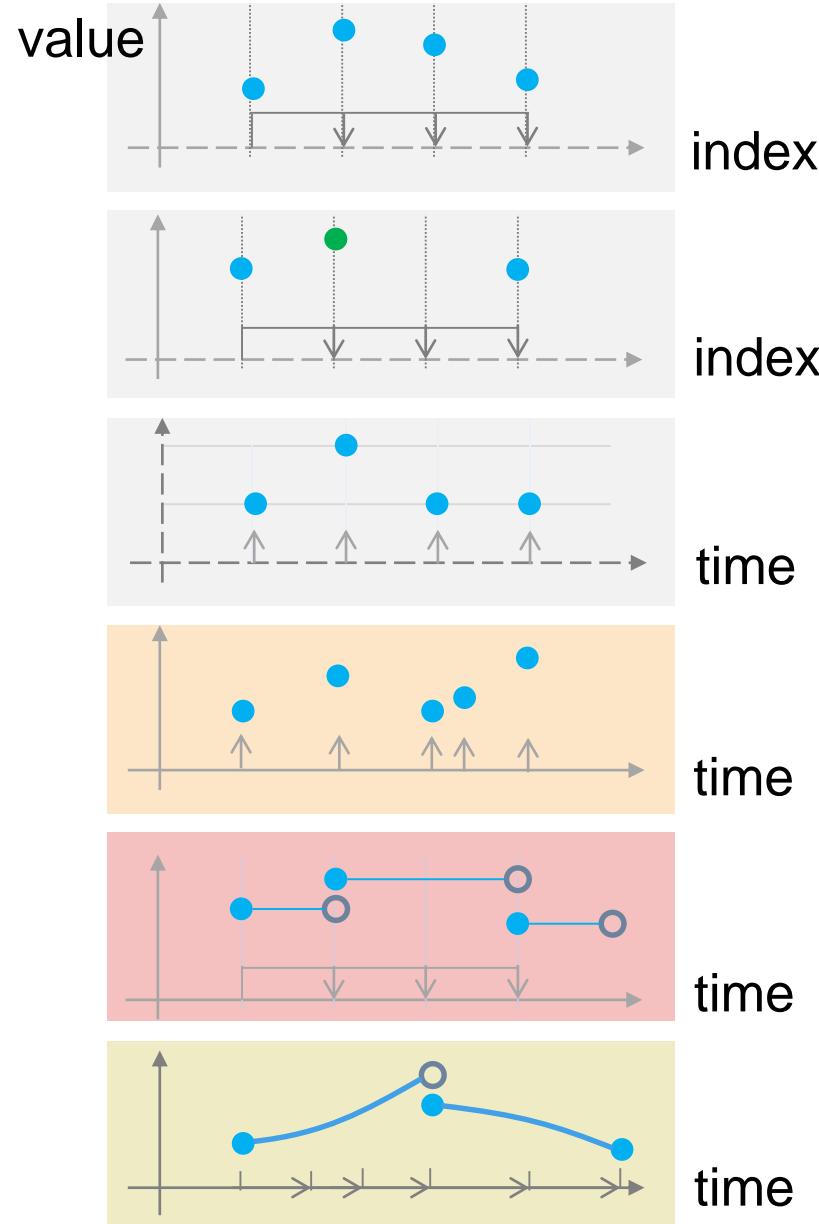
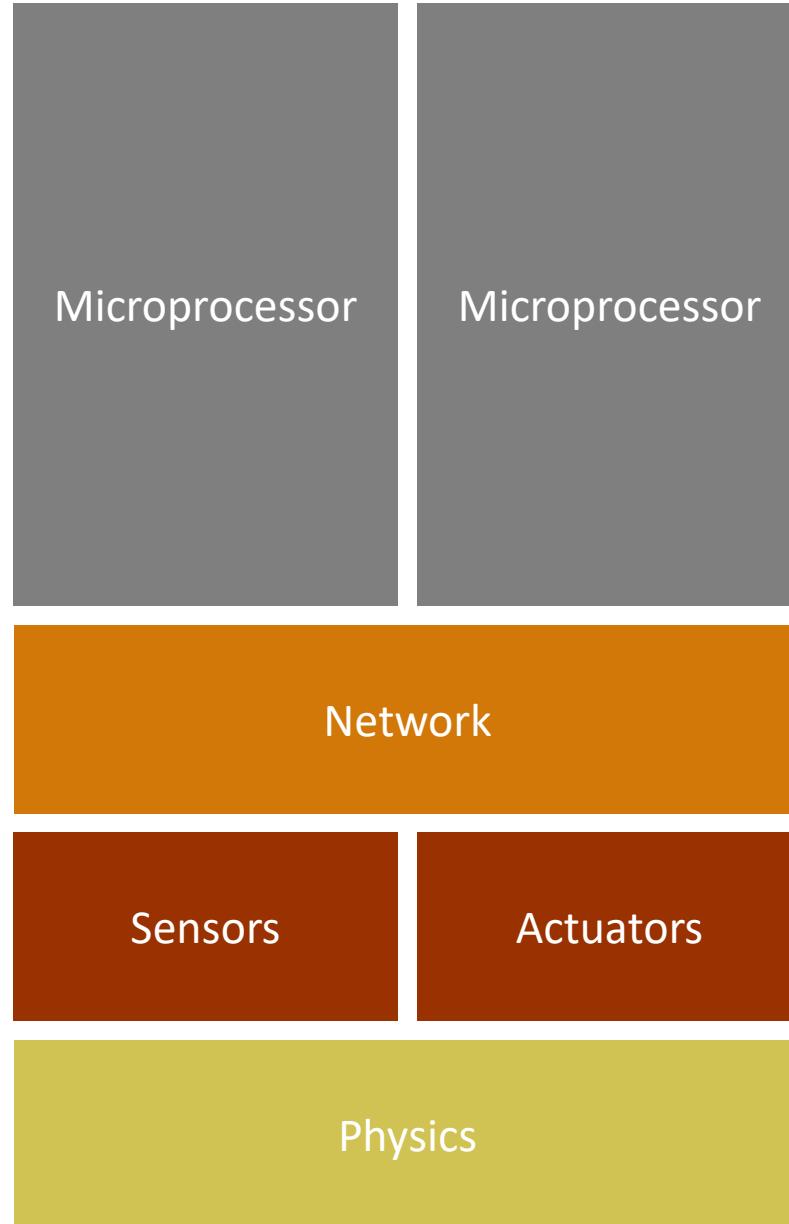
Ride & handling



Chassis controls



ADAS / AD



Not today – how to address heterogeneity formally?

[MRM⁺20] P.J. Mosterman et al., “*Simulation of Hybrid Dynamic Systems*”, Springer Encyclopedia of Systems and Control, Second Edition, to appear.

[RBR⁺14] A. Rajhans et al., “*Supporting Heterogeneity in Cyber-Physical System Architectures*”, IEEE Transactions on Automatic Control, Special Issue on Control of CPS, Vol. 59, Issue 12, pages 3178-3193.

[R13] A. Rajhans, “*Multi-Model Heterogeneous Verification of Cyber-Physical Systems*,” **PhD Thesis**, Carnegie Mellon University, 2013.

[RK13] A. Rajhans and B. H. Krogh, “*Compositional Heterogeneous Abstraction*,” 16th ACM International Conference on HSCC, 2013.

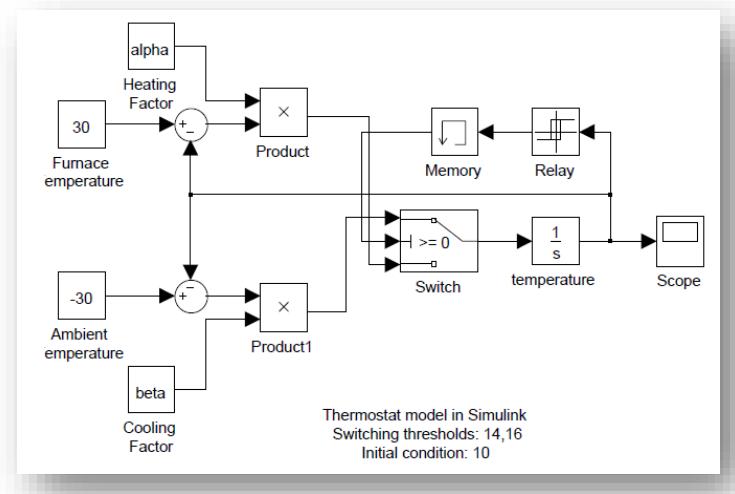
[RK12] A. Rajhans and B. H. Krogh, “*Heterogeneous Verification of Cyber-Physical Systems Using Behavior Relations*,” 15th ACM International Conference on HSCC, 2012.

[RBL⁺11] A. Rajhans et al., “*Using Parameters in Architectural Views to Support Heterogeneous Design and Verification*,” 50th IEEE CDC, 2011.

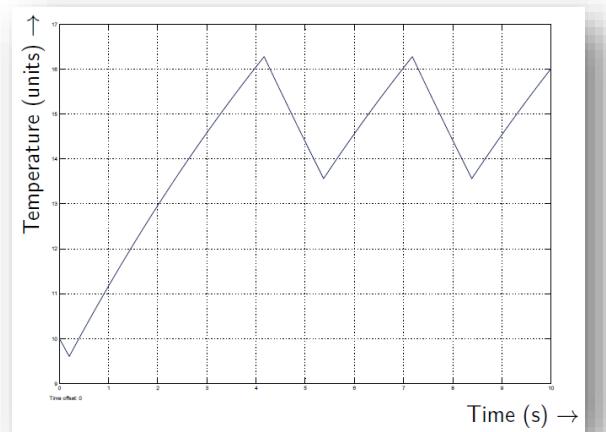
[BDK⁺10b] A. Bhave et al., “*Augmenting Software Architectures with Physical Components*,” Embedded Real Time Software and Systems (ERTS²), 2010.

[RCS⁺09] A. Rajhans et al., “*An Architectural Approach to the Design and Analysis of Cyber-Physical Systems*,” Third International Workshop on Multi-Paradigm Modeling (MPM), 2009.

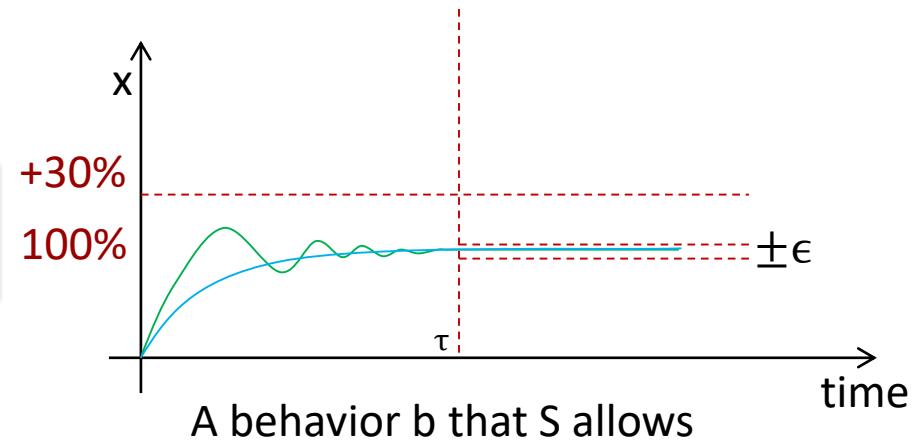
Models and Specifications



Model M

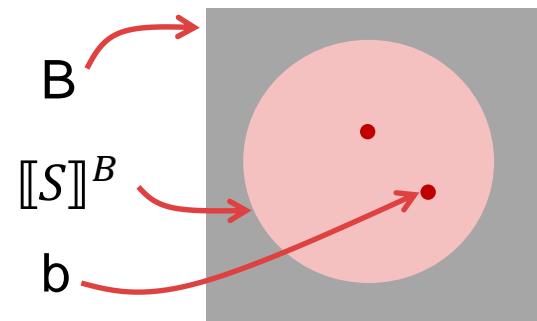
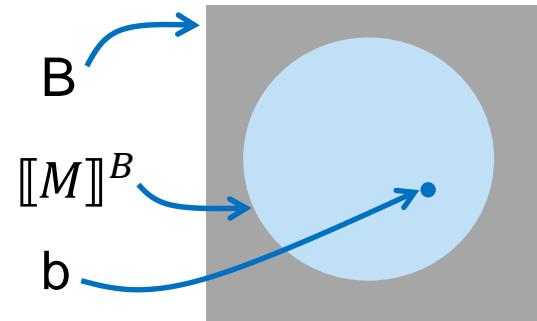


