Value Drivers in a Changing Landscape of Modeling & Simulation

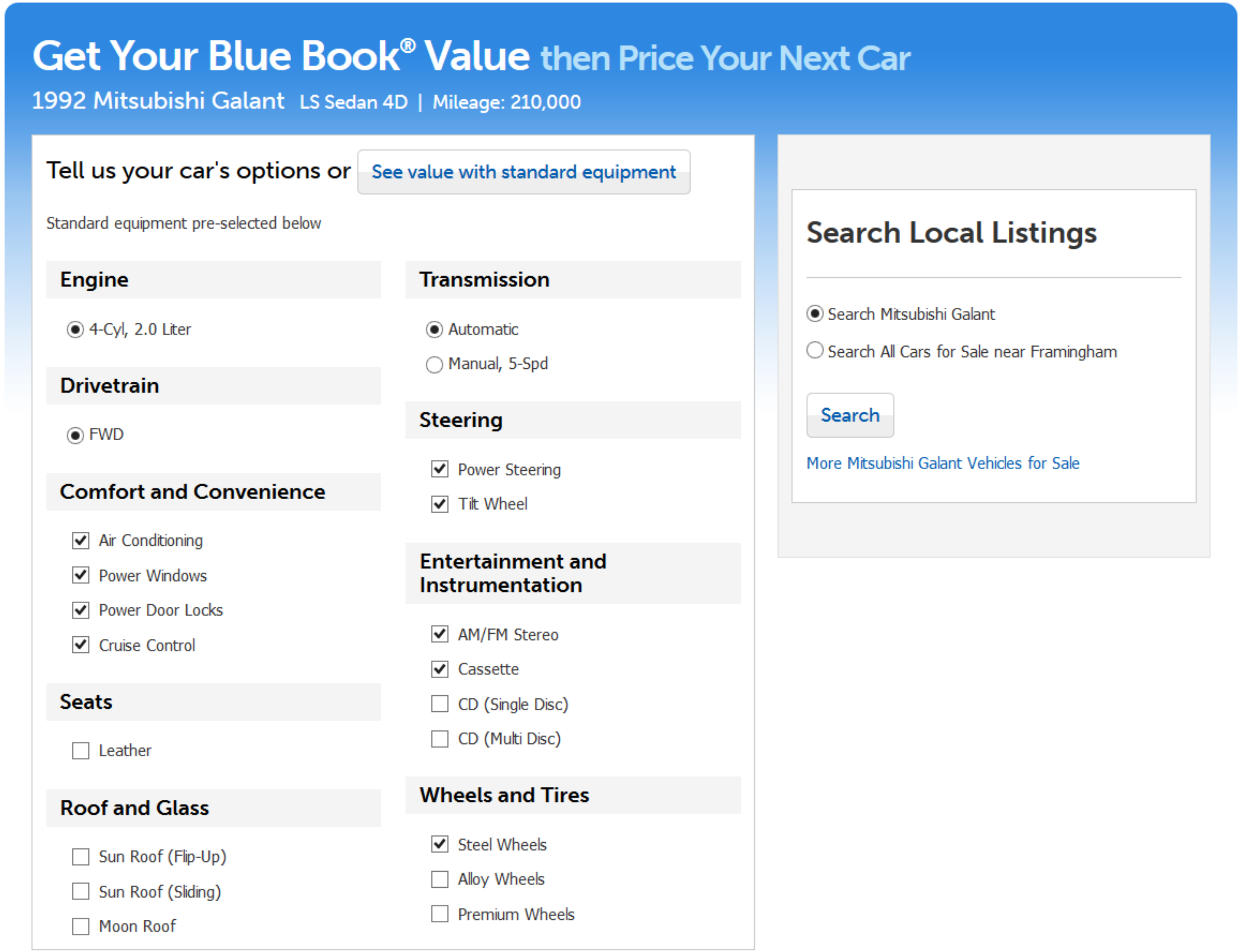
Pieter J. Mosterman

Chief Research Scientist, Director
Advanced Research & Technology Office (MARTO)
MathWorks

Adjunct Professor
School of Computer Science
McGill

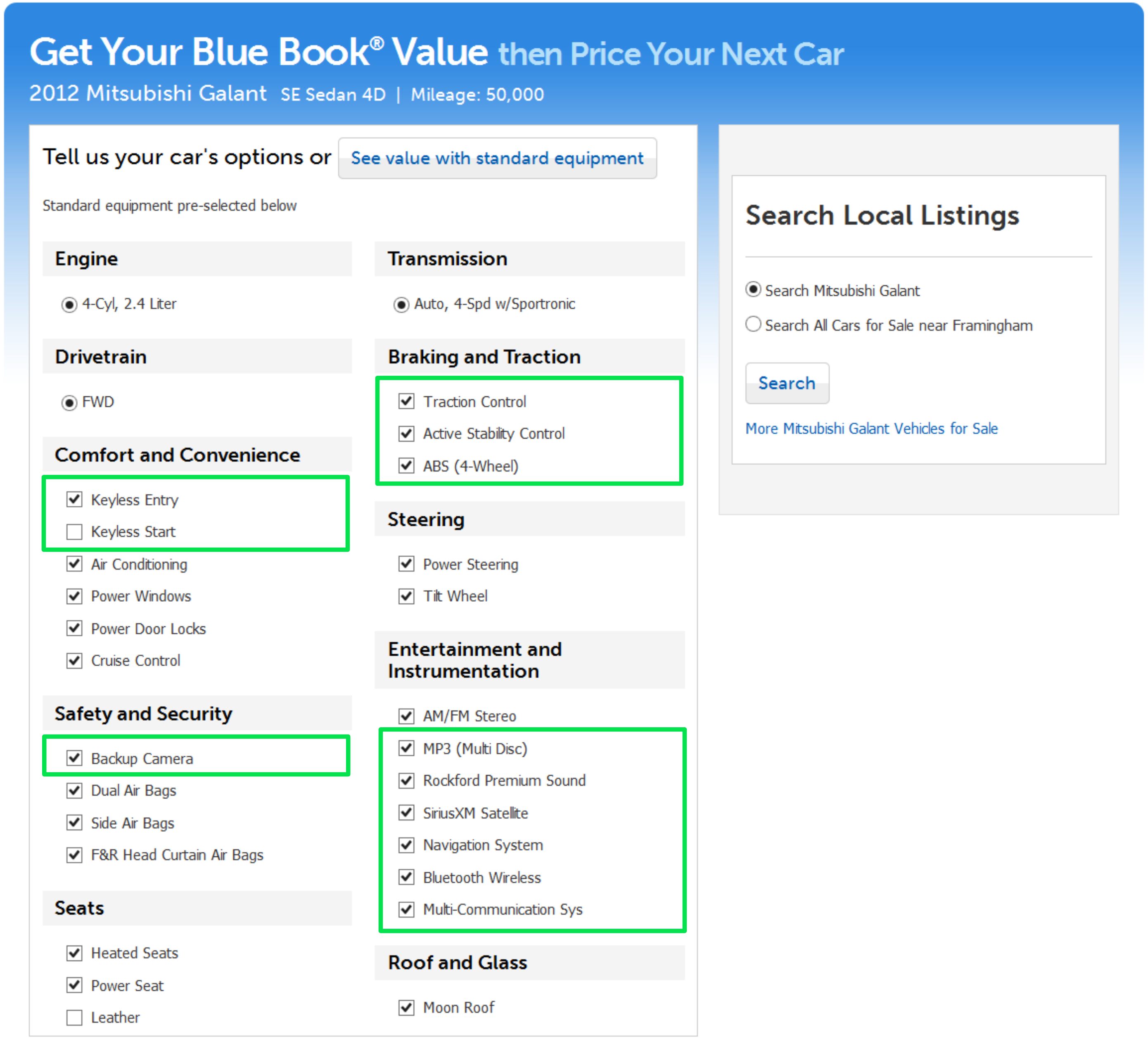
© 2016 The MathWorks, Inc.

$1,052


Get Your Blue Book® Value then Price Your Next Car
1992 Mitsubishi Galant LS Sedan 4D | Mileage: 210,000
Tell us your car's options or
See value with standard equipment
Standard equipment pre-selected below
Engine
4-Cyl, 2.0 Liter
Drivetrain
FWD
Comfort and Convenience
Air Conditioning
Power Windows
Power Door Locks
Cruise Control
Seats
Leather
Roof and Glass
Sun Roof (Flip-Up)
Sun Roof (Sliding)
Moon Roof
Transmission
Automatic
Manual, 5-Spd
Steering
Power Steering
Tilt Wheel
Entertainment and Instrumentation
AM/FM Stereo
Cassette
CD (Single Disc)
CD (Multi Disc)
Wheels and Tires
Steel Wheels
Alloy Wheels
Premium Wheels
Search Local Listings
Search Mitsubishi Galant
Search All Cars for Sale near Framingham
Search
More Mitsubishi Galant Vehicles for Sale

$11,221

Moore’s Law



$35,000

"When one car learns something, the whole fleet learns it"

Moore's Law

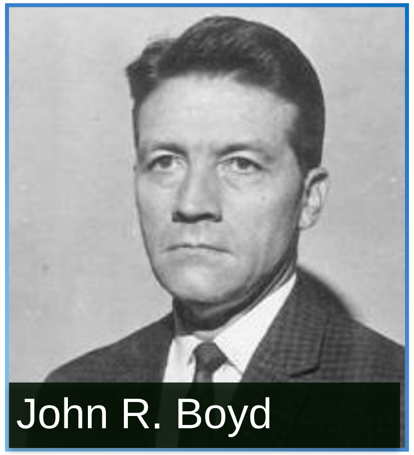
Metcalfe's Law

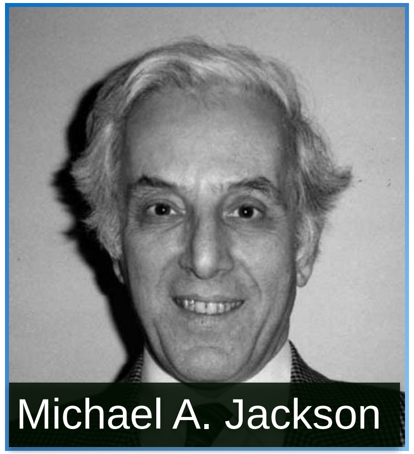
Machines are connecting and collaborating

Where can we have impact, which solutions are needed, what challenges these solutions, and how can we overcome the challenges?

A smart emergency response system

A smart emergency response system → A feature classification → Evolving architectures → Future value drivers



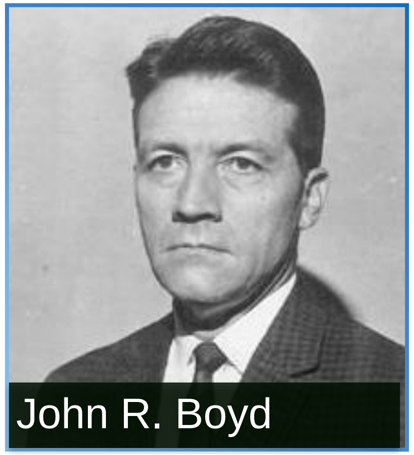


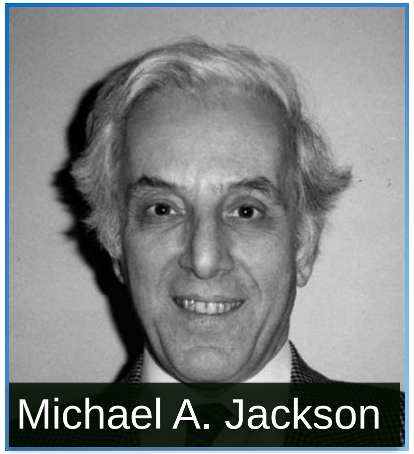
A smart emergency response system

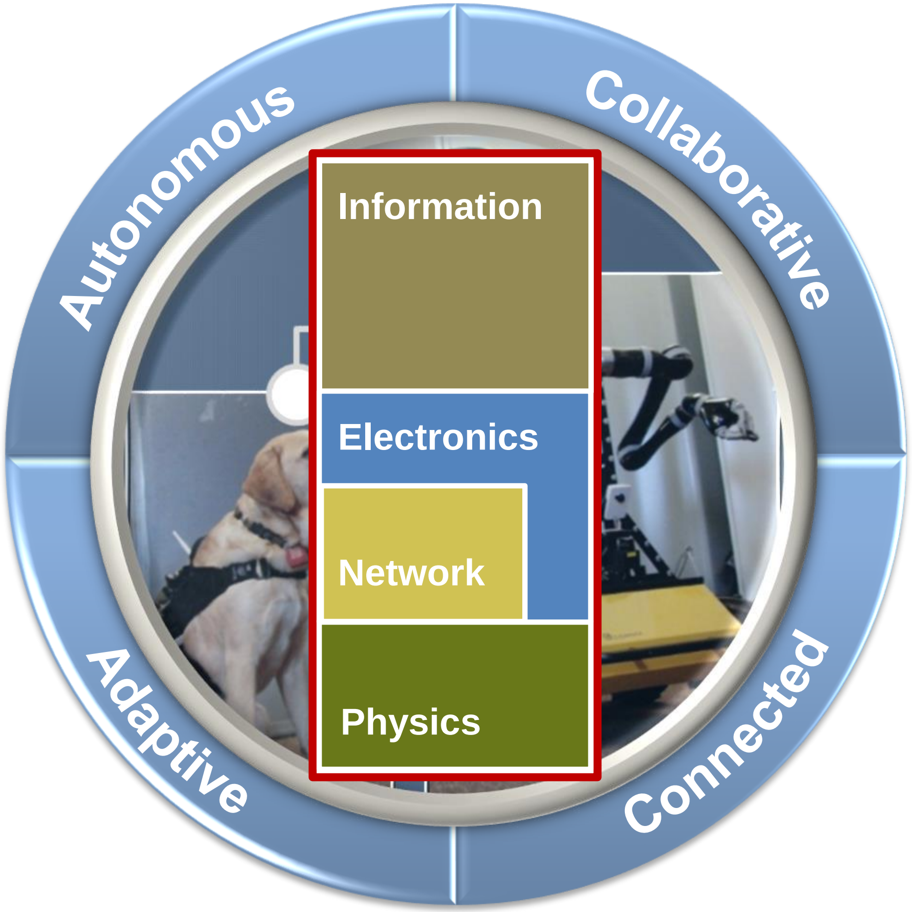
A feature classification

Evolving architectures

Future value drivers





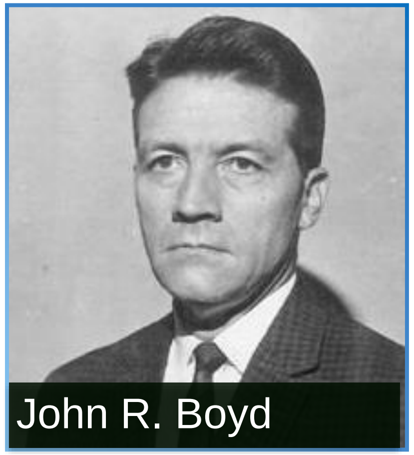


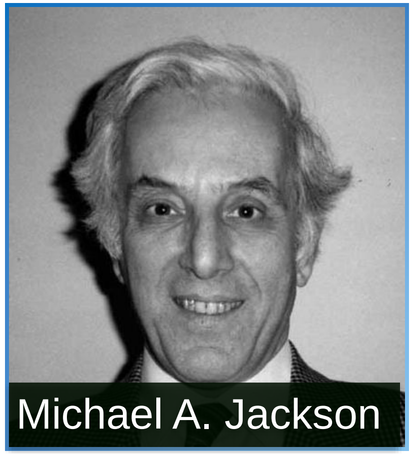
A smart emergency response system

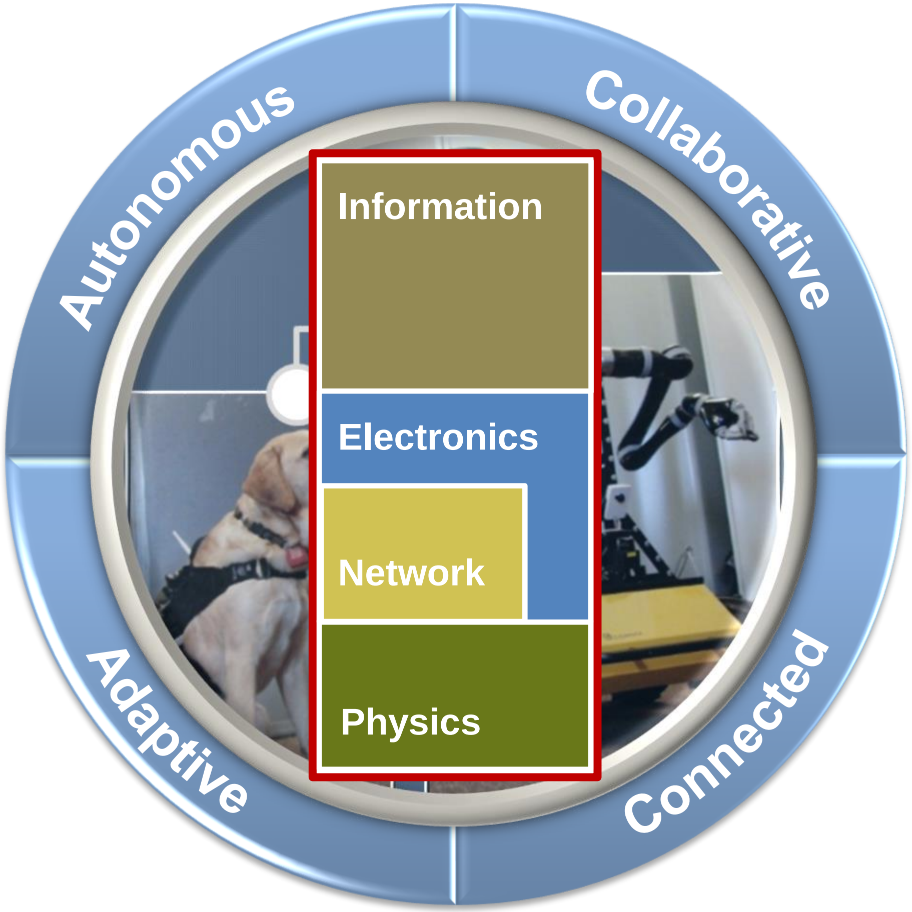
A feature classification

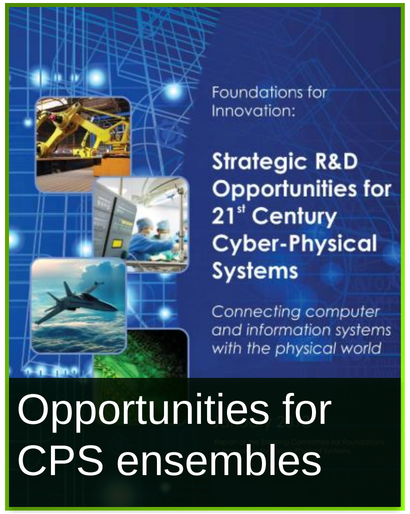
Evolving architectures

Future value drivers









A smart emergency response system

A feature classification

Evolving architectures

Future value drivers

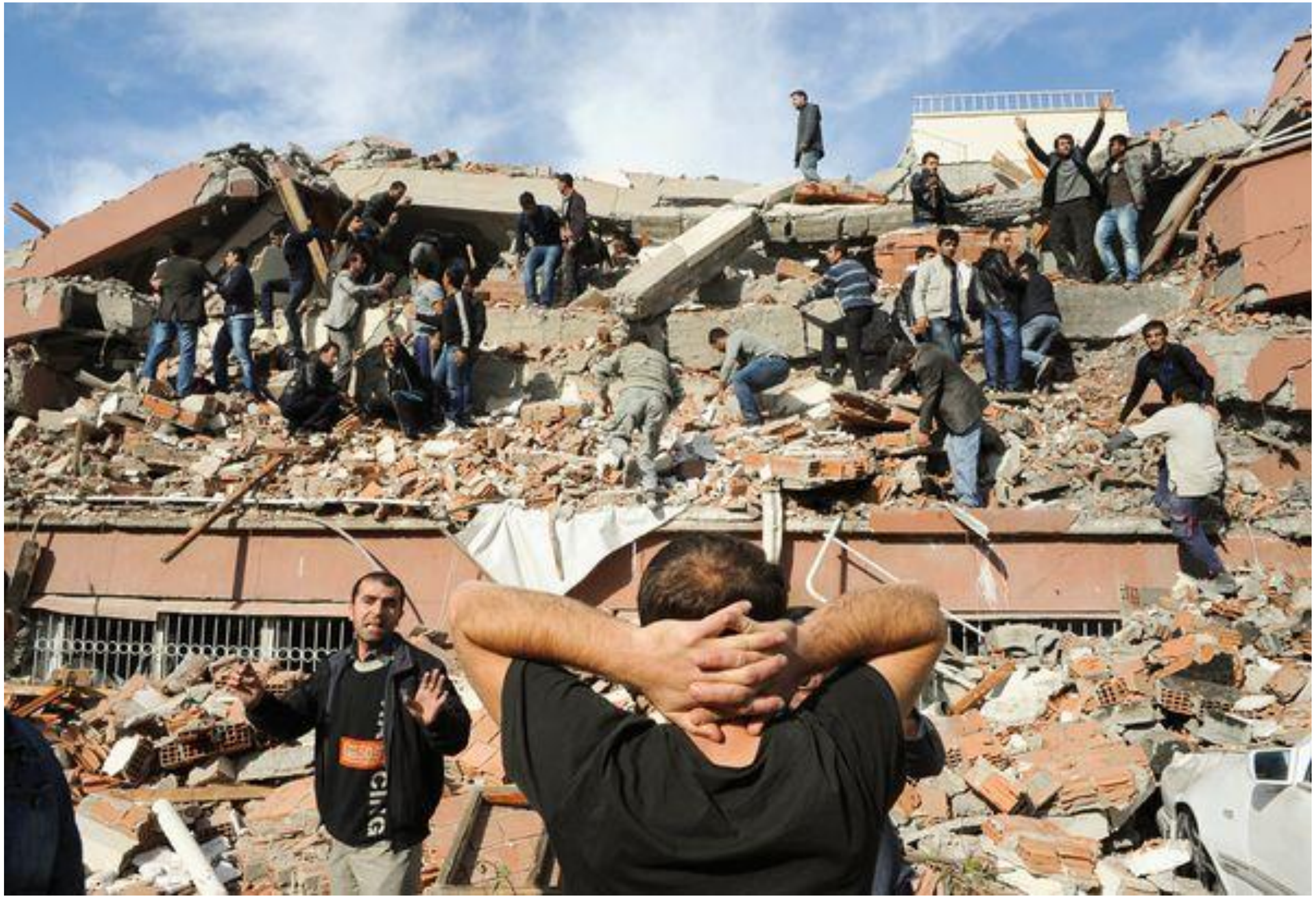
2013 MathWorks Summer Research Internship

Kun Zhang
University of Arizona

Enes Bilgin
Boston University

David Escobar Sanabria
University of Minnesota

İzmit, Turkey, 1999



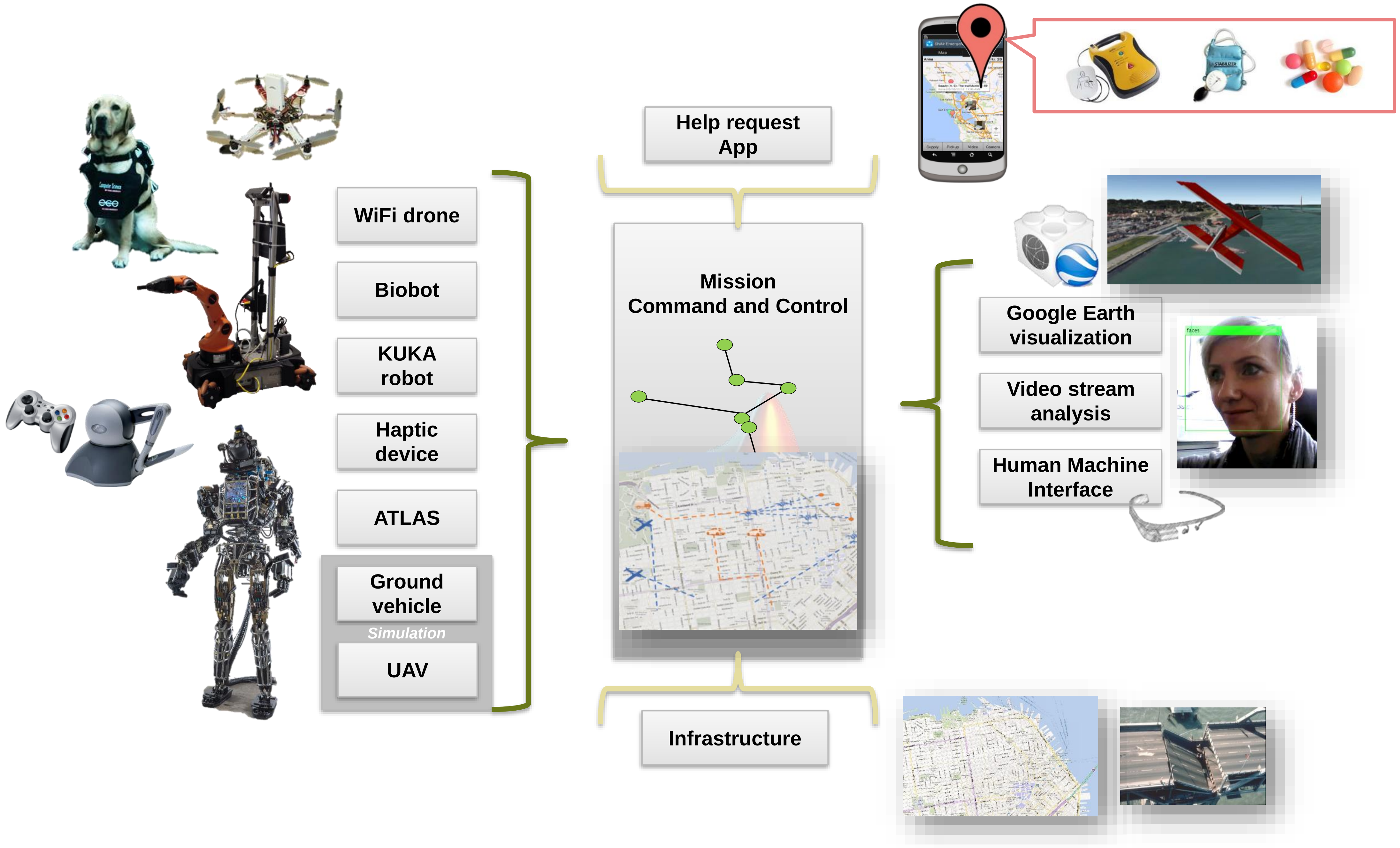
Where can we help make a difference?

Deploy a heterogeneous fleet …

… to serve many (changing!) requests …

… across an uncertain infrastructure …

… all in a time optimal manner!



WiFi drone
Biobot
KUKA robot
Haptic device
ATLAS
Ground vehicle
Simulation
UAV
Help request App
Mission Command and Control
Infrastructure
Google Earth visualization
Video stream analysis
Human Machine Interface