A behavior b that M exhibits

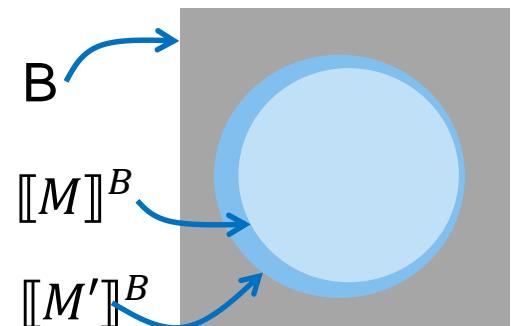


"overshoot is never more than 30% and settling time is less than τ "

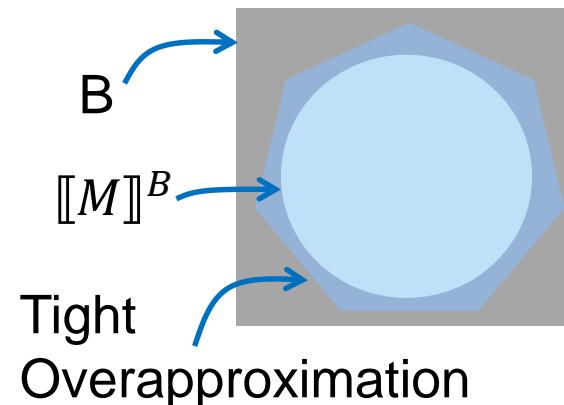
Specification S



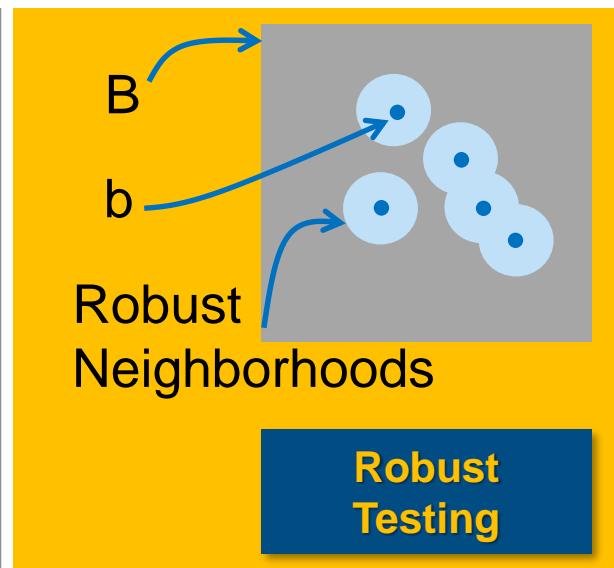
Various verification problems



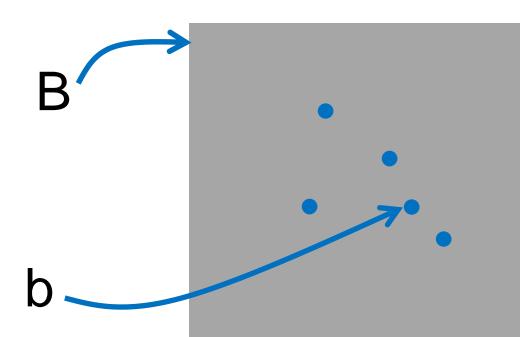
Abstraction



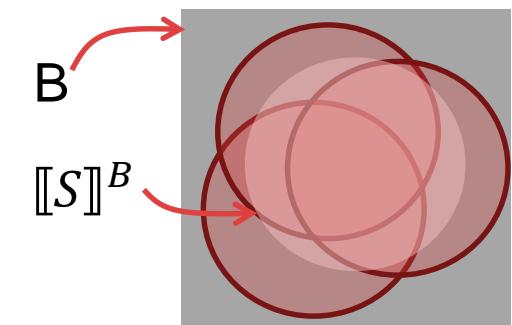
Reachability Analysis



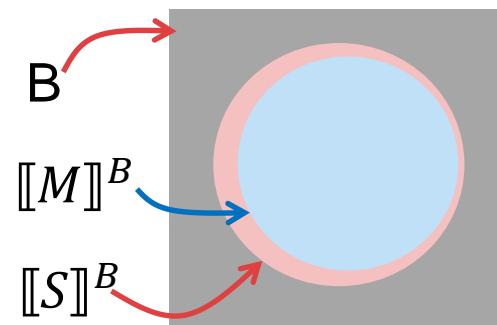
Robust Testing



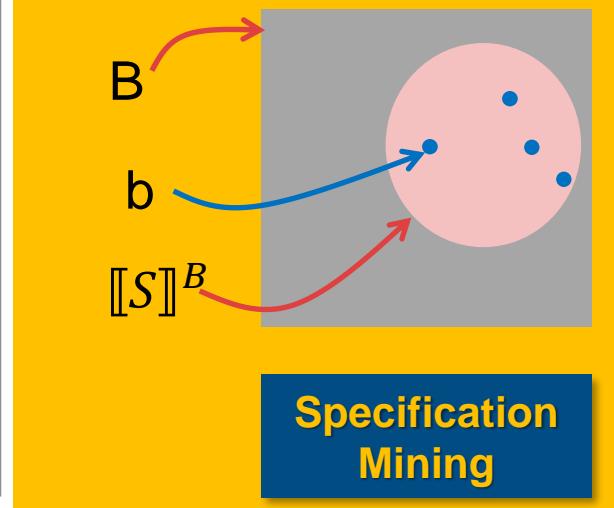
Testing



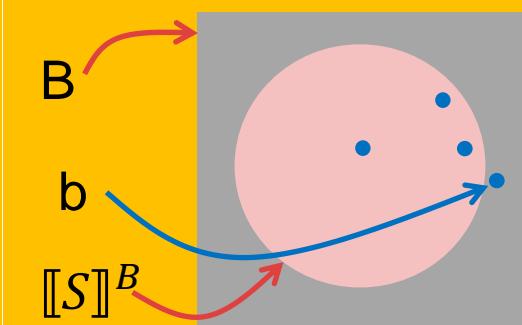
Theorem Proving



Model checking

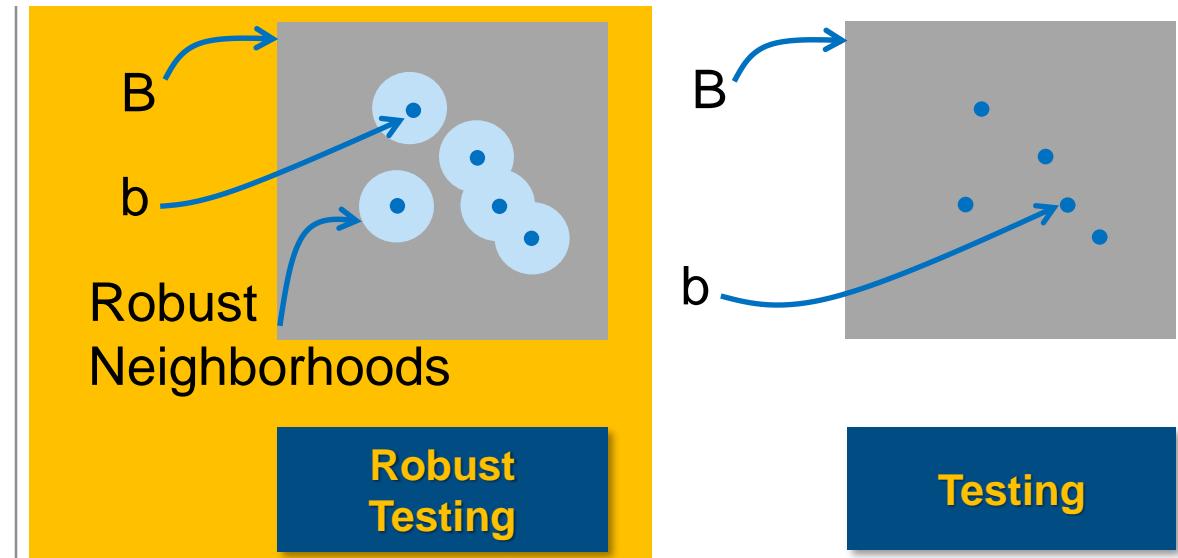


Specification Mining



Falsification

Robust testing a.k.a. simulation-based reachability analysis



Robust testing a.k.a. simulation-based reachability analysis

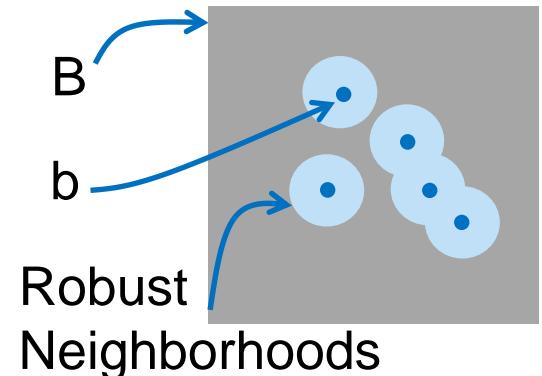
[R07] A. Rajhans, “*Development of Robust Testing Toolbox for Hybrid Systems*,” MS Thesis, University of Pennsylvania, 2007.

[DRJ13] Y. Deng, A. Rajhans, and A. A. Julius, “*STRONG: A Trajectory-Based Verification Toolbox for Hybrid Systems*,” 10th International Conference on Quantitative Evaluation of SysTems (QEST), 2013.

[DKR09] A. Donzé, B. H. Krogh, and A. Rajhans, “*Parameter Synthesis for Hybrid Systems with an Application to Simulink Models*,” 12th IEEE/ACM International Conference on Hybrid Systems: Computation and Control, 2009.

Related work by Fainekos, Pappas, Balkan, Tabuada, Zutshi, Sankaranarayanan, Kanade, Alur, ...

Bisimulation functions



Sensitivity analysis

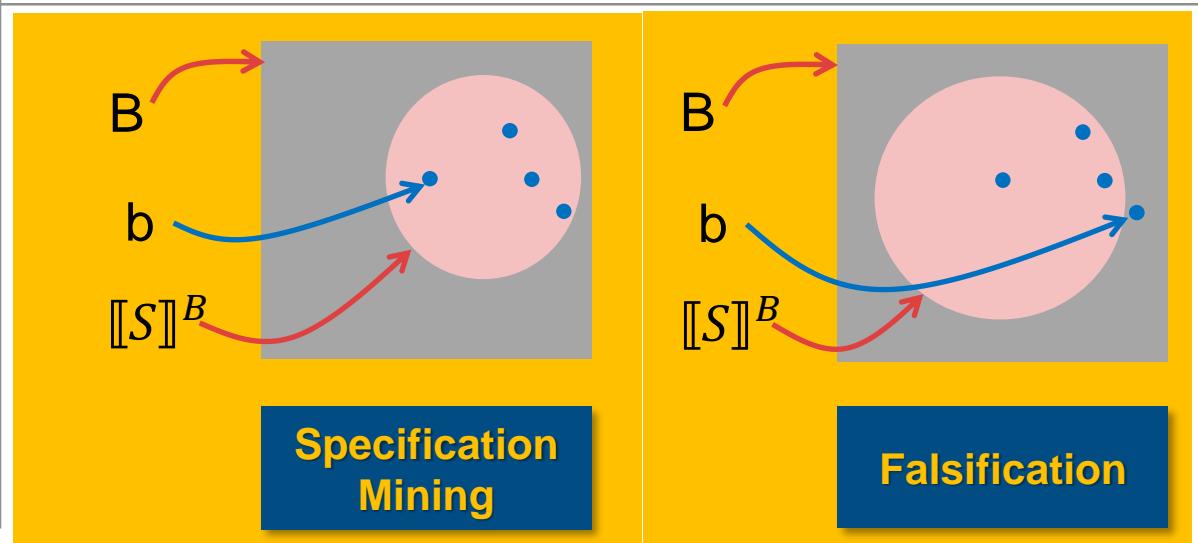
Lyapunov analysis, contraction metrics, barrier certificates, concolic testing ...