2013 MathWorks Summer Research Internship: A Spectacular Challenge (Get cyber real!)

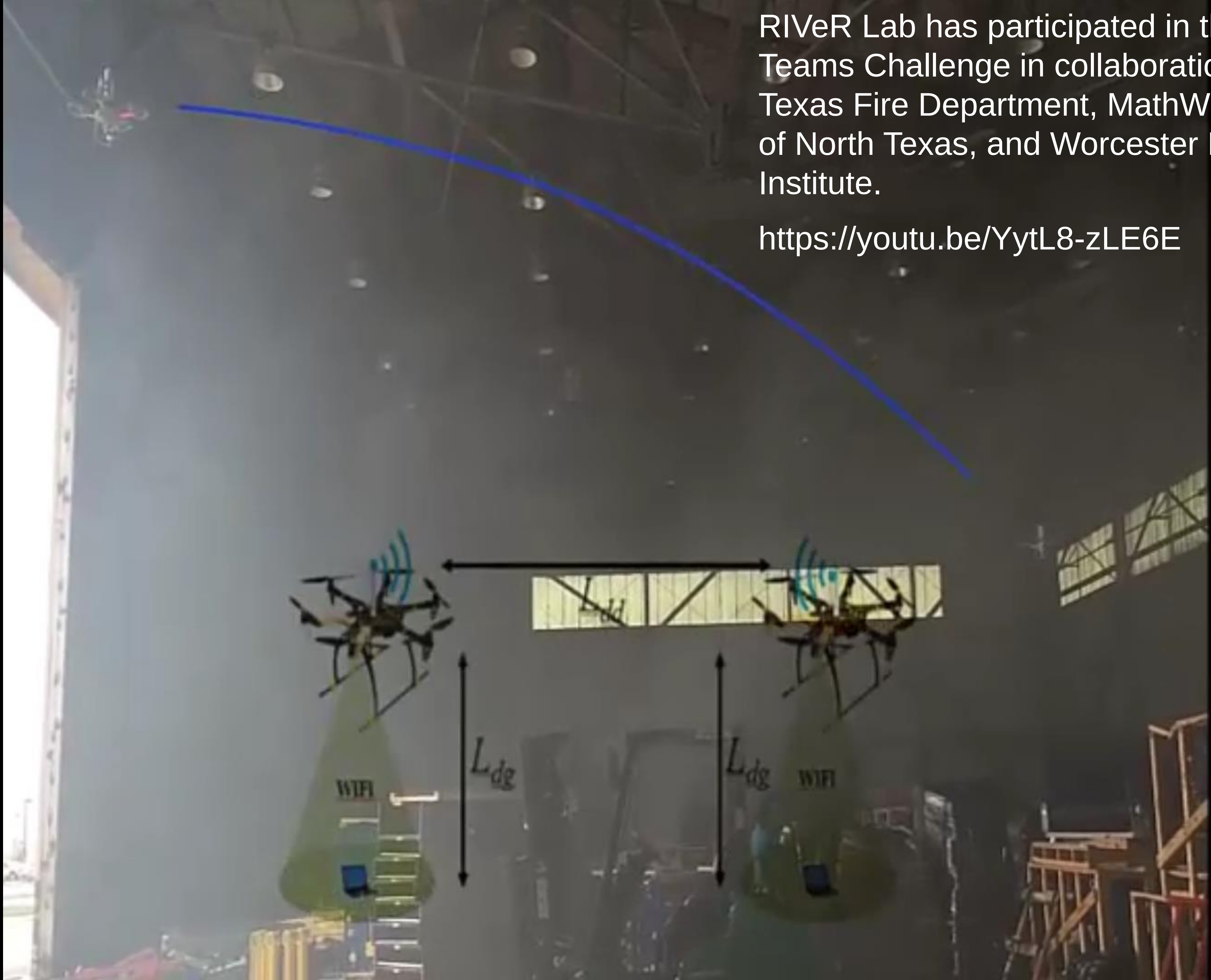
https://youtu.be/MxrySx1m8VQ?t=2m42s

Man-machine control transfer

hover | transfer control | control by responder

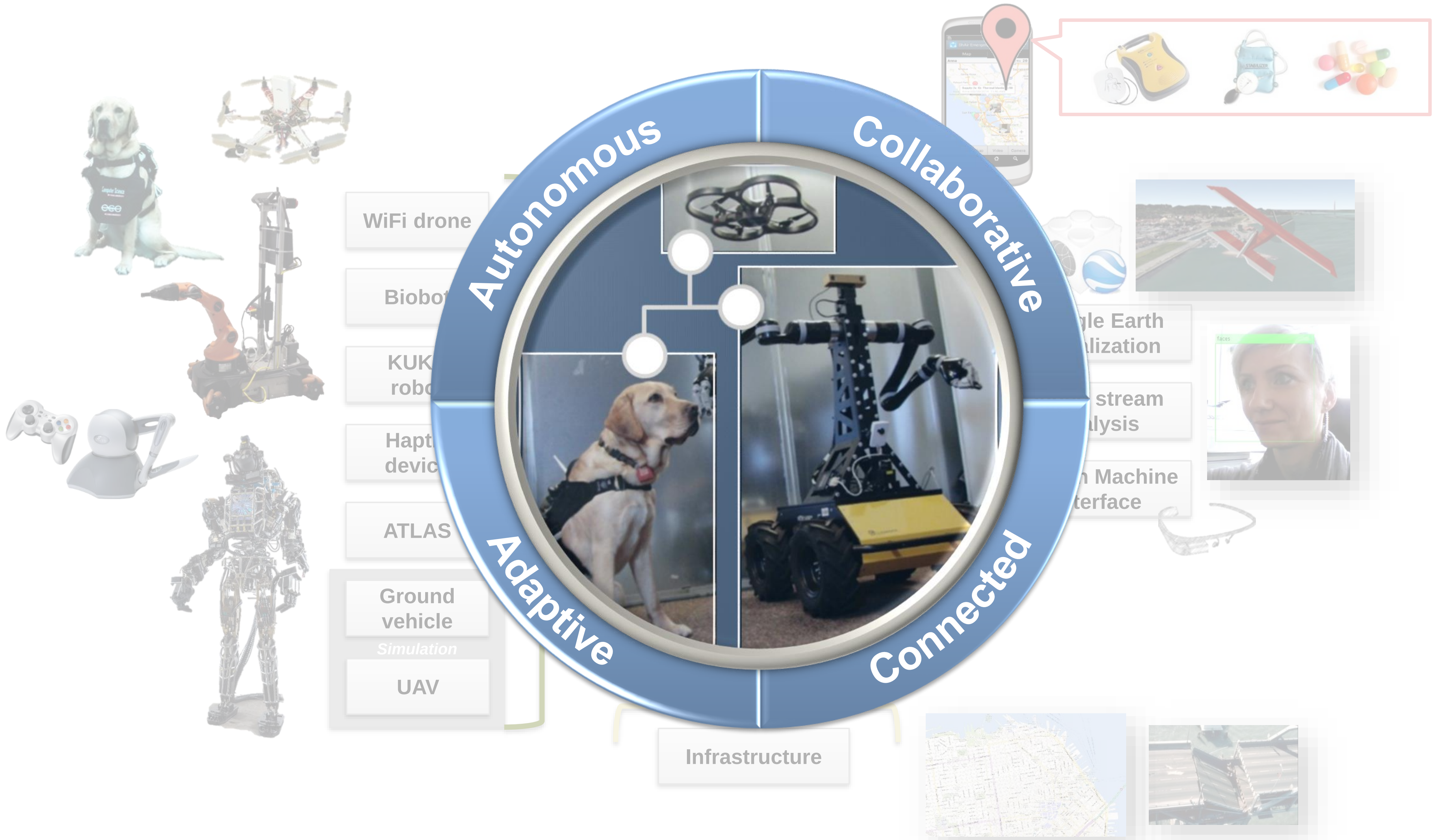
2013 MathWorks Summer Research Internship: A Self-flying Drone (Take cyber control!)

https://youtu.be/M3vq1ywbe10?t=41s



RIVeR Lab has participated in the Global City Teams Challenge in collaboration with Austin Texas Fire Department, MathWorks, University of North Texas, and Worcester Polytechnic Institute.

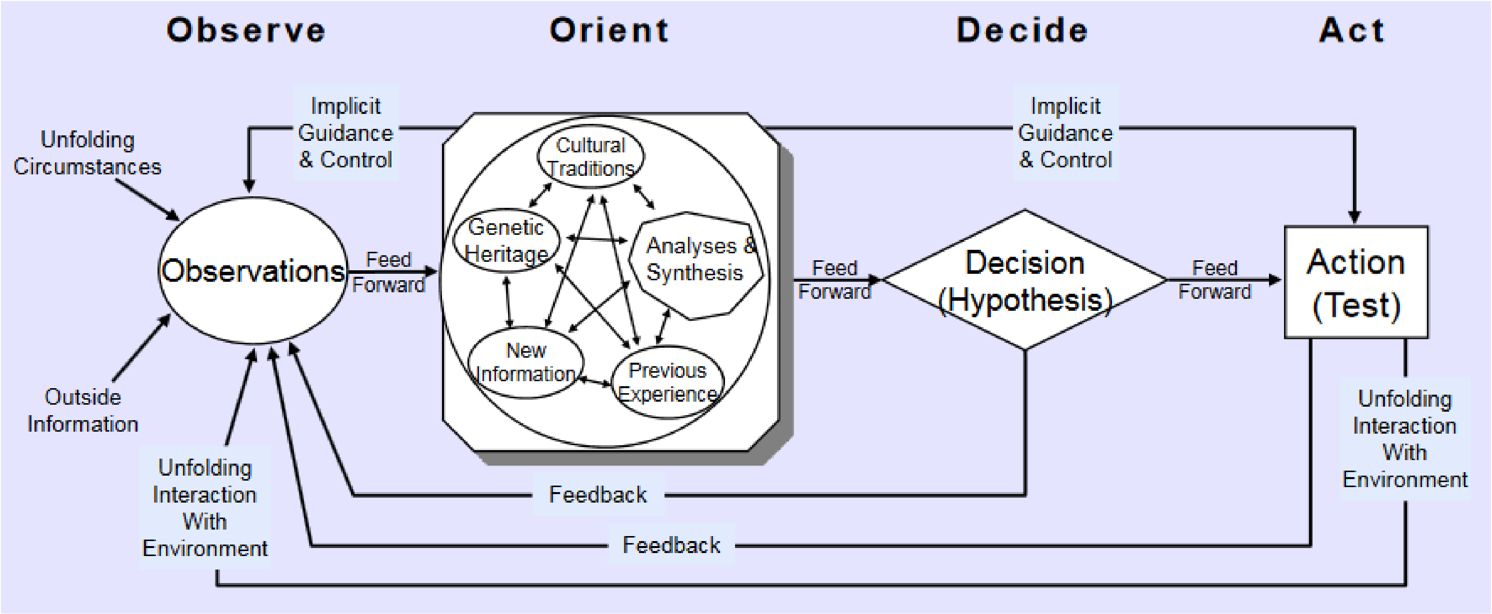
https://youtu.be/YytL8-zLE6E


Autonomous
Collaborative
Adaptive
Connected
WiFi drone
Bioboť
KUK robo
Hapt devic
ATLAS
Ground vehicle
Simulation
UAV
Infrastructure
gle Earth alization
stream alysis
n Machine terface

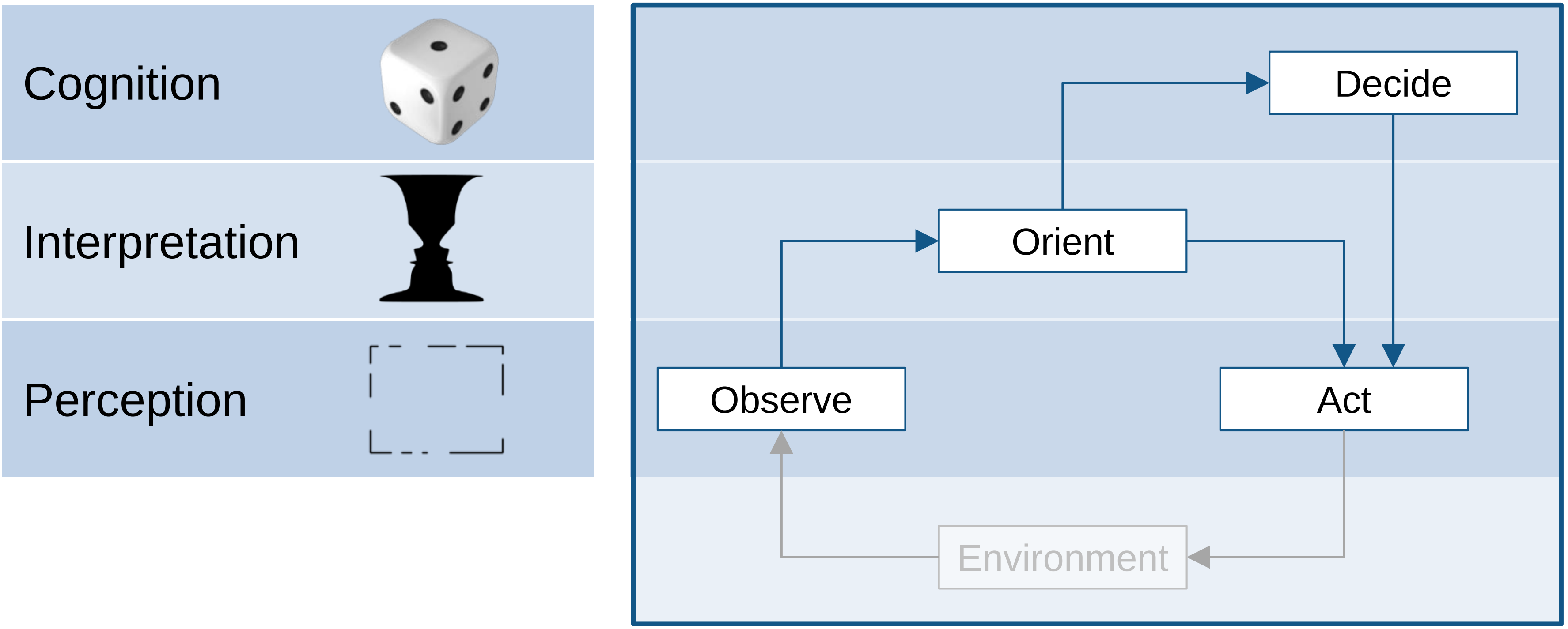

A smart emergency response system
A feature classification
Evolving architectures
Future value drivers

Colonel John Richard Boyd

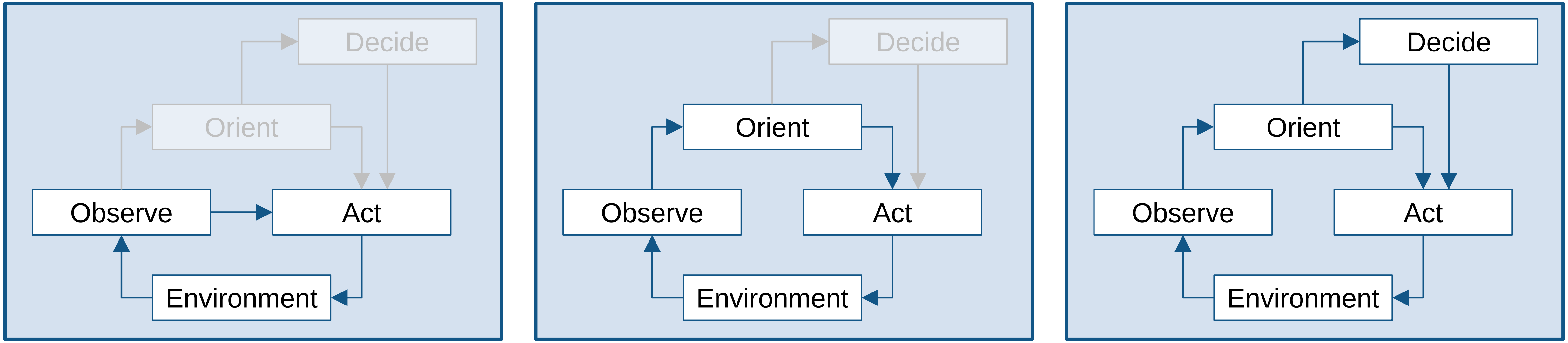
The Observe-Orient-Decide-Act (OODA) loop



OODA and the stages of cognition

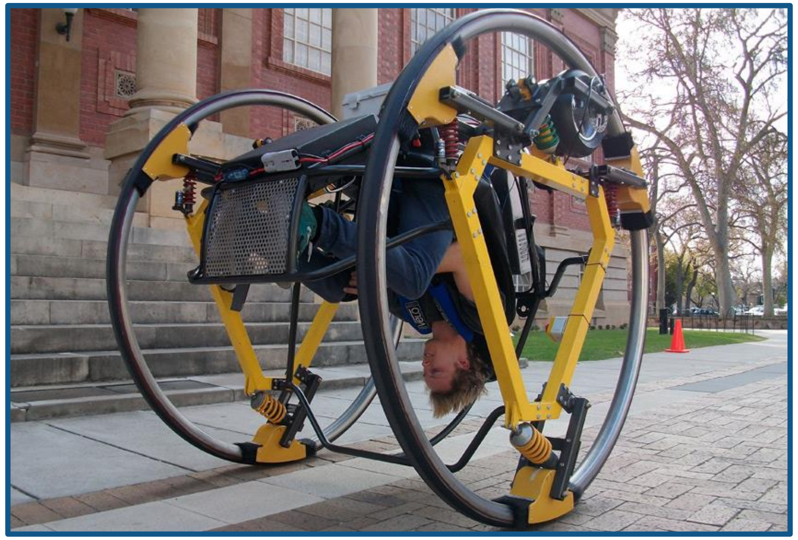


Engineered systems and the stages of cognition



Automatic	Adaptive	Autonomous
Perception Process (video) Analyze for validity (filter, reject) Compute control signals	Interpretation Map to semantic concepts Fuse sensor data Adjust control Reconfigure behavior	Cognition Reason based on knowledge Plan for objectives and constraints Assess alternatives Determine course of action

Engineered systems and the stages of cognition



Automatic

Perception

Process (video)
Analyze for validity (filter, reject)

Compute control signals

Adaptive

Interpretation

Map to semantic concepts
Fuse sensor data

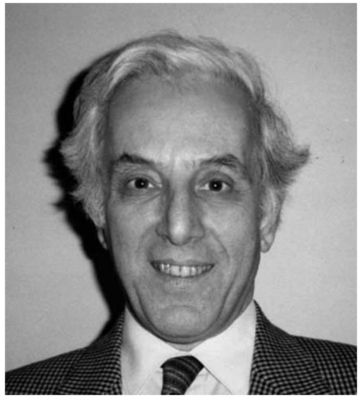
Adjust control
Reconfigure behavior

Autonomous

Cognition

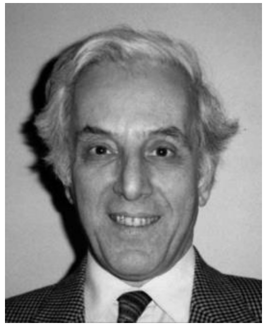
Reason based on knowledge
Plan for objectives and constraints

Assess alternatives
Determine course of action

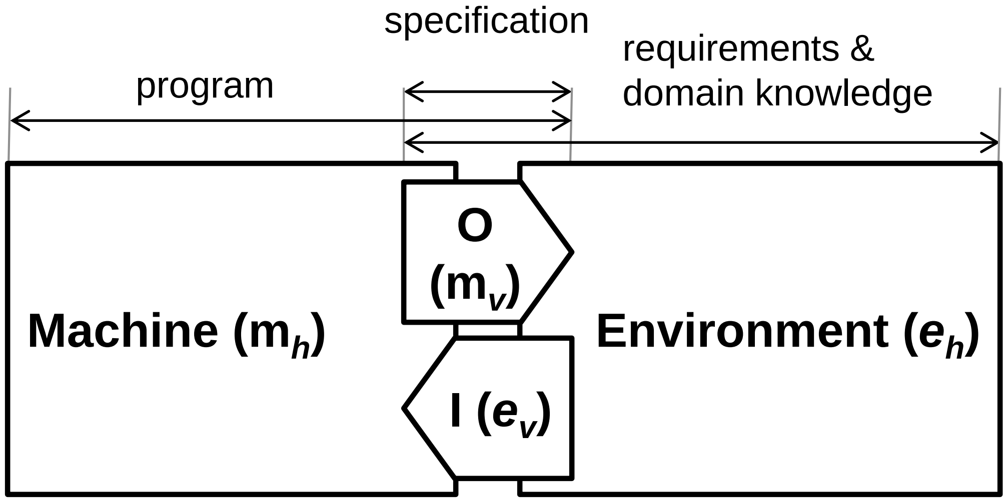


Michael Anthony Jackson

Requirements engineering

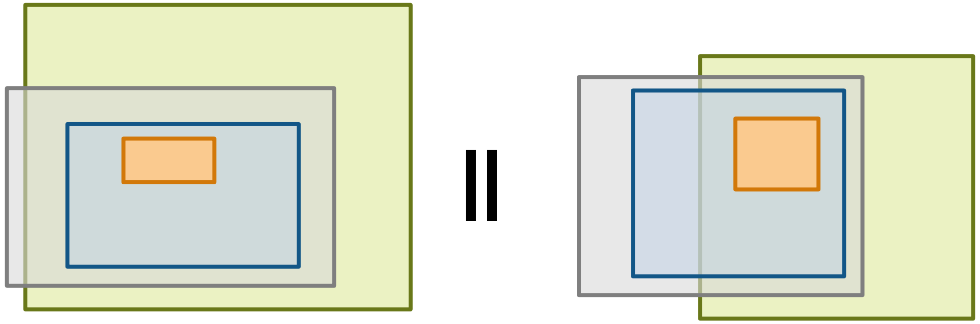


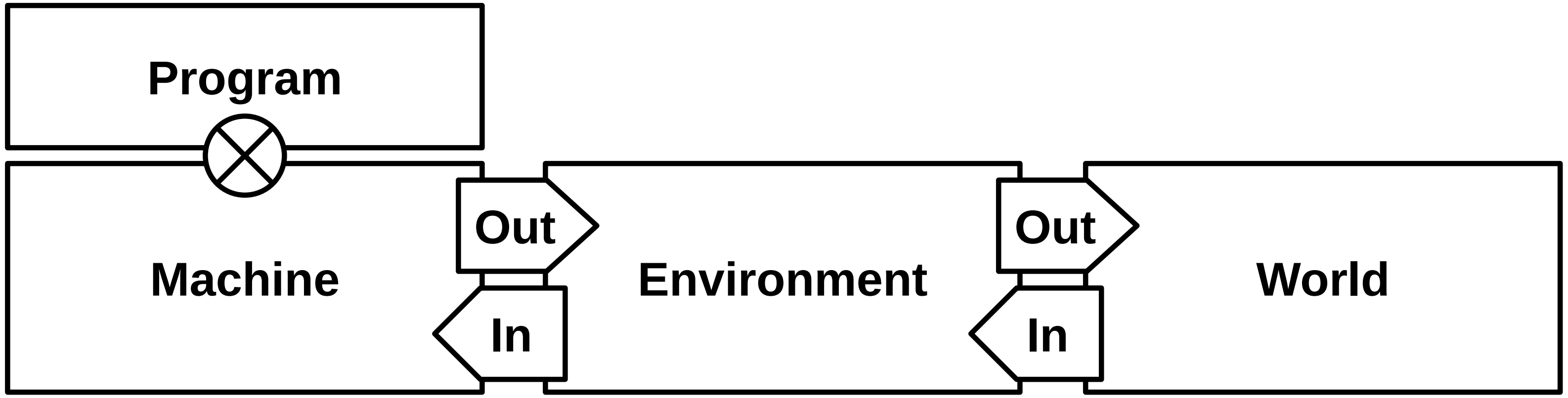
- A **requirement** is a desired relationship among the **phenomena** (e.g., actions/events, states) of the environment
- Phenomena are categorized as
  - ***e****$_h$*: controlled (or initiated) by the **e**nvironment and **h**idden from (i.e., invisible to, not shared with) the machine
  - ***e****$_v$*: controlled by the **e**nvironment but **v**isible to (i.e., shared with) the machine
  - ***m****$_v$*: controlled by the **m**achine but **v**isible to (shared with) the environment
  - ***m****$_h$*: controlled by the **m**achine and **h**idden from (i.e., not shared with) the environment



A behavioral view

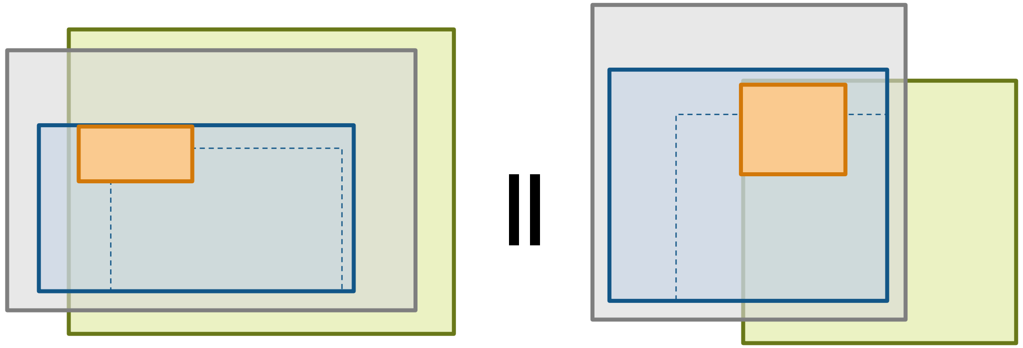
Closed loop designed behavior
Property satisfying behavior
Closed loop possible behavior
Open loop possible behavior

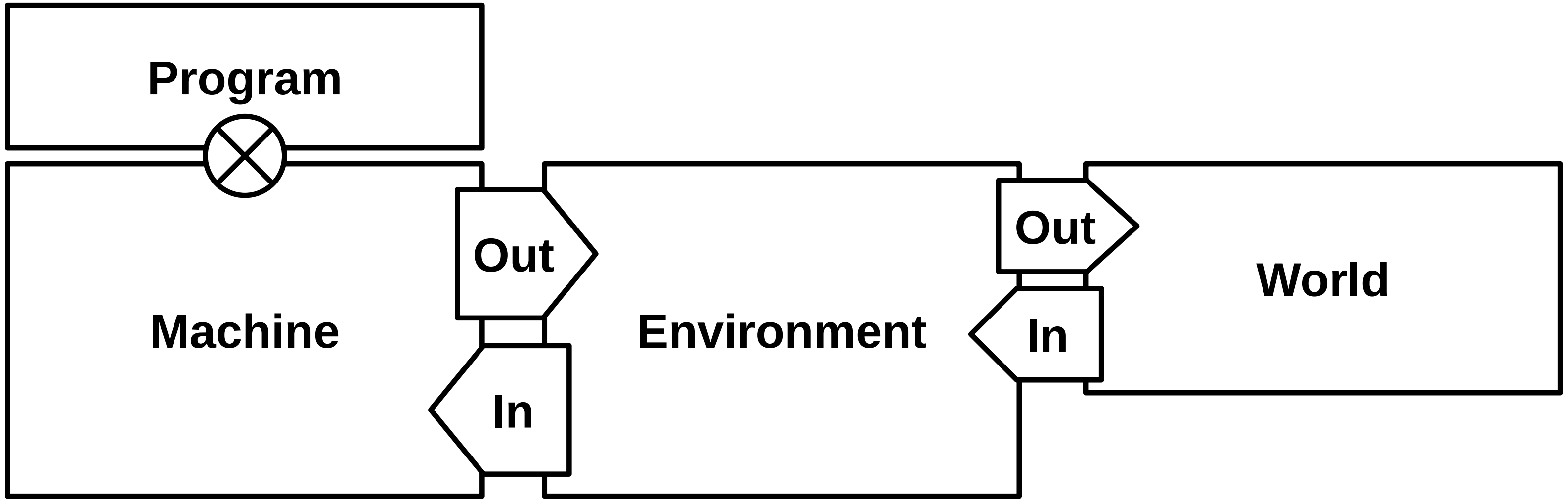




A behavioral view

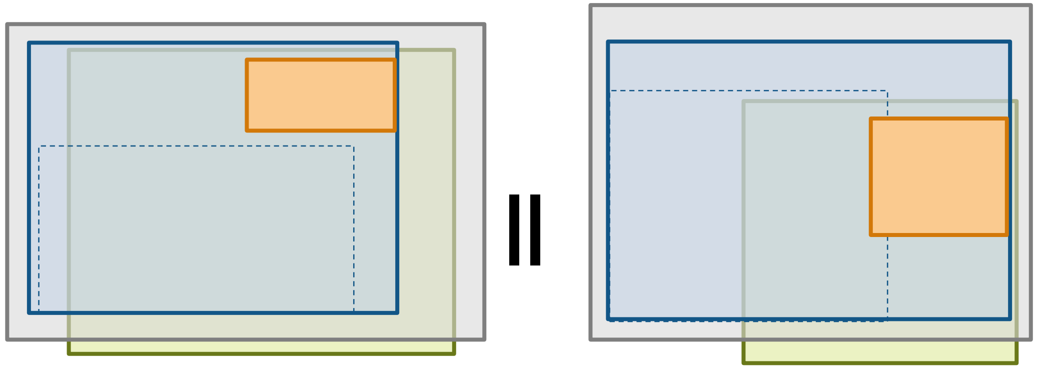
Closed loop designed behavior
Property satisfying behavior
Closed loop possible behavior
Open loop possible behavior

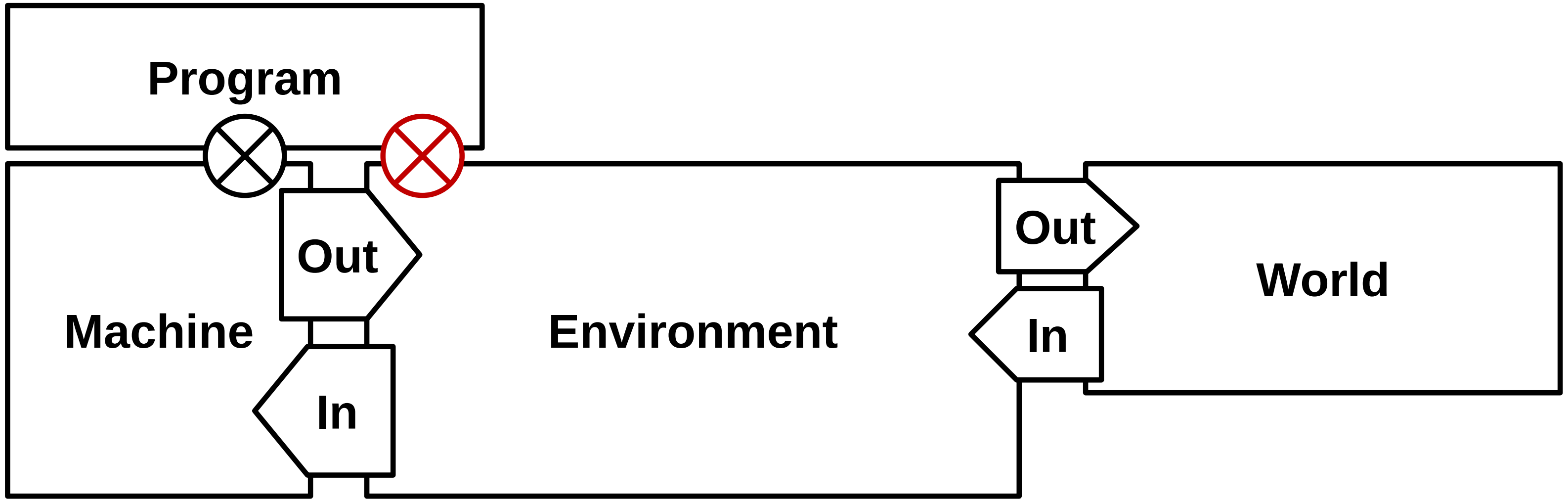




A behavioral view

Closed loop designed behavior
Property satisfying behavior
Closed loop possible behavior
Open loop possible behavior





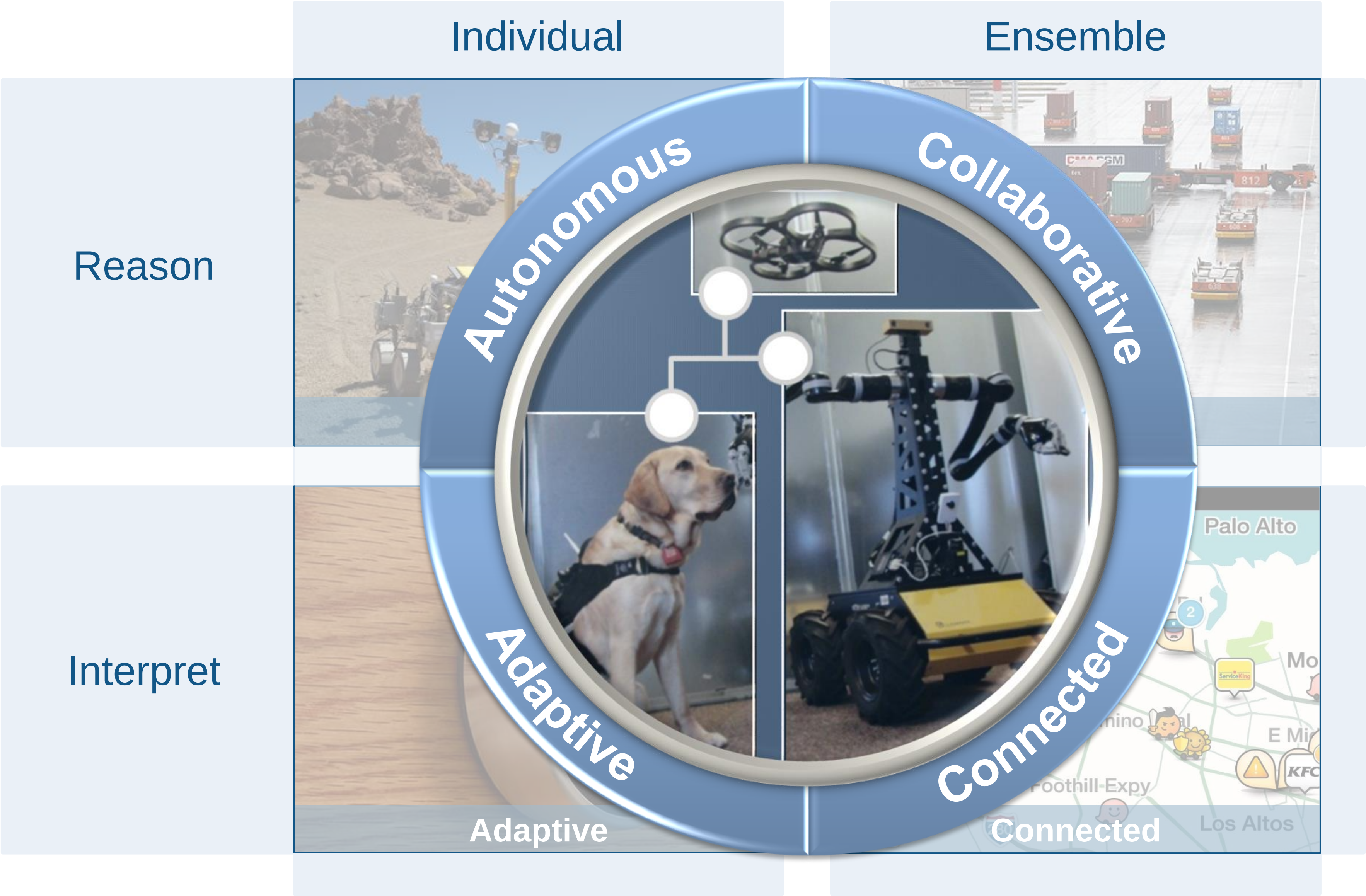
For fully-automated container handling in large terminals and terminal environments, Terex Port Solutions supplies solutions with outstanding performance.