Toyota adoption a success story

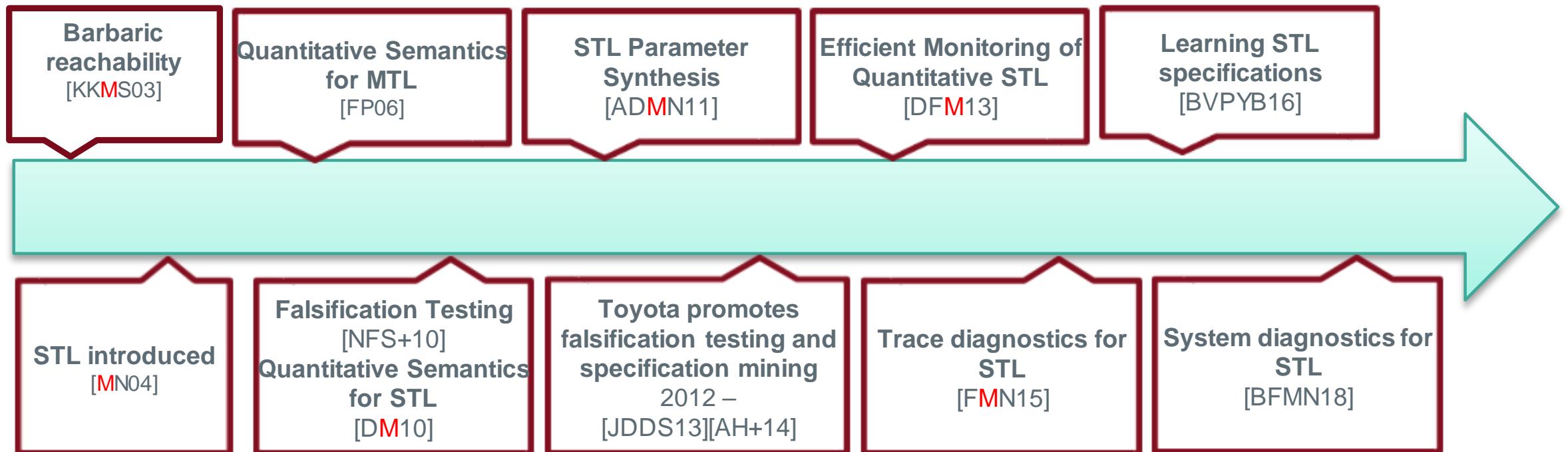
Simulation-Based Approaches for Verification of Embedded Control Systems

JAMES KAPINSKI, JYOTIRMOY V. DESHMUKH, XIAOQING JIN,
HISAHIRO ITO, and KEN BUTTS

Formalizing specifications to enable falsification

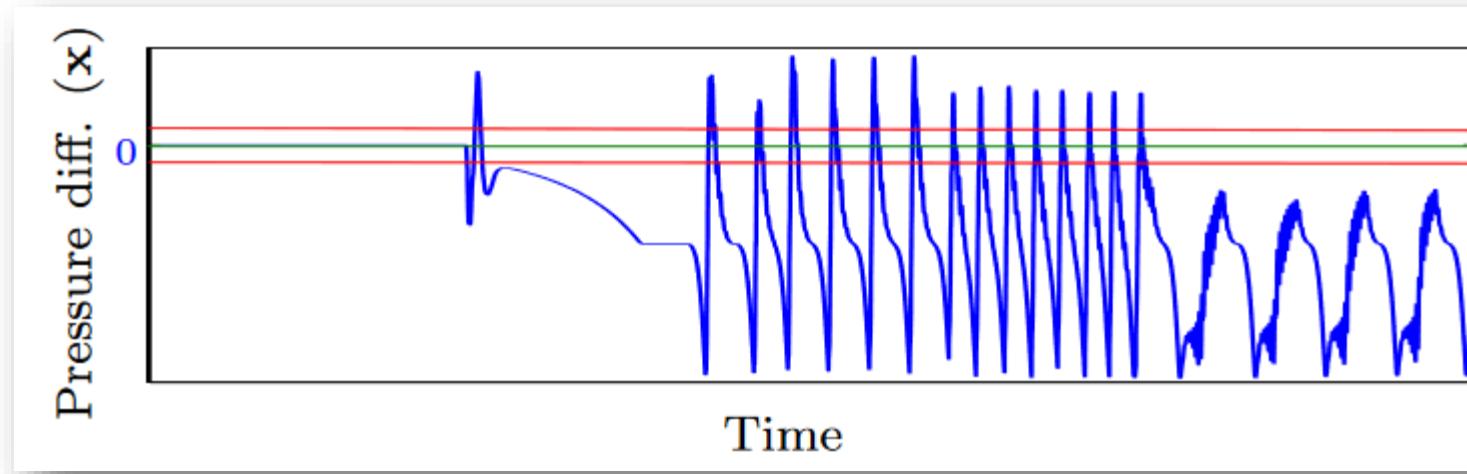


Signal Temporal Logic as a success story



Credit: Dejan Ničković (via Bruce Krogh), *Oded Maler: A memory box full of diamonds*, MT-CPS 2019.

An actual bug uncovered via falsification at Toyota



Mining Requirements from Closed-Loop Control Models

Xiaoqing Jin
Univ. of California Riverside
jinx@cs.ucr.edu

Alexandre Donzé
Univ. of California Berkeley
donze@eecs.berkeley.edu

Jyotirmoy V. Deshmukh
Toyota Technical Center
jyotirmoy.deshmukh@tema.toyota.com

Sanjit A. Seshia
Univ. of California Berkeley
sseshia@eecs.berkeley.edu

Considerations for engineering adoption of temporal logics

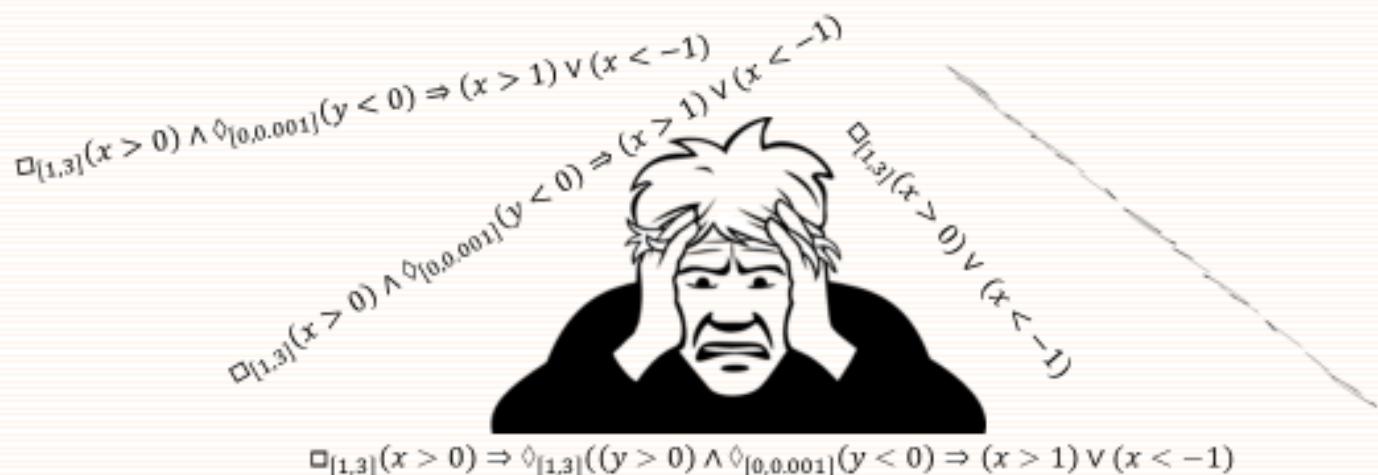
- Engineers are not logicians – logic vocabulary could be a challenge
- Simple engineering concepts may require complex logical formulas
- Multiple modeling formalisms that interact
- Multiple combinations of time/signal domains, data types, solver settings

HSCC 2015 Keynote, Jyotirmoy Deshmukh, (then) Toyota

► Automotive Industry Trends ► MBD Verification ► Techniques ► **Challenges**

Grand Challenge I: Requirement Engineering

- Key challenge for Toyota, Bosch, and others
 - How do you present requirements to control designers?
 - How do they convey their intention without using formalisms?
 - Is Temporal Logic the right requirement language?

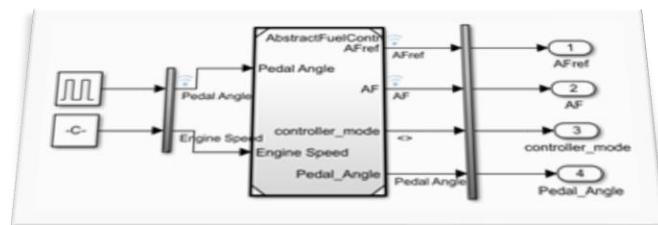


Logical and temporal assessments in Simulink Test

- Formalize and execute requirements directly as Test Assessments

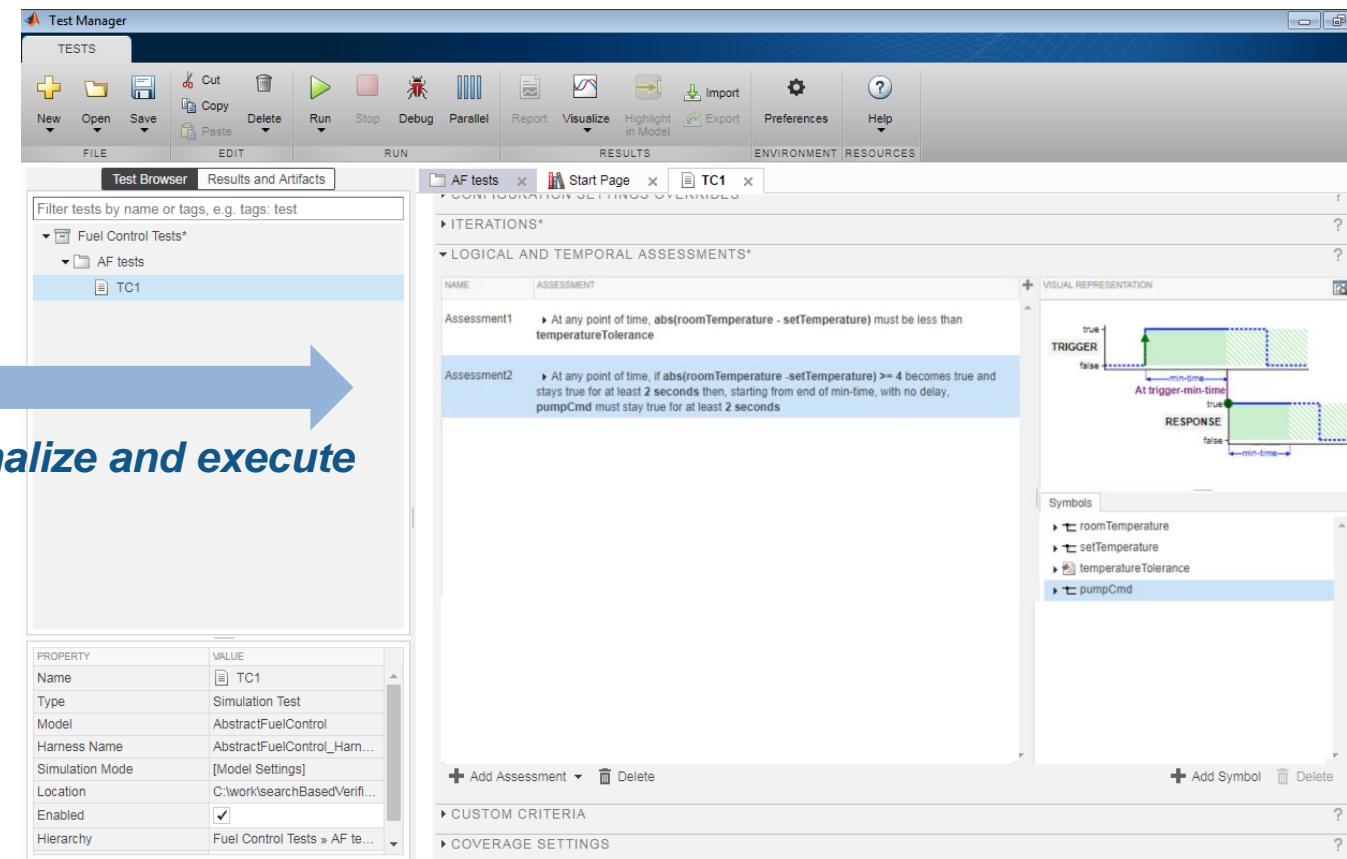
Requirements

- The difference between the room temperature and the set temperature should never exceed 6 degrees.
- If the temperature difference exceeds 4 degrees for more than 2 seconds, then the pump shall activate for at least 2 seconds