https://youtu.be/STlt48wXsyY?t=17s

A feature classification

	Individual	Ensemble
Reason	 Autonomous	 Collaborative
Interpret	 Adaptive	 Connected

A feature classification

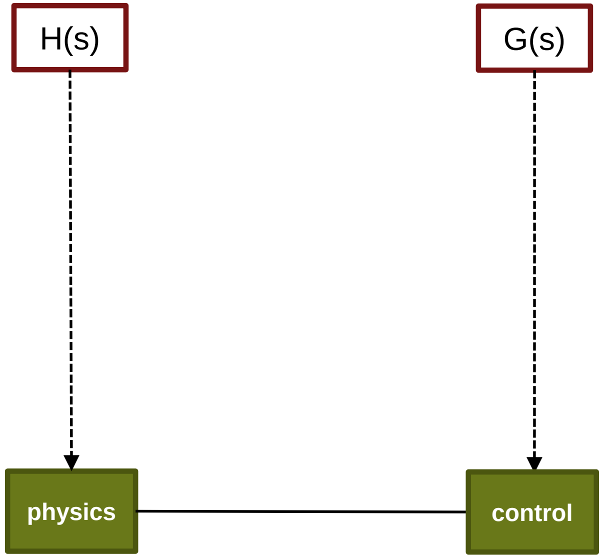


A smart emergency response system

A feature classification

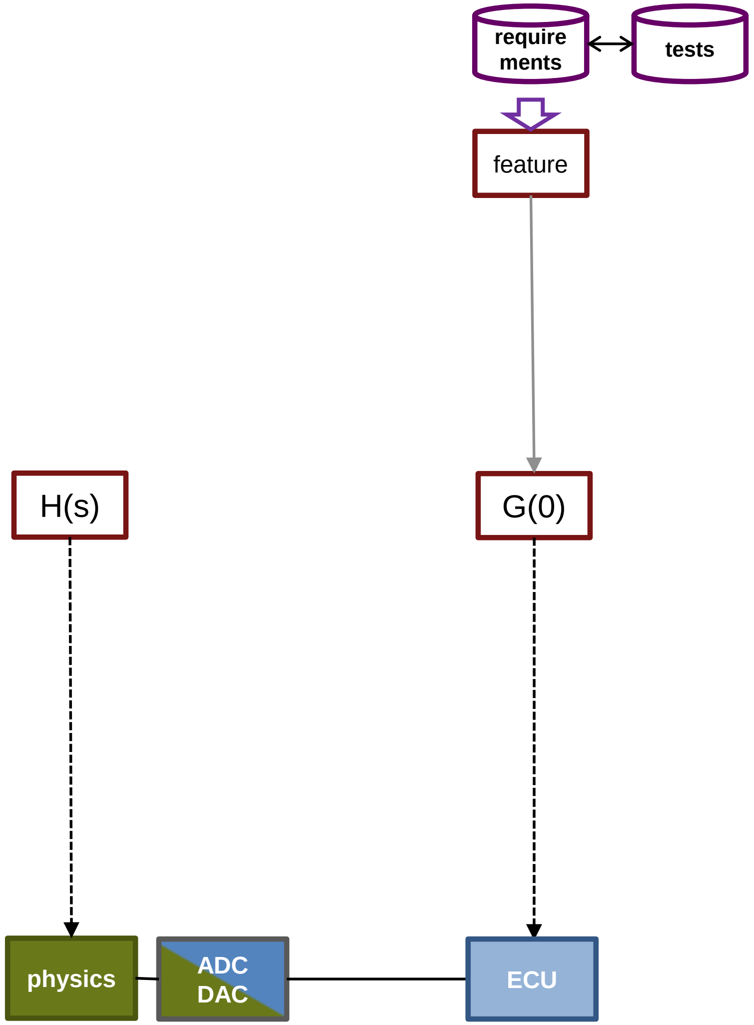
Evolving architectures

Future value drivers



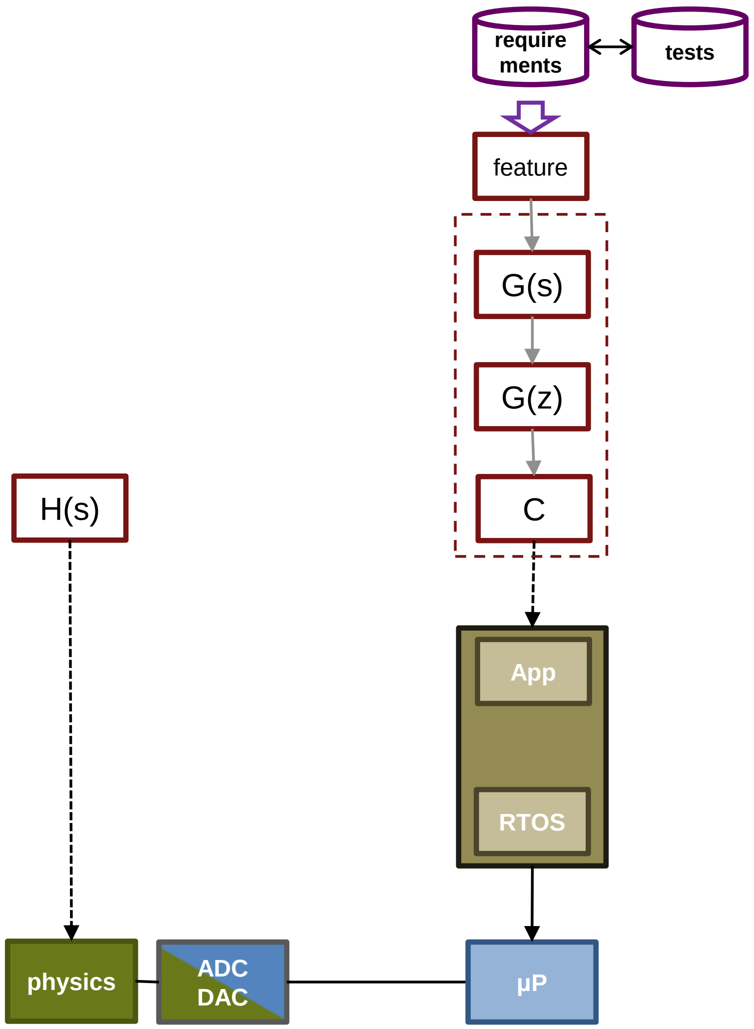
Centrifugal governor

(James Watt designed his first governor in 1788 following a suggestion from his business partner Matthew Boulton)



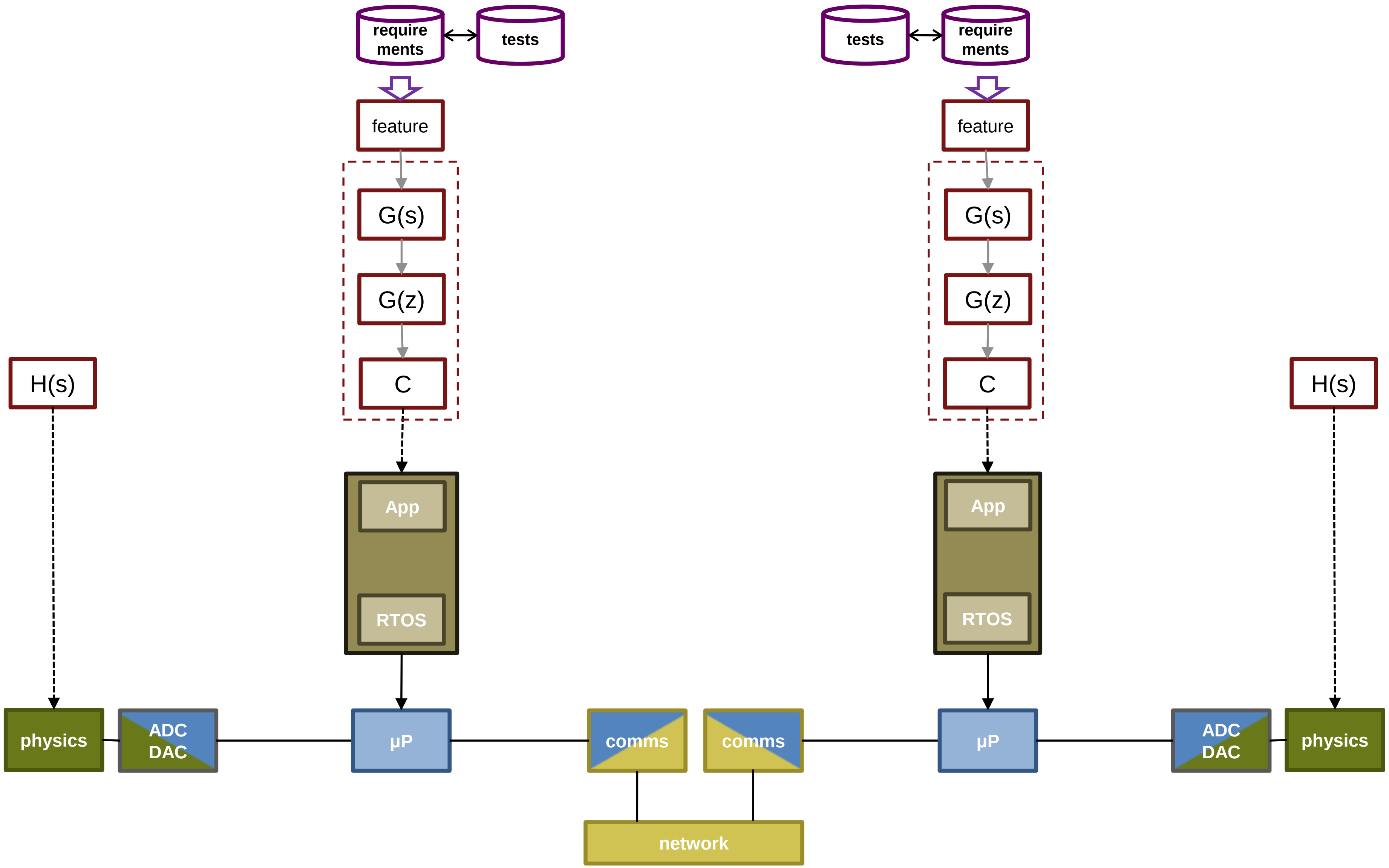
Engine control unit

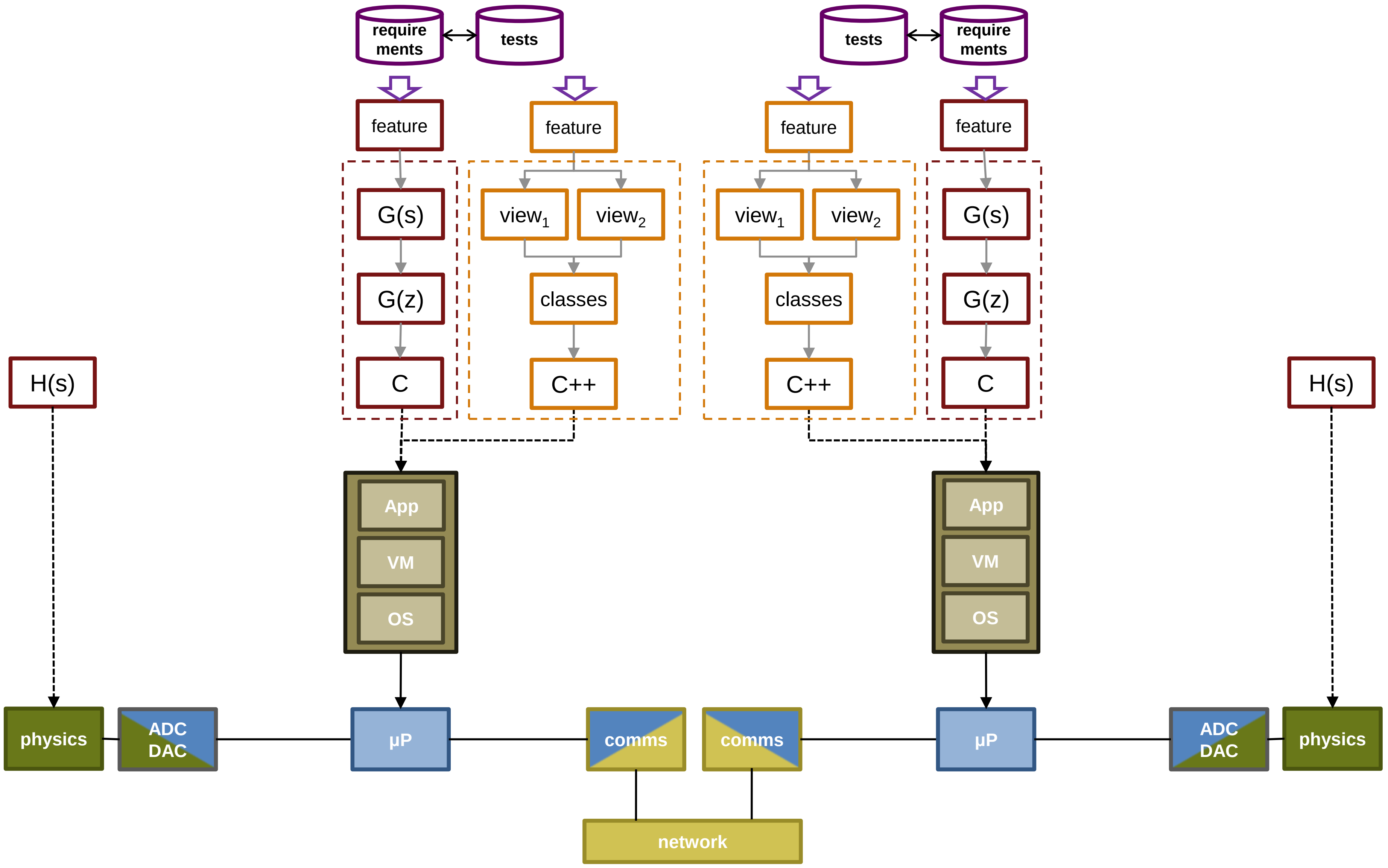
(from a 1996 Chevrolet Beretta)



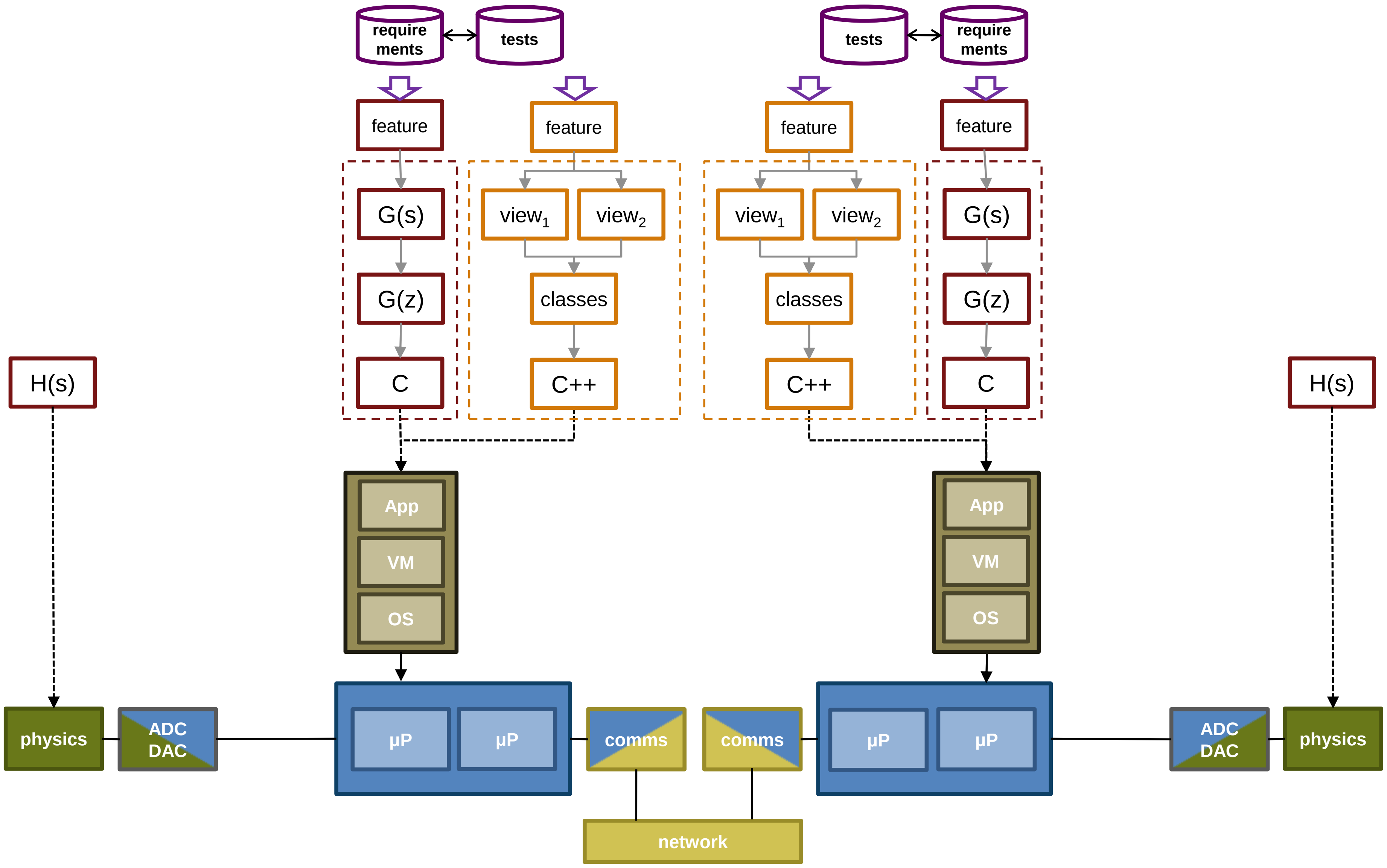
Freescale MPC561 MCU

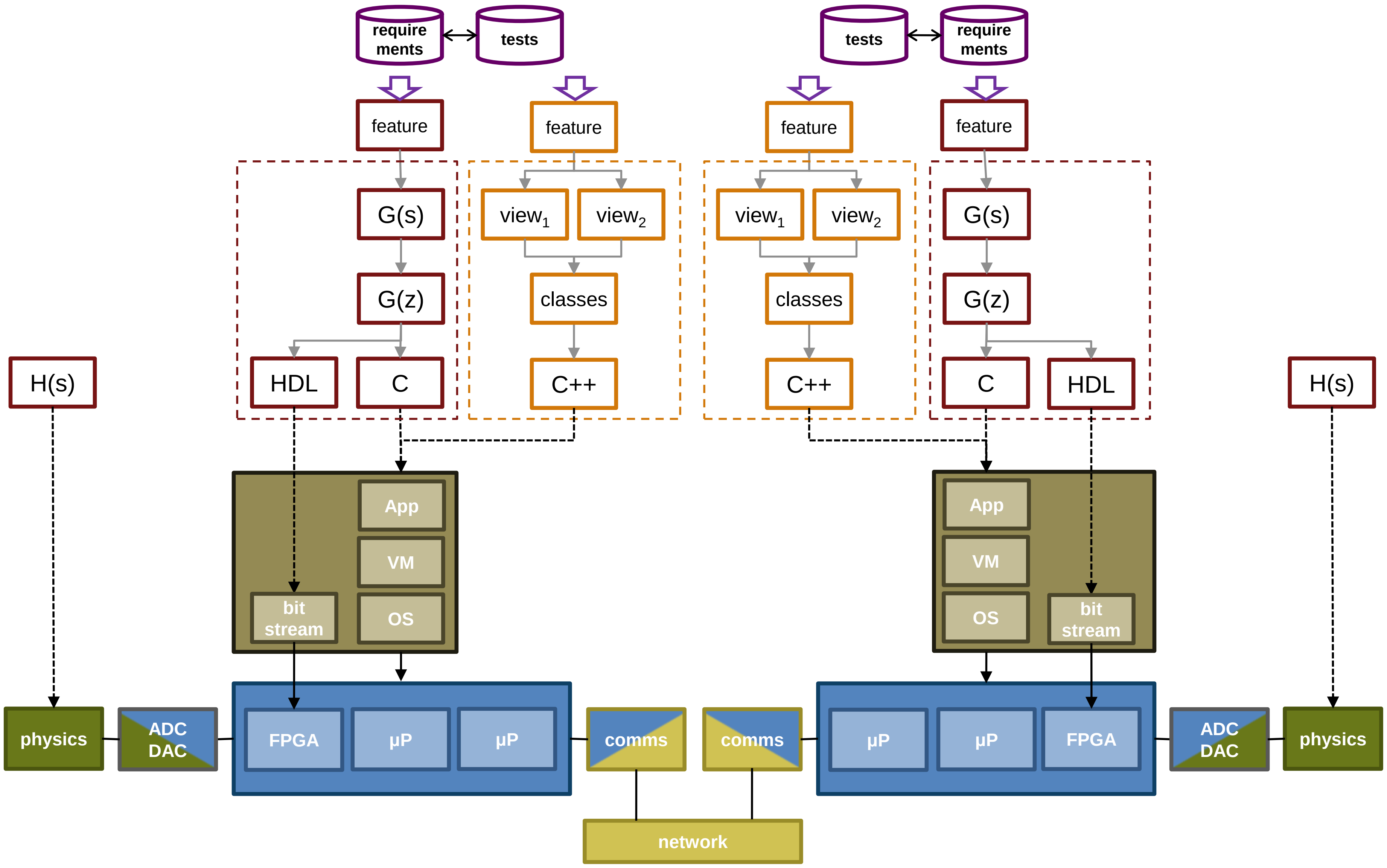
(32-bit PowerPC embedded microprocessors that operate between 40 and 66 MHz, used in engine controllers for General Motors)

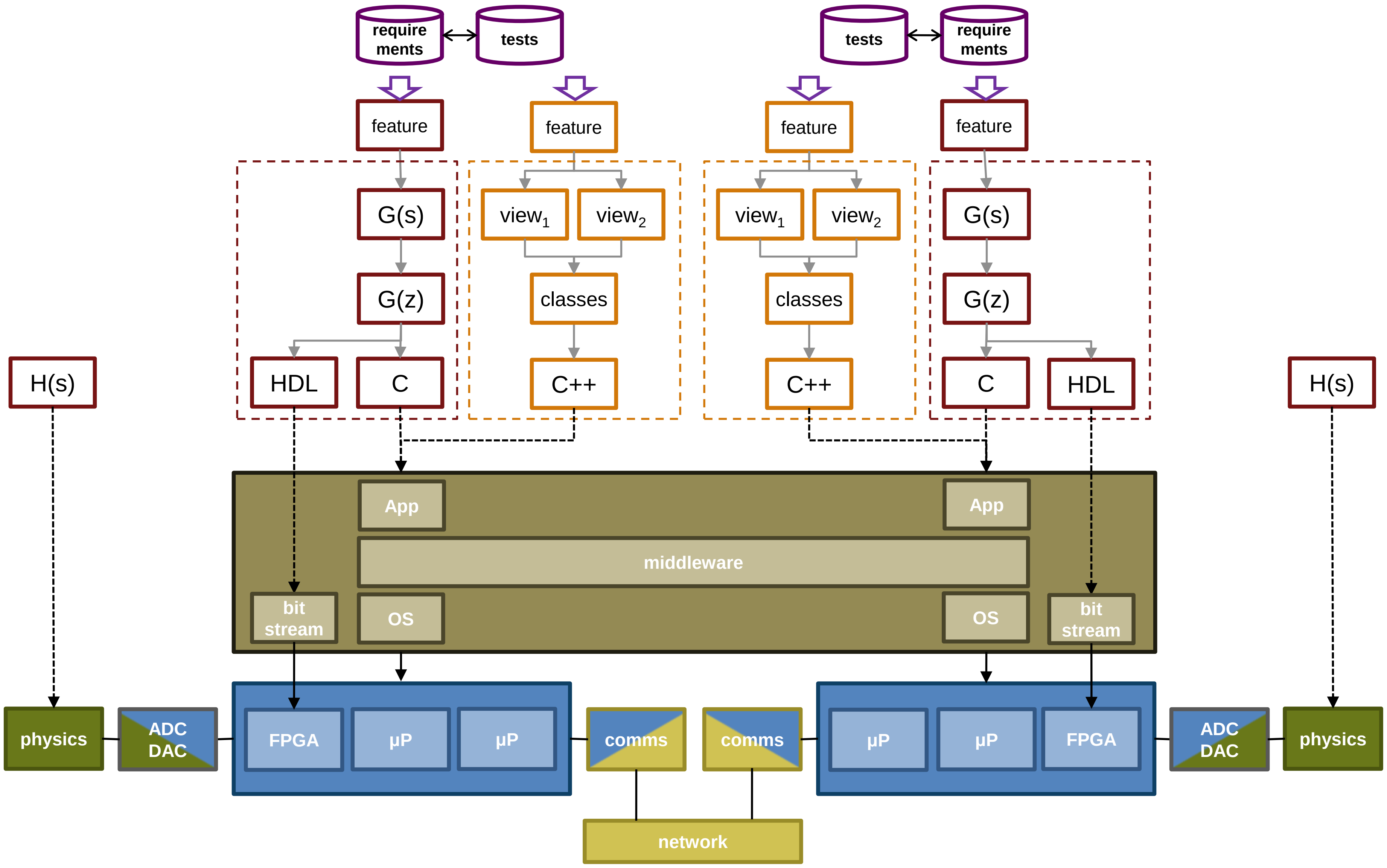

require ments
tests
feature
G(s)
G(z)
C
H(s)
App
RTOS
physics
ADC DAC
µP
comms
comms
network
tests
require ments
feature
G(s)
G(z)
C
H(s)
App
RTOS
µP
ADC DAC
physics