System Under Test

Formalize and execute



Authoring

Assessment

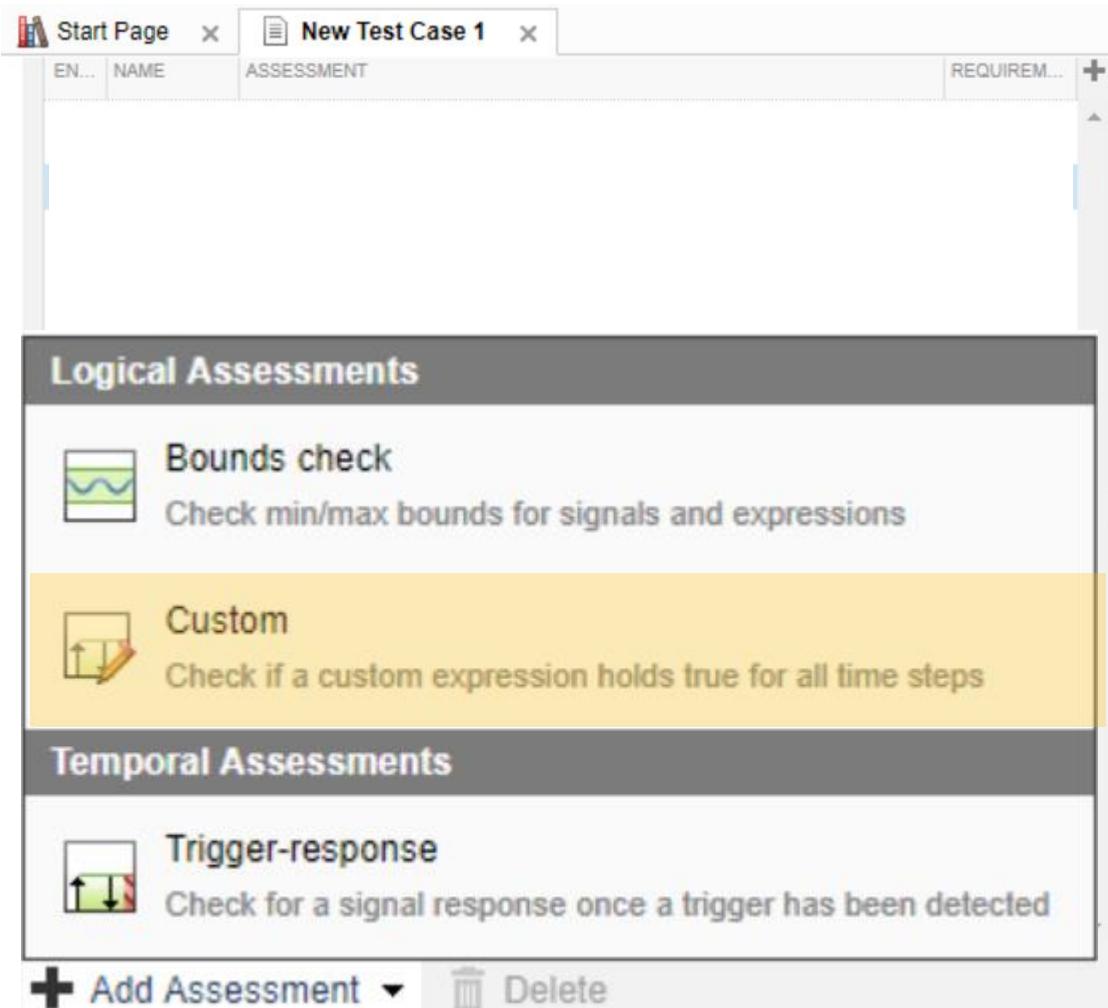
LOGICAL AND TEMPORAL ASSESSMENTS*

EN...	NAME	ASSESSMENT	REQUIREMENTS	+	VISUAL REPRESENTATION
<input checked="" type="checkbox"/>	Assessment1	<ul style="list-style-type: none"> At any point of time ... bounds-check-pattern: always inside bounds <p>signal: sig lower-bound: lb lower-bound-type: greater than upper-bound: ub upper-bound-type: less than</p>	None		
		<ul style="list-style-type: none"> At any point of time ... bounds-check-pattern: sig must be greater than lb and less than ub 		<div style="border: 1px solid #ccc; padding: 5px;"> Pattern explorer <p>New Example Passing Example <input checked="" type="checkbox"/> Dynamic bounds</p> </div>	<div style="border: 1px solid #ccc; padding: 5px;"> Pattern explorer <p>New Example Failing Example <input checked="" type="checkbox"/> Dynamic bounds</p> </div>
		<ul style="list-style-type: none"> At any point of time, sig must be greater than lb and less than ub 		<div style="border: 1px solid #ccc; padding: 5px;"> Pattern explorer <p>New Example Passing Example <input checked="" type="checkbox"/> Dynamic bounds</p> </div>	<div style="border: 1px solid #ccc; padding: 5px;"> Pattern explorer <p>Ib Exp ul Exp New Example Failing Example <input checked="" type="checkbox"/> Dynamic bounds</p> </div>
+ Add Assessment	Delete	Trigger-response Check for a signal response once a trigger has been detected			

Add Assessment **Delete**

Authoring

>> sltestmgr



$$\square_{[t_0, t_f]} \phi$$

Authoring

>> sltestmgr

The screenshot shows the MATLAB Test Manager interface. At the top, there are tabs for 'Start Page' and 'New Test Case 1'. The main area displays a table with columns for 'EN...', 'NAME', 'ASSESSMENT', and 'REQUIRE...'. A single row is present, titled 'Assessm... At any point of time ...' with 'None' under REQUIREMENT. Below this table, a section titled 'Logical Assessments' lists two items: 'Bounds check' (with a wavy line icon) and 'Custom' (with a pencil and document icon). Under 'Temporal Assessments', there is one item: 'Trigger-response' (with a signal waveform icon). At the bottom, there are buttons for '+ Add Assessment' and 'Delete'.

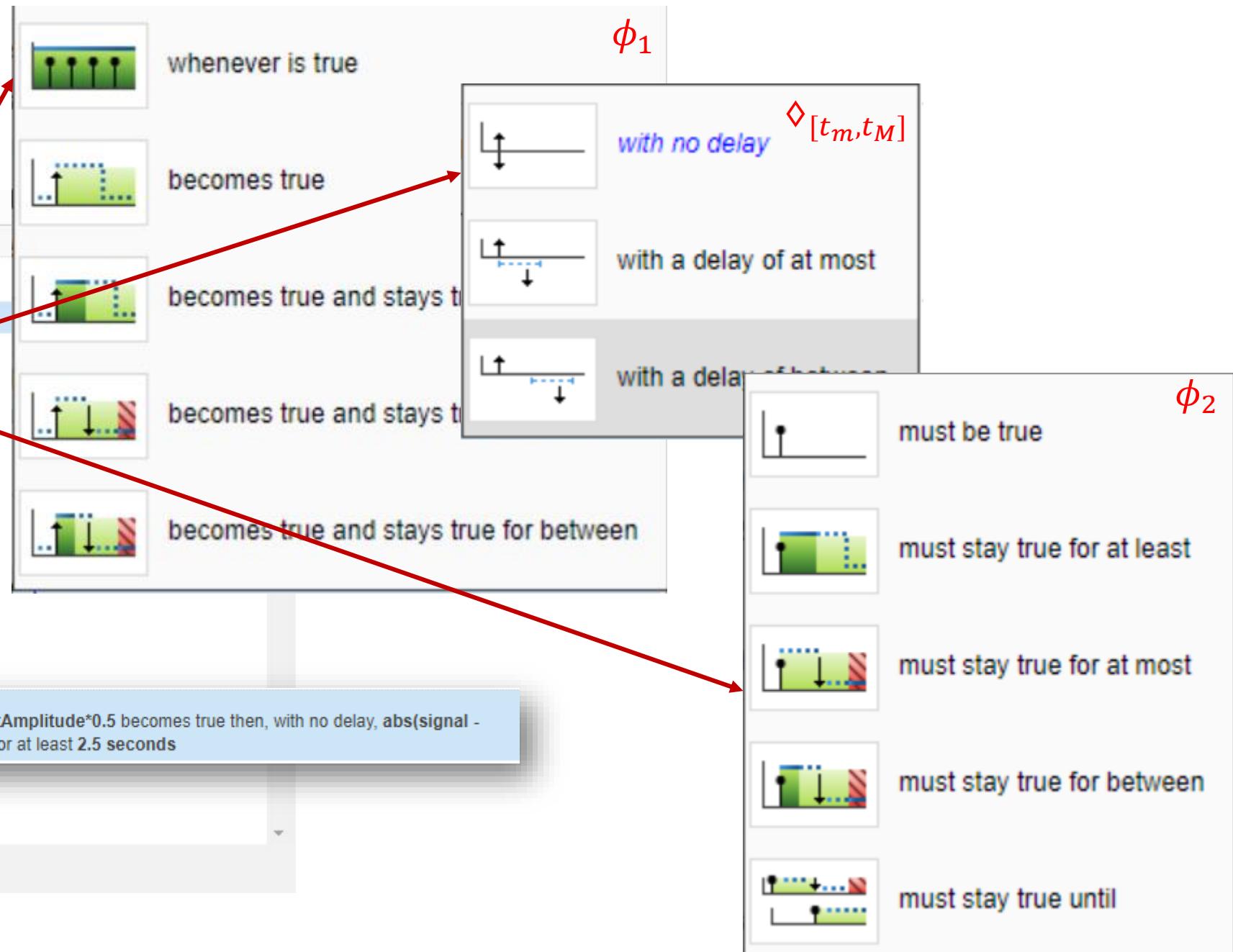
$$\square_{[t_0, t_f]} (\phi_1 \Rightarrow \diamond_{[t_m, t_M]} \phi_2)$$

Authoring

>> sltestmgr

Assessments ▾ At any point of time ...
trigger: <empty>
delay: with no delay ...
response: <empty>

$$\square_{[t_0, t_f]} (\phi_1 \Rightarrow \diamond_{[t_m, t_M]} \phi_2)$$



► At any point of time, if `driverInput > driverInputAmplitude*0.5` becomes true then, with no delay, `abs(signal - signalRef) < overshootTolerance` must stay true for at least 2.5 seconds

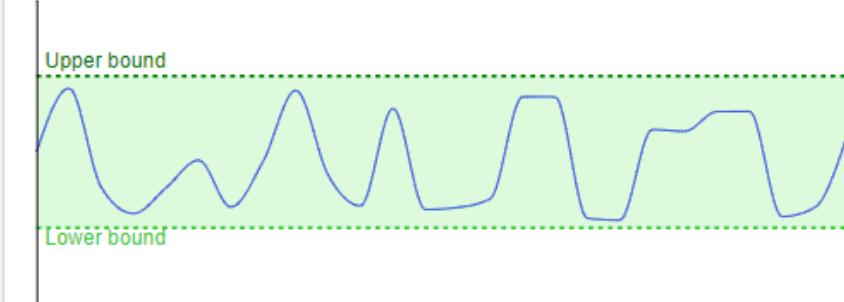
+ Add Assessment ▾ Delete

Symbol resolution and mapping

LOGICAL AND TEMPORAL ASSESSMENTS*

EN...	NAME	ASSESSMENT	REQUIREMENTS
<input checked="" type="checkbox"/>	Assessment1	At any point of time ... bounds-check-pattern: always inside bounds signal: sig lower-bound: lb lower-bound-type: greater than upper-bound: ub upper-bound-type: less than	None

VISUAL REPRESENTATION



SYMBOLS

- sig
 - Name: Sine Wave:1
 - Path: sine_wave/Sine Wave
 - Port Index: 1
 - Field/Element: <type an expression>
- lb
 - Expression: -0.4
- ub
 - Expression: 0.2

Add Assessment Delete Add Symbol Delete

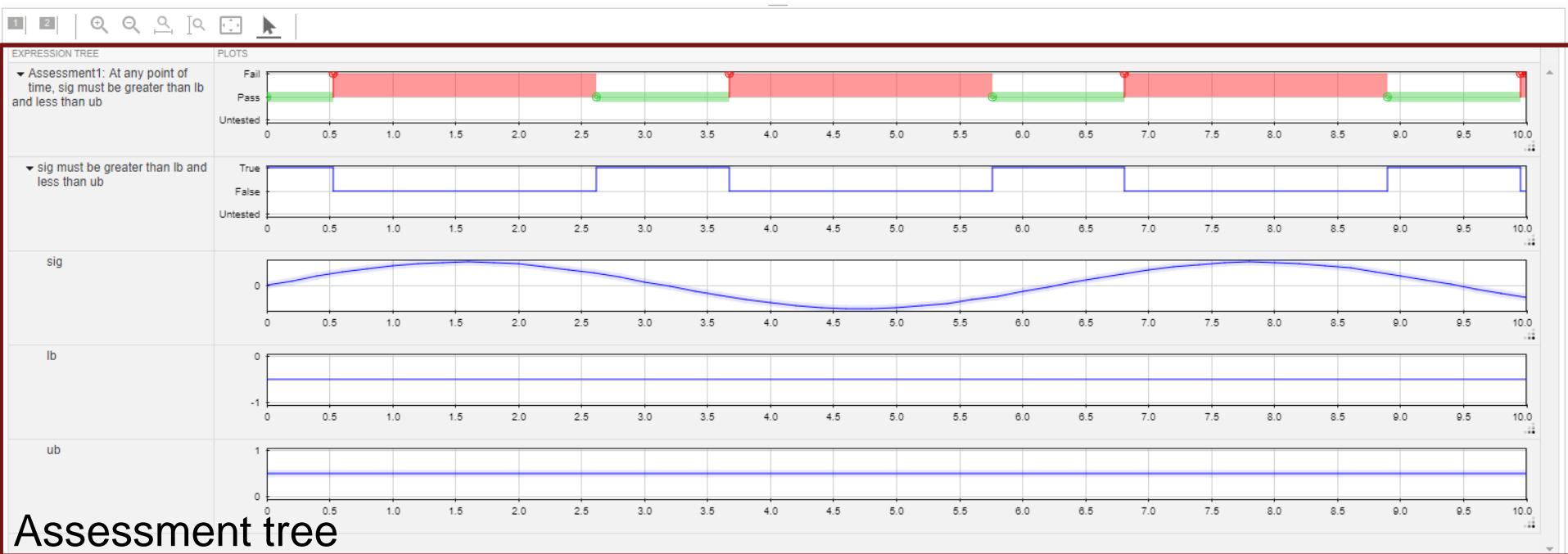
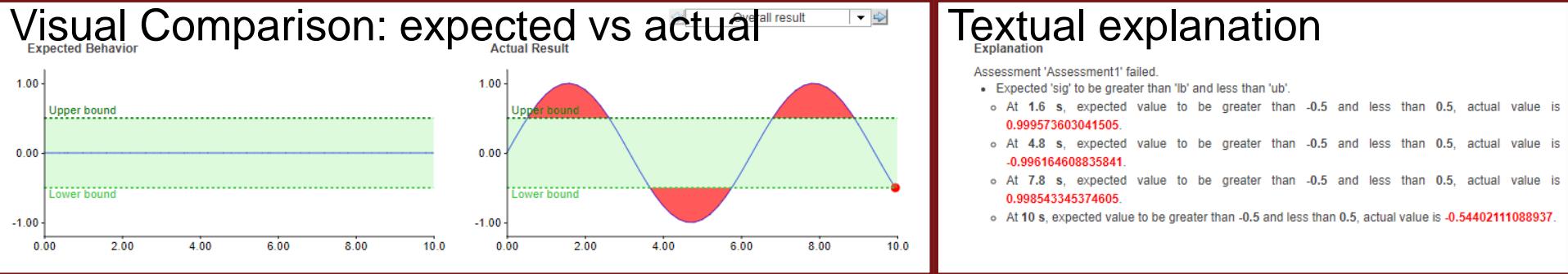
Sine Wave

sine_wave

A red dashed arrow points from the symbol 'sig' in the 'SYMBOLS' list to the 'Sine Wave' block in the model diagram.

```
graph LR; sine_wave[sine_wave] --> simout[simout]; sine_wave --> 1((1)); sine_wave --> square[square]
```

Assessment and explanation in case of failure



Expression tree

$$\Box_{[t_0, t_f]} (\phi_1 \Rightarrow \Diamond_{[0, t_M]} \phi_2)$$

EXPRESSION TREE

- Assessment1: At any point in time, if $(\text{abs}(\text{mag}) > 0.5)$ becomes true then, with a delay of at most 1.5 seconds, $(\text{abs}(\text{mag}) < 0.1)$ must stay true for at least 1 seconds
- if $(\text{abs}(\text{mag}) > 0.5)$ becomes true then, with a delay of at most 1.5 seconds, $(\text{abs}(\text{mag}) < 0.1)$ must stay true for at least 1 seconds
- $(\text{abs}(\text{mag}) > 0.5)$ becomes true
- $(\text{abs}(\text{mag}) > 0.5)$
- with a delay of at most 1.5 seconds, $(\text{abs}(\text{mag}) < 0.1)$ must stay true for at least 1 seconds
- $(\text{abs}(\text{mag}) < 0.1)$ must stay true for at least 1 seconds
- $(\text{abs}(\text{mag}) < 0.1)$

PLOTS

Tested and failed

True False Untested

Rising edge

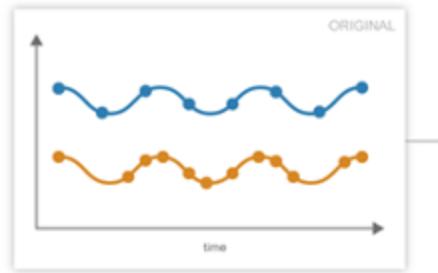
[t, T] \oplus [0, 1.5]

[t, T] \ominus [0, 1]

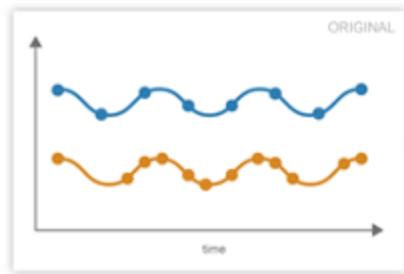
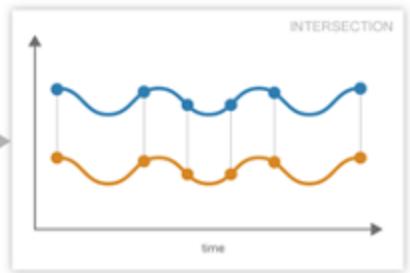
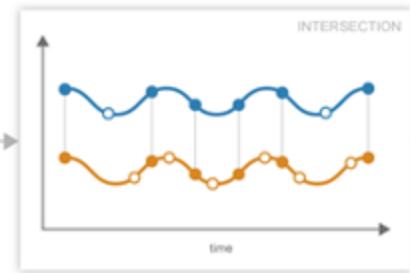
($\phi_1 \Rightarrow \phi_2$) $\equiv \neg \phi_1 \vee \phi_2$?

Did not get tested

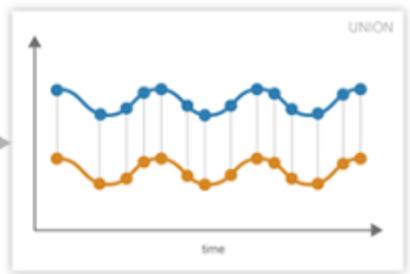
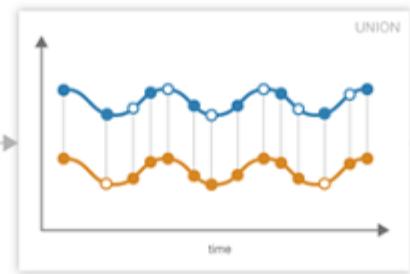
Synchronization and interpolation



Synchronize

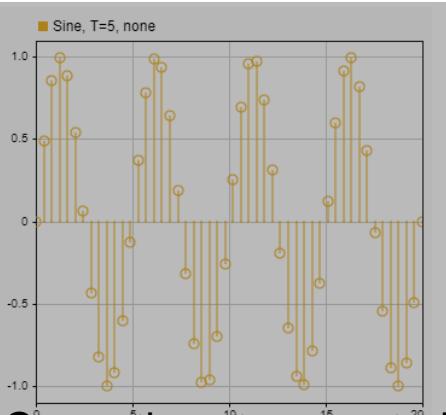
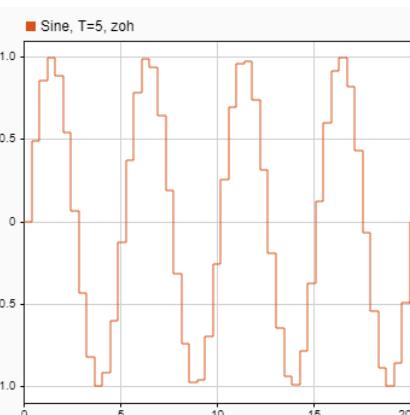
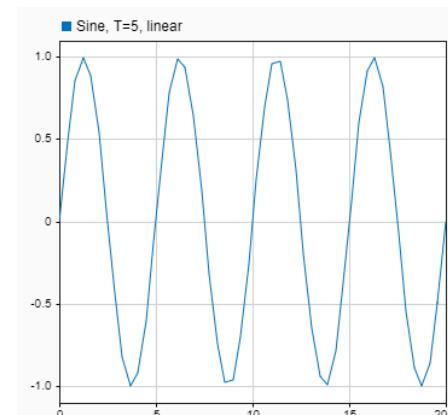


Synchronize



Research challenge: heterogeneity

- discrete and continuous time
- discrete and continuous value
STL \heartsuit LTL ?
- Needing to up/down-sample may impact frequency domain characteristics
- Dataflow domain: cannot insert or remove data points