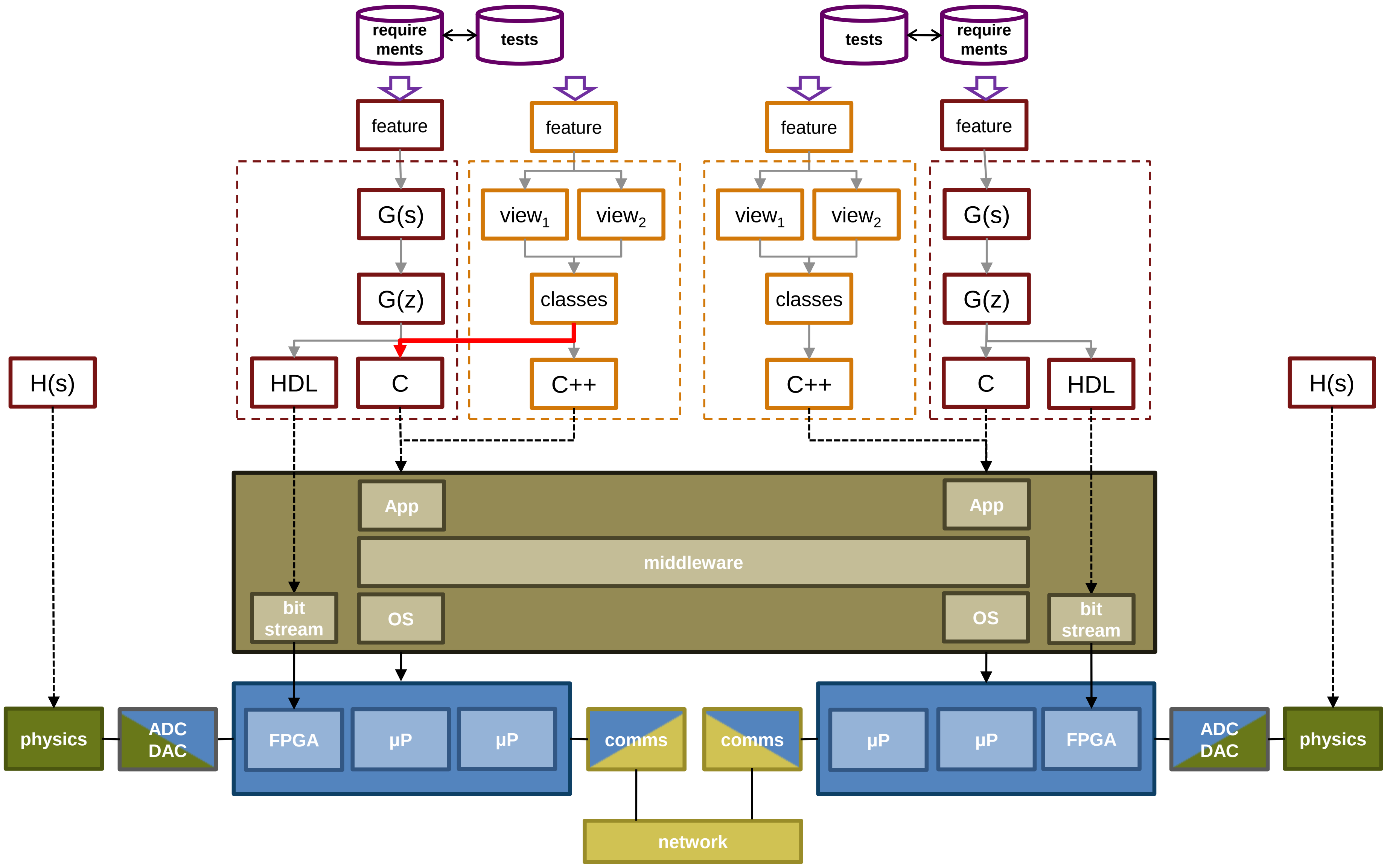


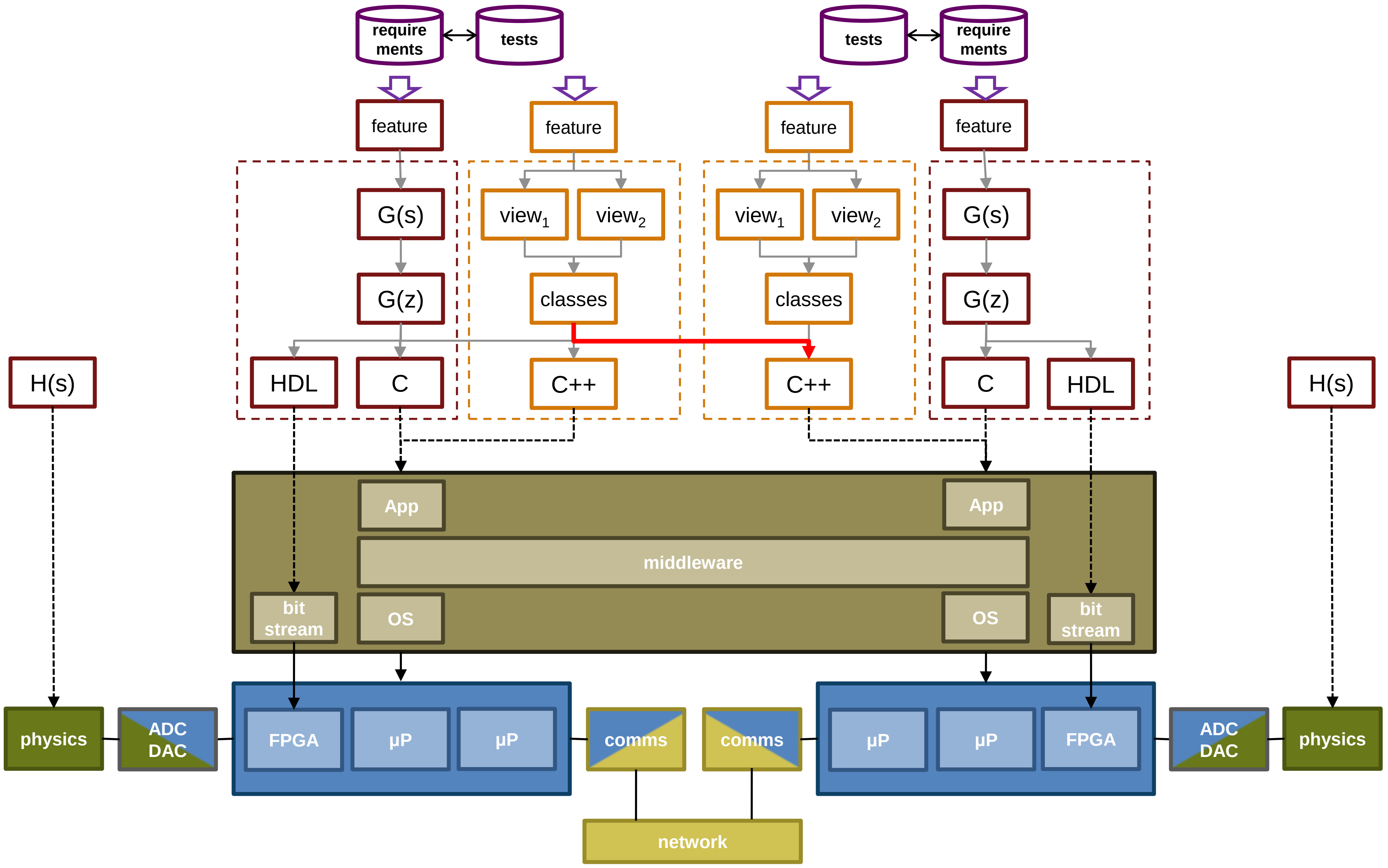
require ments
tests
tests
require ments
feature
feature
feature
feature
G(s)
view1
view2
view1
view2
G(s)
G(z)
classes
classes
G(z)
H(s)
C
C++
C++
C
H(s)
App
VM
OS
App
VM
OS
physics
ADC DAC
μP
comms
comms
μP
ADC DAC
physics
network

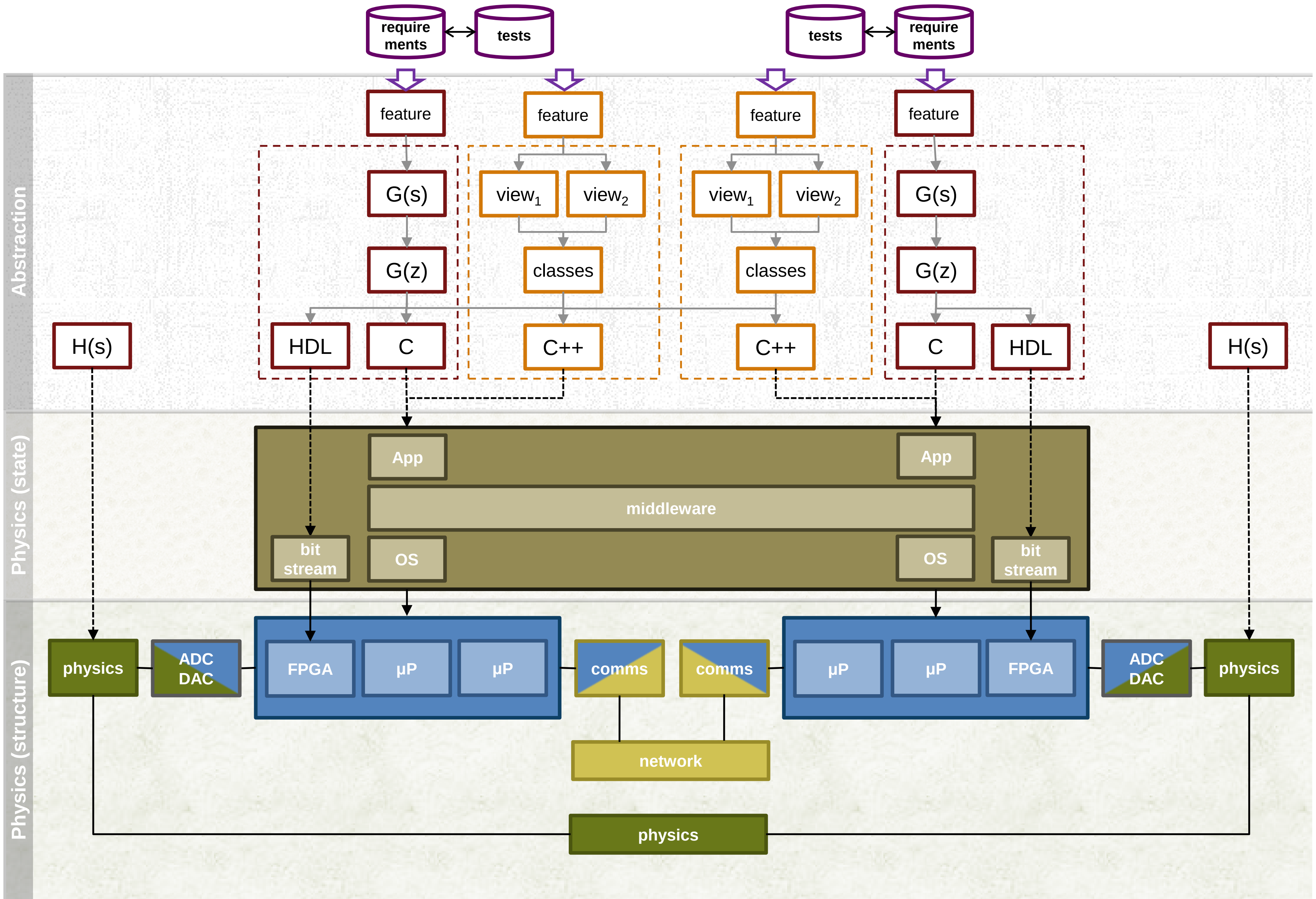

require ments
tests
feature
feature
G(s)
view1
view2
G(z)
classes
C
C++
H(s)
tests
require ments
feature
feature
view1
view2
G(s)
classes
G(z)
C++
C
H(s)
App
VM
OS
App
VM
OS
physics
ADC DAC
µP
µP
comms
comms
µP
µP
ADC DAC
physics
network


require ments
tests
feature
G(s)
G(z)
HDL
C
view$_1$
view$_2$
classes
C++
H(s)
App
VM
OS
bit stream
physics
ADC DAC
FPGA
µP
comms
network
tests
require ments
feature
view$_1$
view$_2$
classes
C++
G(s)
G(z)
C
HDL
H(s)
App
VM
OS
bit stream
µP
FPGA
ADC DAC
physics

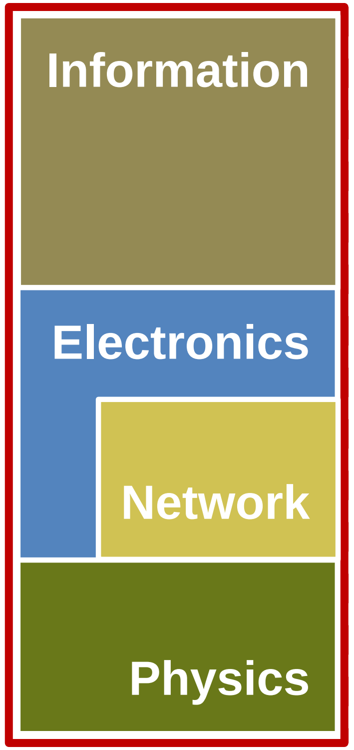

require ments
tests
tests
require ments
feature
feature
feature
feature
G(s)
view1
view2
view1
view2
G(s)
G(z)
classes
classes
G(z)
H(s)
HDL
C
C++
C++
C
HDL
H(s)
App
App
middleware
bit stream
OS
OS
bit stream
physics
ADC DAC
FPGA
µP
µP
comms
comms
µP
µP
FPGA
ADC DAC
physics
network

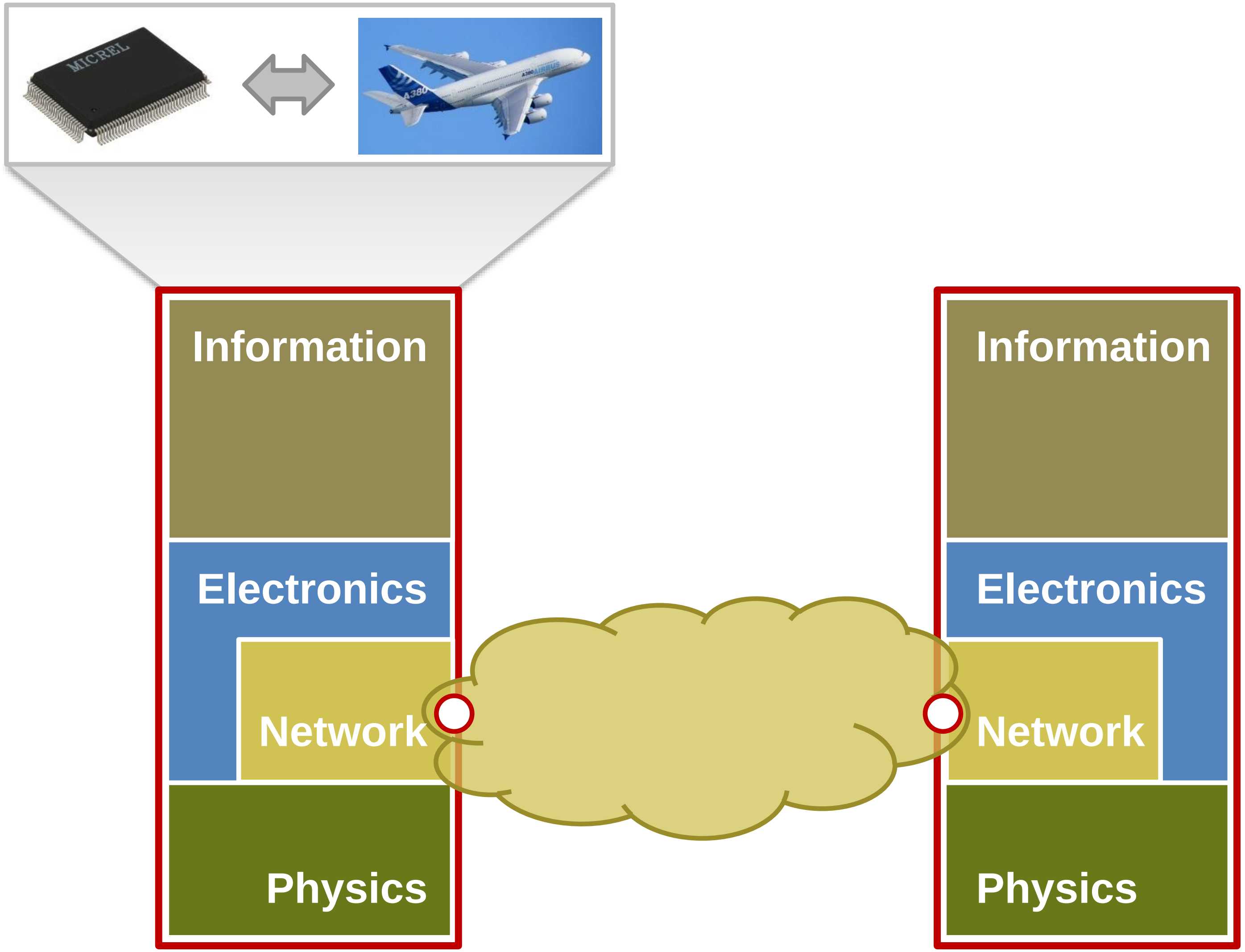

require ments
tests
tests
require ments
feature
feature
feature
feature
G(s)
view1
view2
view1
view2
G(s)
G(z)
classes
classes
G(z)
H(s)
HDL
C
C++
C++
C
HDL
H(s)
App
App
middleware
bit stream
OS
OS
bit stream
physics
ADC DAC
FPGA
µP
µP
comms
comms
µP
µP
FPGA
ADC DAC
physics
network

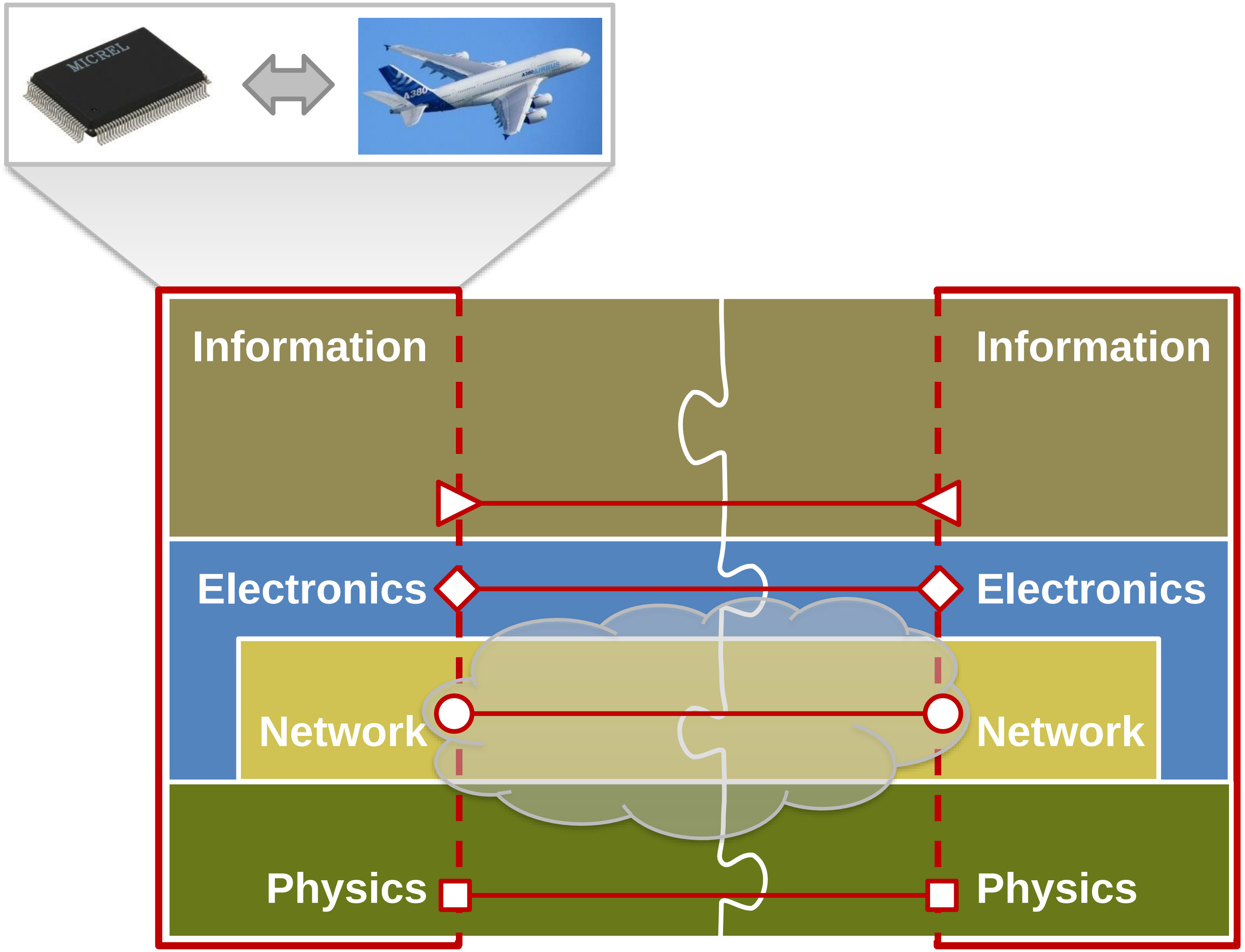

require ments
tests
tests
require ments
feature
feature
feature
feature
G(s)
view1
view2
view1
view2
G(s)
G(z)
classes
classes
G(z)
H(s)
HDL
C
C++
C++
C
HDL
H(s)
App
App
middleware
bit stream
OS
OS
bit stream
physics
ADC DAC
FPGA
µP
µP
comms
comms
µP
µP
FPGA
ADC DAC
physics
network