Currently not supported

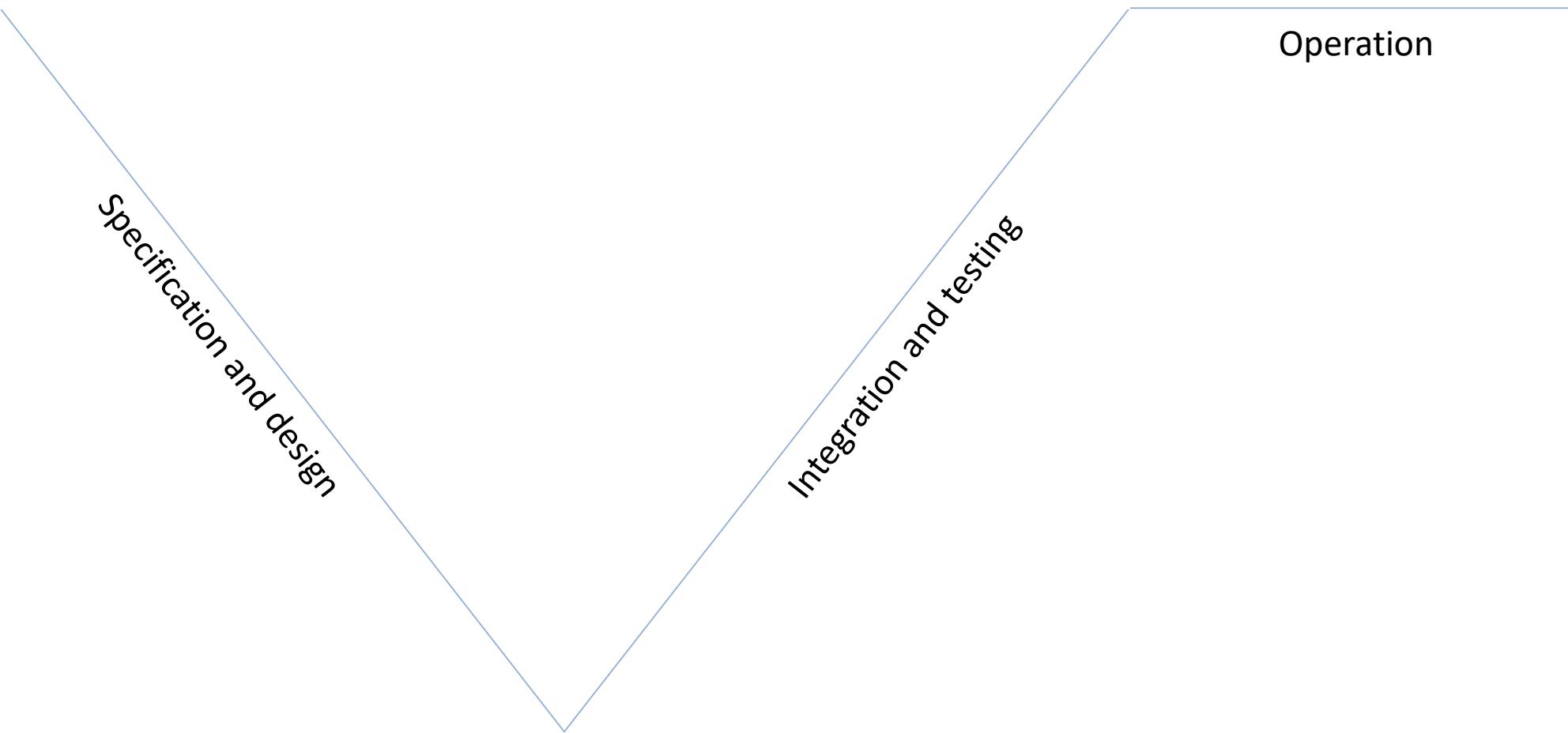
References

- [DMN19] A. Donzé and A. Rajhans, “*Tools Perspective*”, J. V. Deshmukh, O. Maler, and D. Nickovic, eds., “*Specification Formalisms for Modern Cyber-Physical Systems (Dagstuhl Seminar 19071)*”, 2019.
- [KKR19] J.-F. Kempf, Khoo Y. P., and A. Rajhans, “*Specification and Assessment of Temporal Requirements using Simulink Test*”, Fourth International Workshop on Monitoring and Testing of Cyber-Physical Systems (MT-CPS 2019), part of CPS-IoT Week 2019.
- [ABB+18] S. Anderson, et al., “*On the Use of Modeling and Simulation in Robotics*,” Workshop Report, NIST/NSF/DoD Workshop on Simulation and Machine Learning in Robotics, 2018.
- [DRJ13] Y. Deng, A. Rajhans, and A. A. Julius, “*STRONG: A Trajectory-Based Verification Toolbox for Hybrid Systems*,” 10th International Conference on Quantitative Evaluation of SysTems (QEST), 2013.
- [DKR09] A. Donzé, B. H. Krogh, and A. Rajhans, “*Parameter Synthesis for Hybrid Systems with an Application to Simulink Models*,” 12th IEEE/ACM International Conference on Hybrid Systems: Computation and Control, 2009.
- [R07] A. Rajhans, “*Development of Robust Testing Toolbox for Hybrid Systems*,” MSE Thesis, University of Pennsylvania, 2007.

Outline

- Cyber-physical systems: a feature classification
- “Runtime” verification at design time: simulation-based approaches
- Runtime analysis at operation time: From CPS to IoT and Digital Twins
- Challenges and future outlook

Models are useful in both design and operation

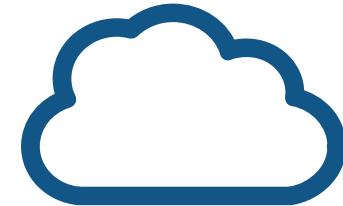


Challenges in the Operation and Design of Intelligent Cyber-Physical Systems, S. Castro, P.J. Mosterman, A.H. Rajhans, and R.G. Valenti, book chapter, *Complexity Challenges in Cyber Physical Systems: Using Modeling and Simulation (M&S) to Support Intelligence, Adaptation and Autonomy*, S. Mittal and A. Tolk, eds., Wiley, 2019.

Internet of Things topology



Thing



Internet

Internet of Things topology

mytoaster
@mytoaster

Social Networking for your Toaster, The Internet of Things (#iot) powered by @ThingSpeak, created by @scharler

- 📍 Pittsburgh, PA
- 🔗 nothans.com/social-network...
- 📅 Joined December 2008
- 🖼 Photos and videos

CheerLights
@cheerlights

CheerLights is an #internetofthings project by @scharler to synchronize lights to the same color at the same time all around the world. #iot #thingspeak

- 📍 Pittsburgh, PA
- 🔗 cheerlights.com
- 📅 Joined November 2011
- 🖼 35 Photos and videos

Internet

Tweets 1,055 Following 455 Followers 1,203 Likes 1,284

Tweets **Tweets & replies** **Media**

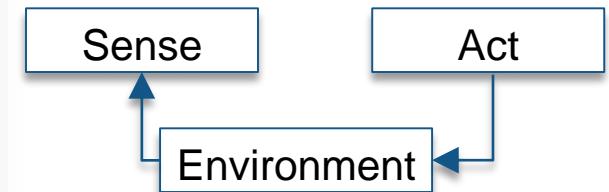
CheerLights @cheerlights · Jul 16
JavaScript CheerLights Lamp by @LizMyers cheerlights.com/javascript-che...

CheerLights Retweeted
Hans Scharler @scharler · Jun 27
Real-time Colors on CheerLights using the MQTT Protocol nothans.com/real-time-color...

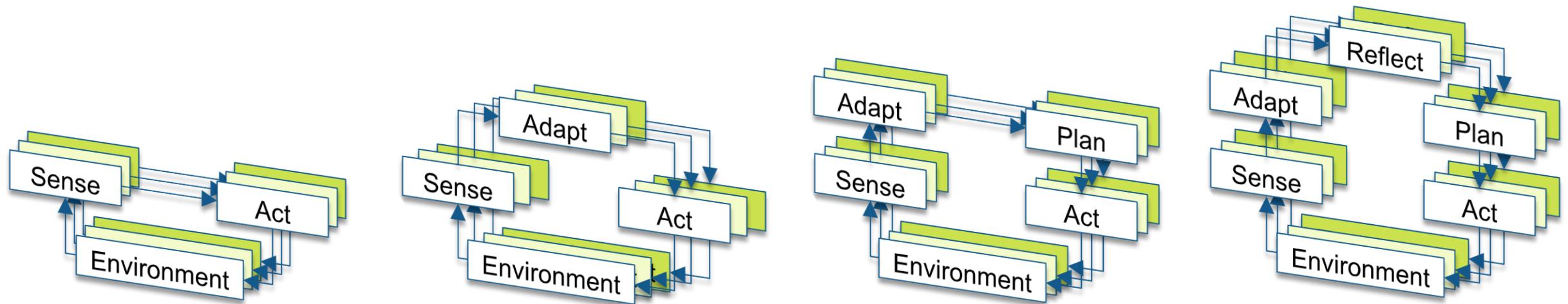
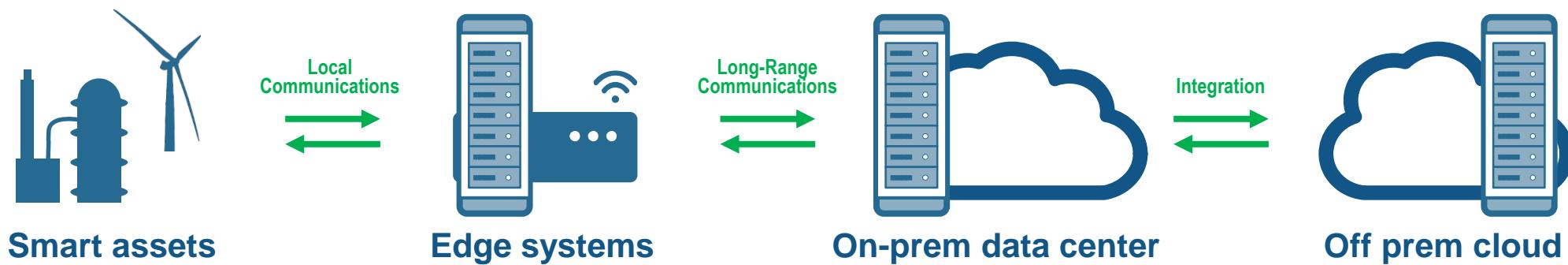
Cheerlights



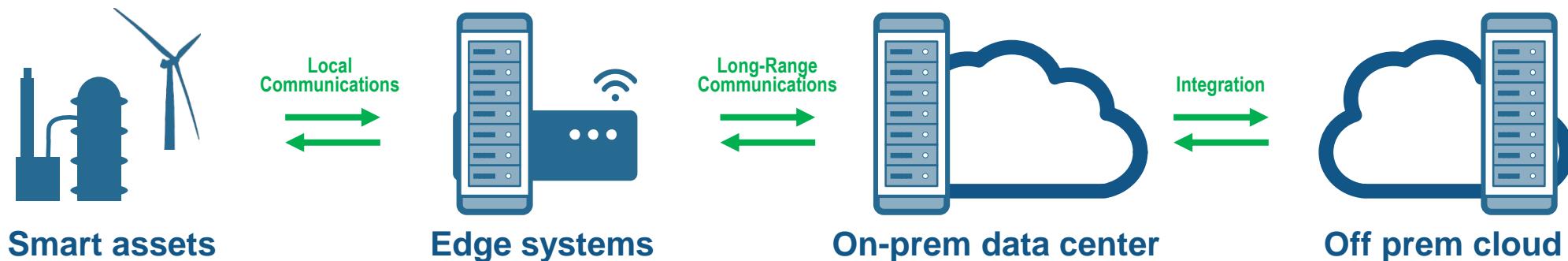
Internet



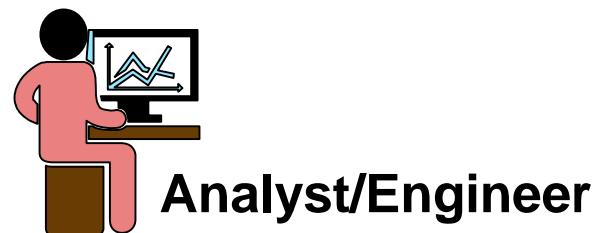
Industrial Internet of Things topology – Enterprise level operations



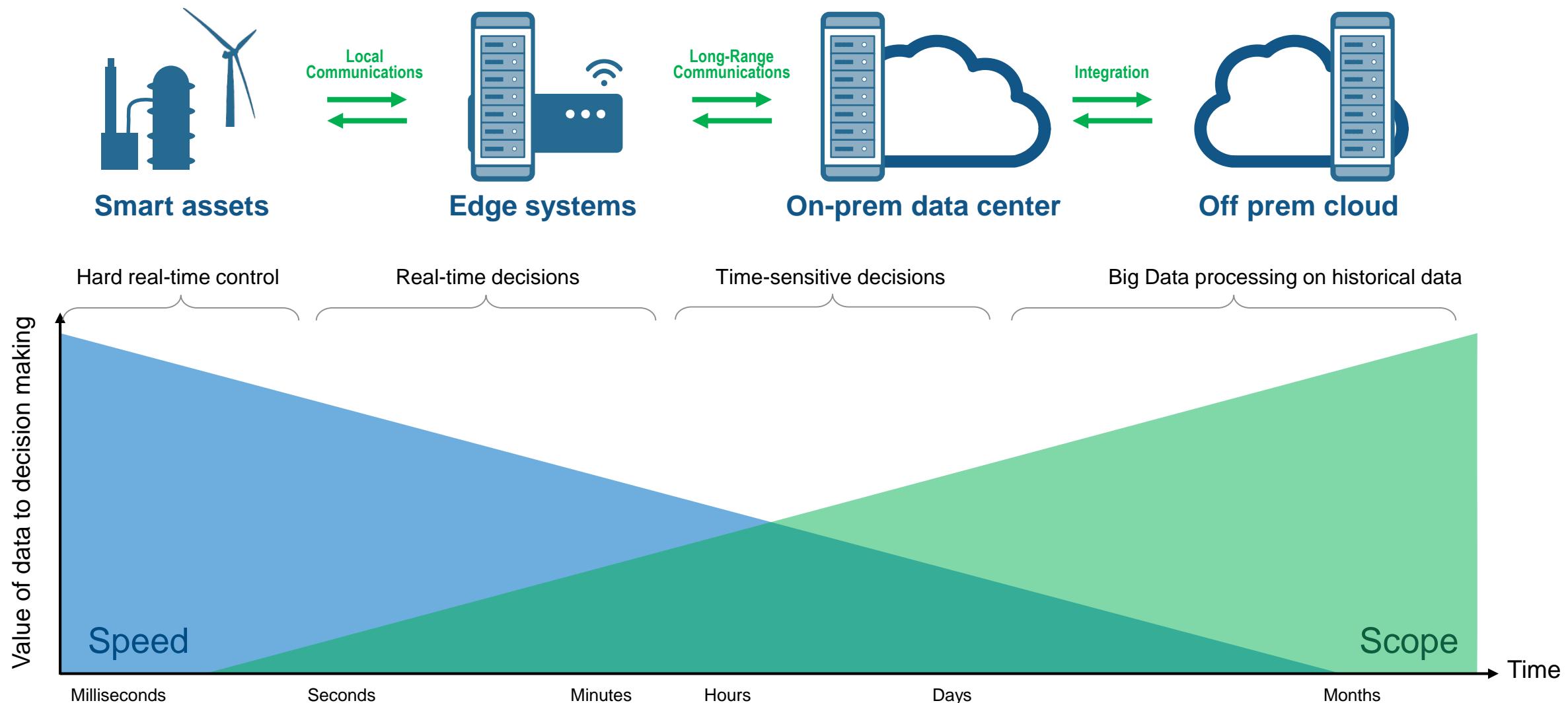
A complex collection of tools, platforms, and protocols



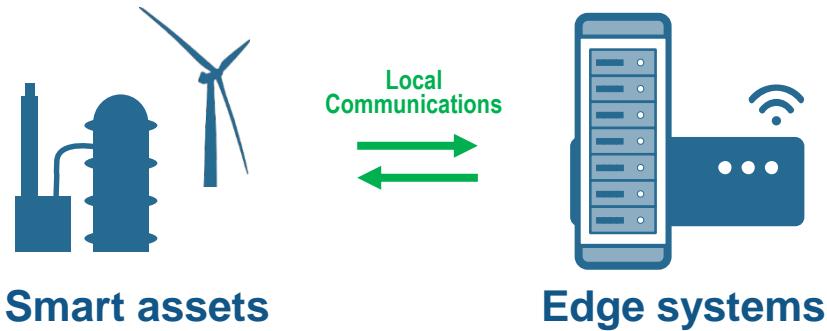
- Chipmakers
- Transport protocols
- Operating systems running on edge or on-premises
- Cloud providers
- Streaming protocols for getting data in and out of the cloud platforms
- Services for managing data, timing, and other Industrial IoT requirements
- Dashboard tools for visualizing information in any area of the landscape



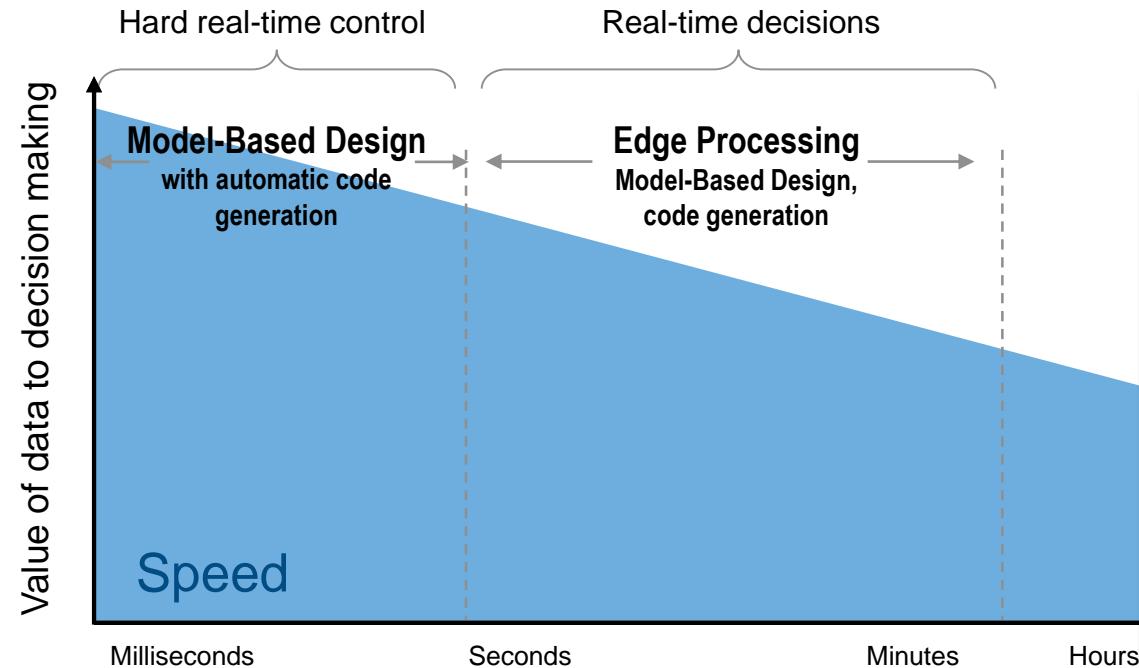
Applications at the Asset, the Edge, or Operational Technology Platform



Development for Fast and Highly-Deterministic Systems



- Compute limited but have a choice
- Data usually as a memory read
- Product design focus



Model-Based Design

Multi-domain system modeling

Parameter estimation

Automatic code generation

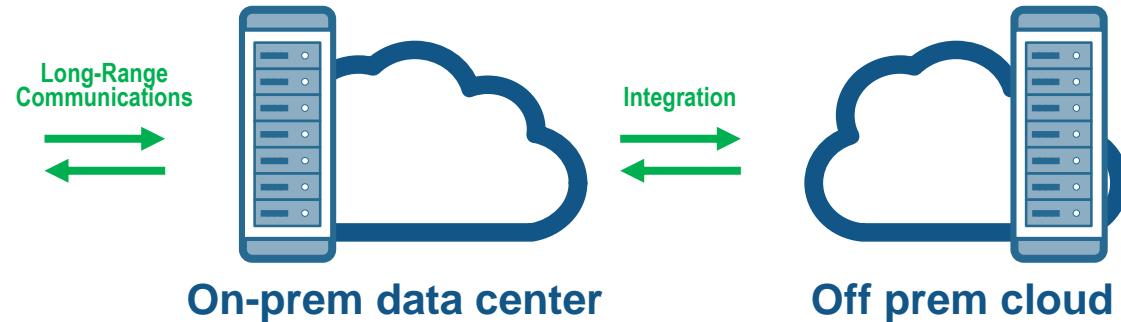
CODE GENERATION

C, C++
VHDL, Verilog
Structured Text
MCU
DSP
FPGA
ASIC
PLC

The central part of the slide shows a screenshot of the MATLAB/Simulink environment. It displays a multi-domain model of a wind turbine system. Below the screenshot, four main features of Model-Based Design are listed: Multi-domain system modeling, Parameter estimation, and Automatic code generation. The 'Automatic code generation' section includes a table of supported languages and hardware platforms.

Development to OT/IT On-Prem and in Cloud

- Compute abundant but less control
- Data access as streaming messages
- Service focus

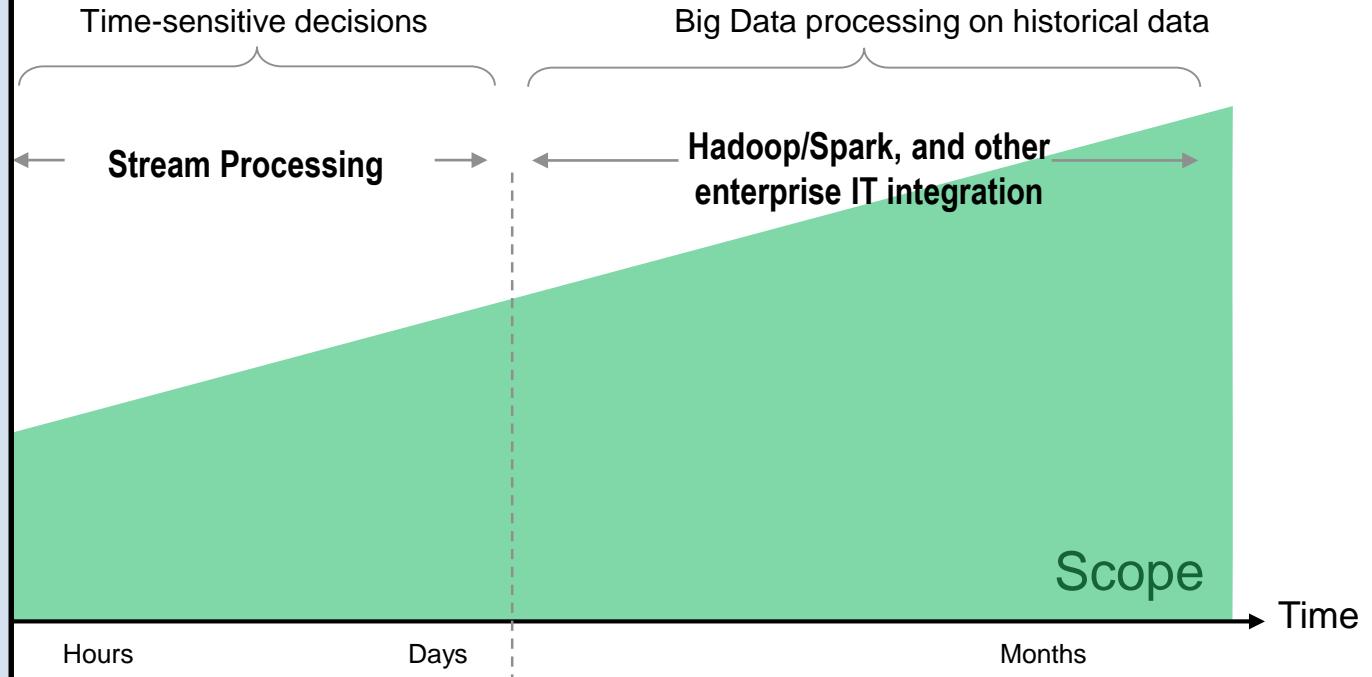


On-prem data center

Off-prem cloud

The collage includes:

- Variety and Volumes of Data:** A 3D bar chart showing the probability of arrival delays (1 hour or greater) by month and arrival delay (in minutes).
- Optimization:** A screenshot of a software interface showing multiple plots and data points, likely related to optimization algorithms.
- Machine Learning and Deep Learning:** A neural network diagram showing inputs X_1, X_2, X_3 being processed through layers of neurons and fully connected (FC) layers to produce a "Probability" output.
- Enterprise system integration, (on-prem/cloud):** An illustration of a computer monitor connected via a network to a cloud icon containing server racks.





MathWorks Cloud

MathWorks Cloud provides access to MATLAB and other products and services via a managed cloud infrastructure. MATLAB can be run in a web browser or installed on any software. MathWorks Cloud gives you the ability to store, analyze, and share data across devices, use them to teach, and perform analytics for a variety of applications.

Learn more about hosting

Public Clouds

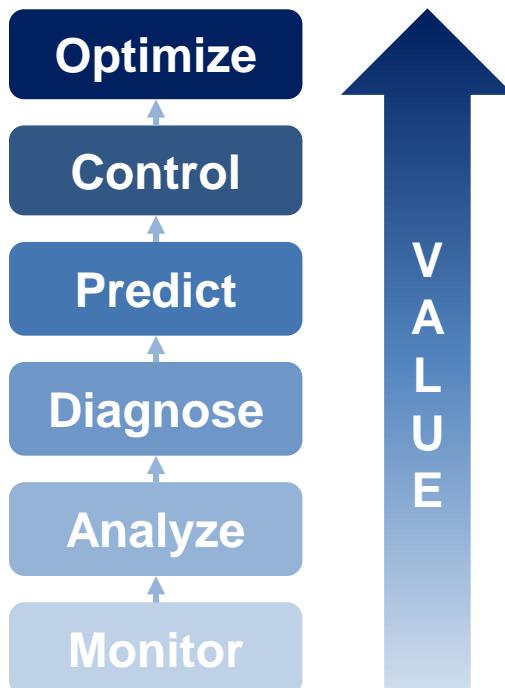
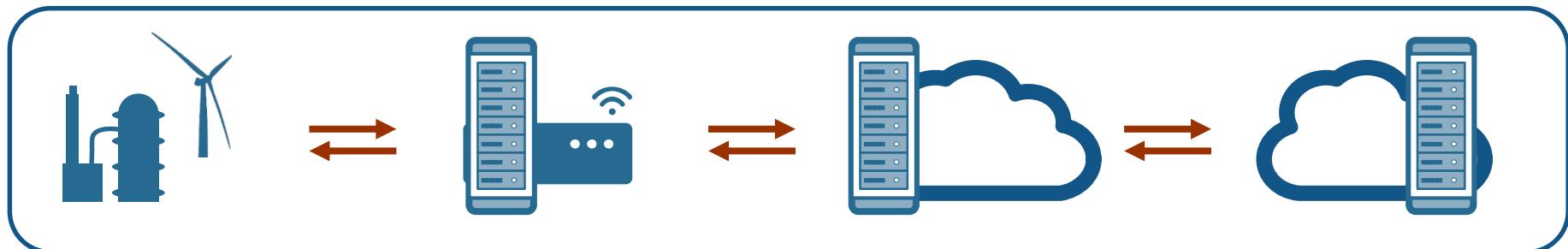
Use MATLAB on virtual machines in public cloud environments like Amazon Web Services (AWS) and Microsoft Azure. These vendors provide access to on-demand computing resources. They also offer wide-ranging, prebuilt services for data storage, data streaming, elastic scaling, load balancing, security, and more.

If you are not a cloud expert, or if you want a head start, use a MathWorks published [reference architecture](#). Templates in these reference architectures automatically create and configure the cloud infrastructure for running MATLAB. You can also adapt or extend the reference architectures to better meet your specific needs.

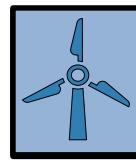
Learn more about running MATLAB and other products on:

[AWS](#)[Azure](#)[Other Clouds](#)

Digital Twin



Create computational model of asset in operation



- Data-driven (MATLAB) or first-principles (Simulink) models
- Reuse models from development process (e.g. MBD)
- Kept up-to-date during asset operation (e.g. aging, wear, environment)

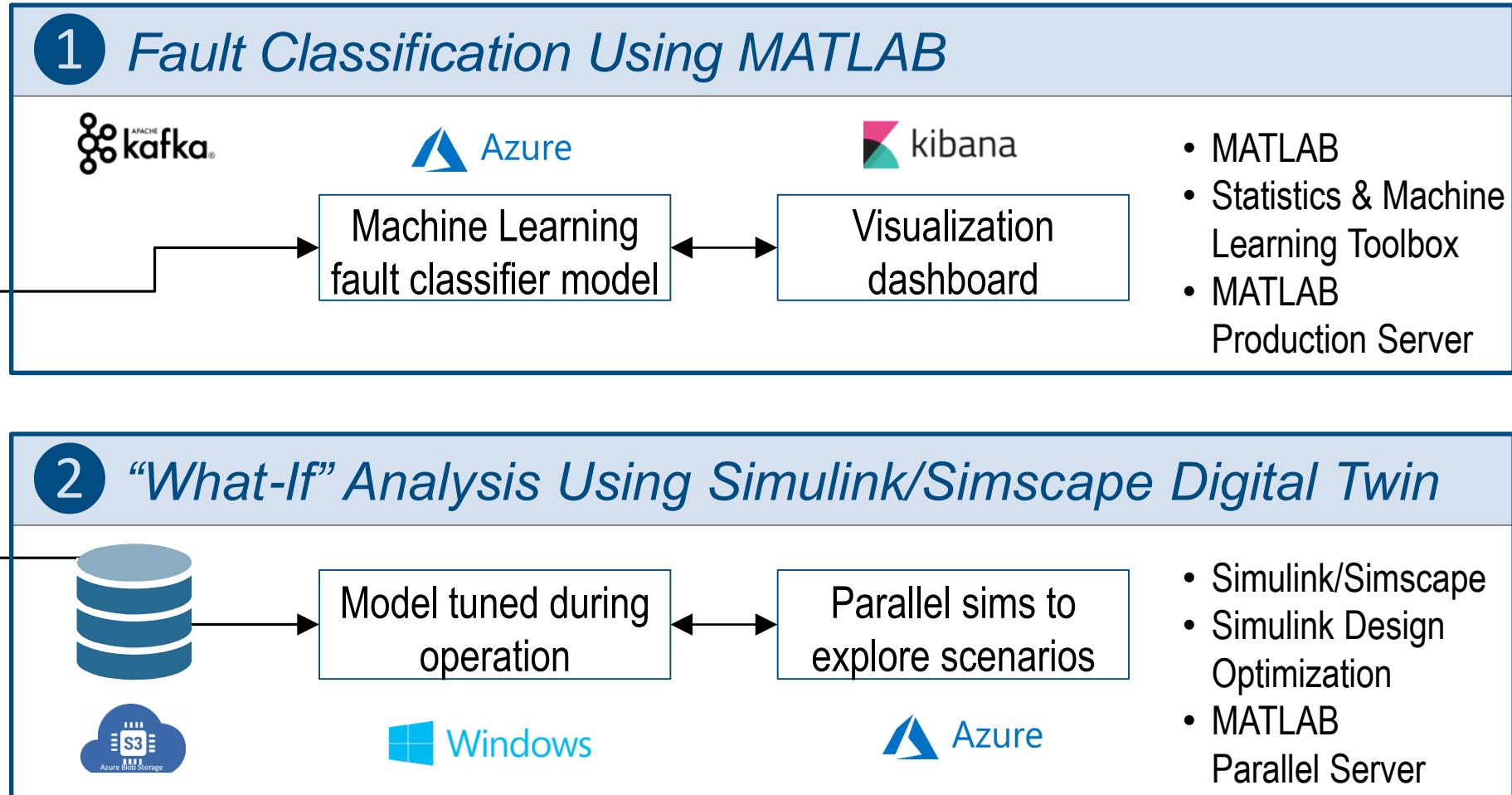
Use the computational model (digital twin) during operation

- Optimize fleet or system behavior
- Calculate control setpoints or parameters
- Predict future behavior or events

Reference example



Triplex Pump



Research Connections

2018 Annual American Control Conference (ACC)
June 27–29, 2018, Wisconsin Center, Milwaukee, USA

Coordinated Control of Wind Turbine Generator and Energy Storage System
for Frequency Regulation under Temporal Logic Specifications

Zhe Xu, Agung Julius and Joe H. Chow

IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING

Advisory Temporal Logic Inference and Controller Design for Semiautonomous Robots

Zhe Xu[✉], Student Member, IEEE, Sayan Saha[✉], Student Member, IEEE,
Botao Hu[✉], Student Member, IEEE, Sandipan Mishra, Member, IEEE,
and A. Agung Julius[✉], Member, IEEE

2017 IEEE 56th Annual Conference on Decision and Control (CDC)
December 12–15, 2017, Melbourne, Australia

Provably Correct Design of Observations for Fault Detection with Privacy Preservation

Zhe Xu, Sayan Saha and Agung Julius

Verification of Hybrid Automata Diagnosability with Measurement Uncertainty

Yi Deng, Alessandro D’Innocenzo, Maria D. Di Benedetto,
Stefano Di Gennaro, and A. Agung Julius

2018 Annual American Control Conference (ACC)
June 27–29, 2018, Wisconsin Center, Milwaukee, USA

Robust Temporal Logic Inference for Hybrid System Observation—An Application on Occupancy Detection of Smart Buildings

Zhe Xu, Yi Deng and Agung Julius

Abstract—In modern smart buildings modeled as hybrid systems, occupancy detection can be cast as observing the discrete states of a hybrid system using the available discrete and continuous system outputs. In this paper, we present a method to construct observers of the hybrid system to distinguish between different locations of the hybrid system by inferring metric temporal logic (MTL) formulae from the simulated trajectories. We first approximate the system behavior by simulating finitely many trajectories with time-robust tube segments around them. These time-robust tube segments account for both spatial and temporal uncertainties that exist in the hybrid system with initial state variations. The inferred MTL formulae classify different time-robust tube segments and thus can be used for classifying the hybrid system behaviors in a provably correct fashion. We implement our approach on a model of a smart building tested to distinguish two cases of room occupancy.

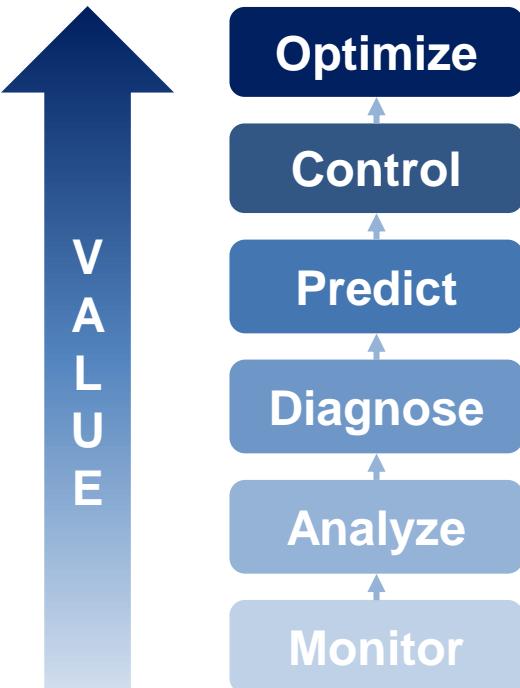
for hybrid system occupancy detection. We mainly focus on distinguishing between different occupancy states and observing the location of the modeled hybrid system at any time. For the learning aspect, there has been a growing interest in learning (inferring) dense-time temporal logic formulae from system trajectories [4], [5], [6], [7], [18], [9], [10]. We infer dense-time temporal logic formulae from the temperature and humidity sensor data as dense-time temporal logics can effectively capture the time-related features in the transient period when people enter a room. In the meantime, we also utilize the model information so that the MTL formula that classifies the finite trajectories we simulated (or gathered) also classifies the infinite trajectories that differ from the simulated trajectories by a small margin in both space and time. In our previous work in [11], we

ete-time temporal logic specifications, the system as a Markov Decision Process (MDP). The control design is transformed into a controller strategy that maximizes a sequence of states in the specification [8]. For dense-time temporal logic, the system can be abstracted [10] and the design process is after the timed automaton

controller synthesis approach [11]. The specifications with stochastic environments are converted into a controller for the trajectory of the process with robustness

tion

gical advances in robotic systems, the paradigm of robots is gradually shifting from autonomy or human–robot interaction platforms to adversarial environment, robots with all the necessary tasks within specified regions. To improve the situations, we can utilize the robot in successful or failed operations. For example, to reach the goal region around an obstacle with the initial state, the robot is moving obstacle, but if the moving obstacle is known, in this case, we can record the trajectories in a



References

- [RD19] A. Rajhans and D. Lluch, “*A Digital Twin Approach to Online Monitoring in Industrial Internet of Things Applications*”, Fourth International Workshop on Monitoring and Testing of Cyber-Physical Systems (MT-CPS 2019), part of CPS-IoT Week 2019
- Steve Miller (2019). Predictive Maintenance in Hydraulic Pump (<https://www.mathworks.com/matlabcentral/fileexchange/65605-predictive-maintenance-in-hydraulic-pump>), MATLAB Central File Exchange. Retrieved October 9, 2019.
- <https://www.mathworks.com/cloud.html>
- <https://github.com/mathworks-ref-arch>

In summary

- Cyber-physical systems continue to gain intelligence and autonomy
- CPS are open, interconnected, and change after deployment
- Formal specification and simulation-based approaches fill an important scalability gap w.r.t formal verification
- Model-Based Design approaches are being supplanted by model-based operation
- Scalability to enterprise-level system will be the value driver

Thank you!

- Thank you to the RV organizers, Leo, and Bernd
- Dagstuhl on Specification and Verification of CPS
- Jean-Francois Kempf, Khoo Yit Phang, Isaac Ito, and team
- Terri Xiao, Dan Lluch, Jim Tung, Pieter Mosterman, and team