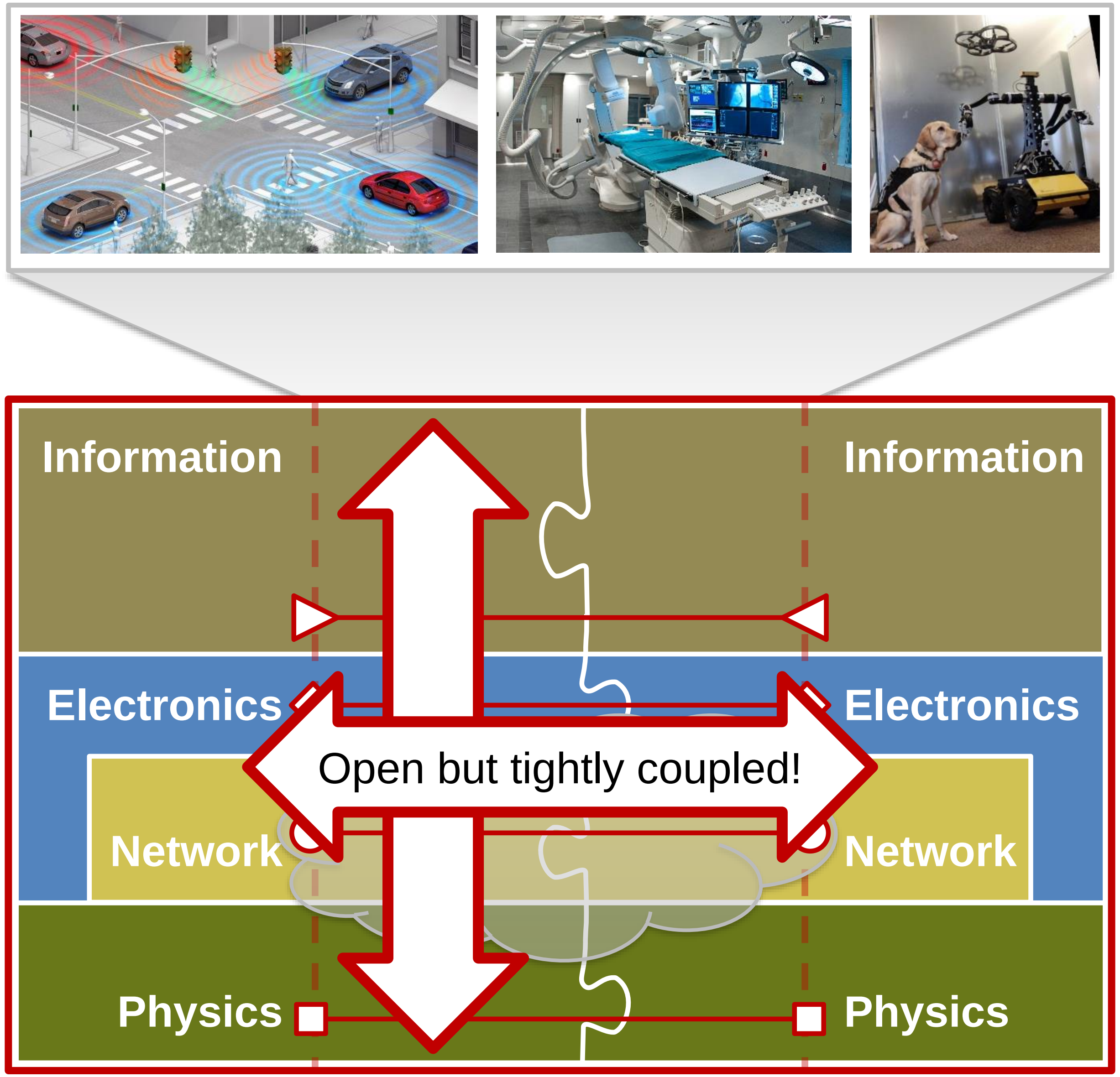


require ments
tests
tests
require ments
feature
feature
feature
feature
G(s)
view1
view2
view1
view2
G(s)
G(z)
classes
classes
G(z)
H(s)
HDL
C
C++
C++
C
HDL
H(s)
Abstraction
App
App
middleware
bit stream
OS
OS
bit stream
Physics (state)
physics
ADC DAC
FPGA
µP
µP
comms
comms
µP
µP
FPGA
ADC DAC
physics
network
physics
Physics (structure)


Information
Electronics
Network
Physics


MICREL
Information
Electronics
Network
Physics
Information
Electronics
Network
Physics


MICREL
Information
Information
Electronics
Electronics
Network
Network
Physics
Physics



Information
Information
Electronics
Electronics
Open but tightly coupled!
Network
Network
Physics
Physics

Argentinian director Fernando Livschitz of Black Sheep Films transforms a busy intersection into a choreographed dance by cloning cars, bikes, and people.

https://youtu.be/ufK2XRGUjuc

A smart emergency response system

A feature classification

Evolving architectures

Future value drivers

Connected

Exploit distributed information resources

Reliably configure features with varying quality of service

Assemble available functionality into features after deployment

Data sharing

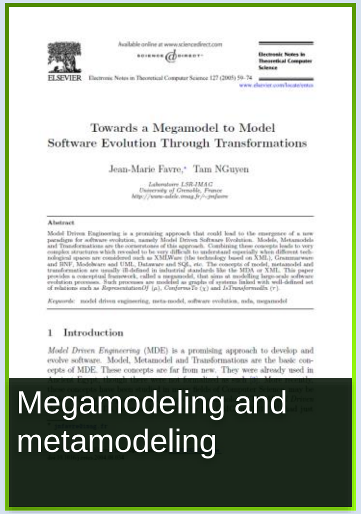
Wireless communication

Service utilization

Connected

Exploit distributed information resources

Technology challenges in CPS



Megamodeling and metamodeling

Reliably configure features with varying quality of service

Assemble available functionality into features after deployment

Data sharing

Wireless communication

Service utilization

Multirate architectures

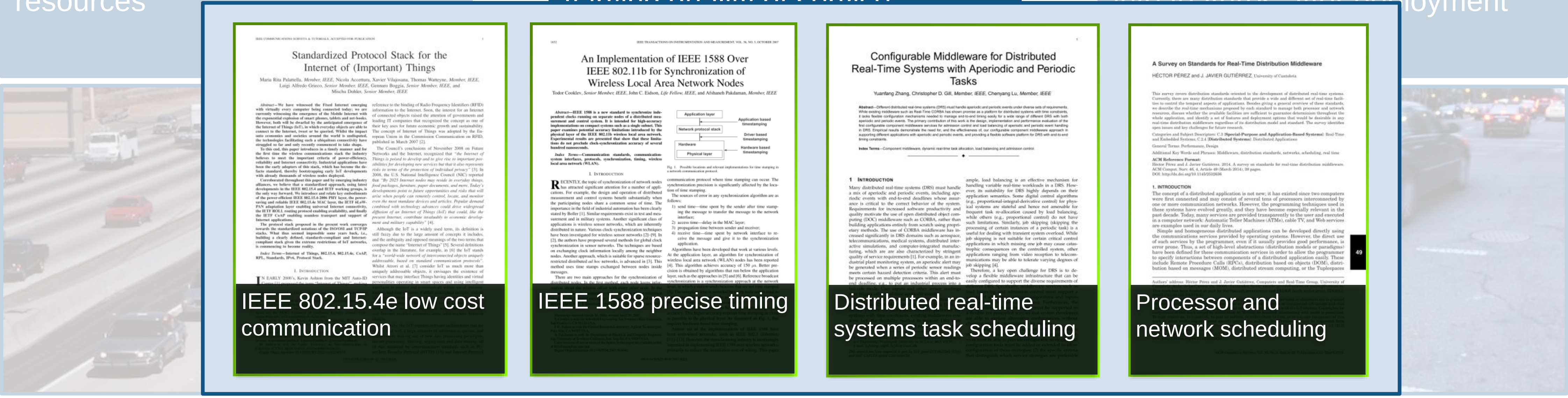
Extracting and deriving specific value from general information

Connected

Exploit distributed information resources

Reliably configure features with varying quality of service

Assemble available functionality into features after deployment



Data sharing

Multirate architectures

Extracting and deriving specific value from general information

Wireless communication

Physically aware configurable protocol stack that is IP compatible

Precise timing and synchronization in a distributed environment

Service utilization

Connected

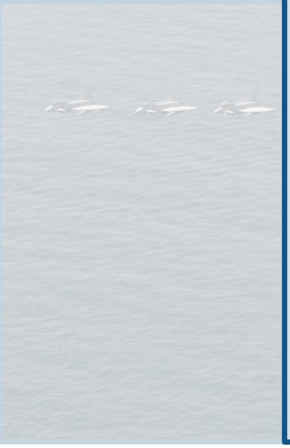
Exploit distributed information resources

Data sharing

Multirate architectures

Extracting and deriving specific value from general information

Reliably configure features with varying quality of service

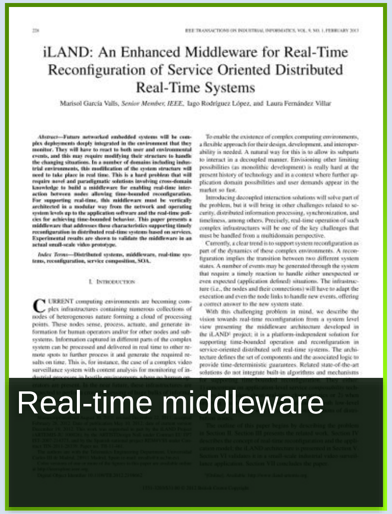


Wireless communication

Physically aware configurable protocol stack that is IP compatible

Precise timing and synchronization in a distributed environment

Assemble available functionality into features after deployment



iLAND: An Enhanced Middleware for Real-Time Reconfiguration of Service Oriented Distributed Real-Time Systems

Marisol García Valls, Senior Member, IEEE, Iago Rodríguez López, and Laura Fernández Villar

A Taxonomy of Discovery Services and Gap Analysis for Ultra-Large Scale Systems

Joe Hoffert, Shanshan Jiang, and Douglas C. Schmidt
Institute for Software Integrated Systems
Vanderbilt University
2015 Terrace Place
Nashville, TN 37212

Semantic Middleware for the Internet of Things

Zhexuan Song, Alvaro A. Cárdenas, Ryusuke Masuoka
Fujitsu Laboratories of America, Inc.

Service utilization

Real-time embedded services operating in a physical environment

Smart services discovery

Information sharing in a heterogeneous system ensemble

Collaborative

Reliably configure an ensemble online to exploit exogenous functionality

Runtime system adaptation

Contract out endogenous resources and balance use of exogenous resources

Hardware resource sharing

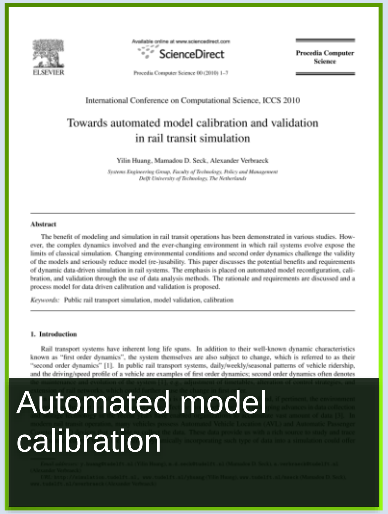
Purpose functionality to create novel system features post deployment

Functionality sharing

Collaborative

Reliably configure an ensemble online to exploit exogenous


Mechanisms for Leveraging Models at Runtime in Self-adaptive Software


ScienceDirect
International Conference on Computational Science, ICCS 2010
Towards automated model calibration and validation in rail transit simulation
Yilin Huang, Mamadou D. Seck, Alexander Verbraeck

Runtime system adaptation

Reasoning and planning adaptation of an ensemble of systems

Contract out endogenous resources and balance use of exogenous resources

Hardware resource sharing

Purpose functionality to create novel system features post deployment

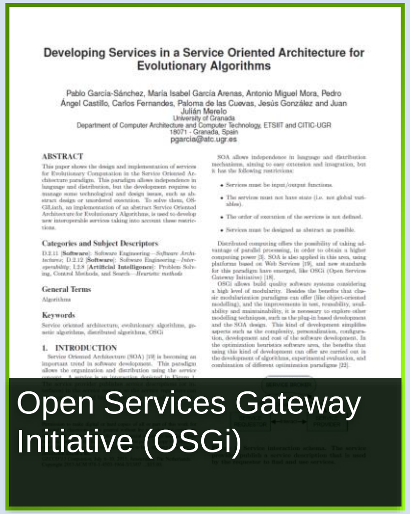
Functionality sharing

Collaborative

Reliably configure an ensemble online to exploit exogenous functionality

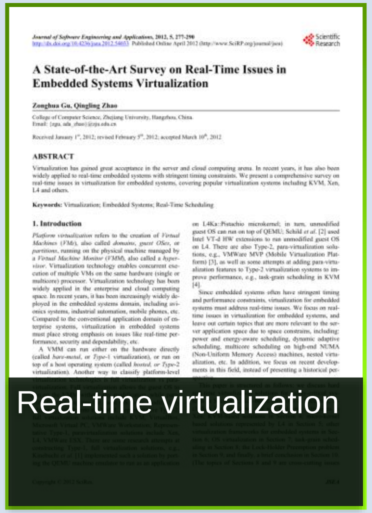
Contract out endogenous resources and balance use of

Purpose functionality to create novel system features post ...ent


Developing Services in a Service Oriented Architecture for Evolutionary Algorithms

Pablo García-Sánchez, María Isabel García Arenas, Antonio Miguel Mora, Pedro Ángel Castillo, Carlos Fernandes, Paloma de las Cuevas, Jesús González and Juan Julián Merelo

Open Services Gateway Initiative (OSGi)


A State-of-the-Art Survey on Real-Time Issues in Embedded Systems Virtualization

Zonghua Gu, Qingling Zhao

Real-time virtualization


A Next-Generation Design Framework for Platform-Based Design

Platform-based design

Runtime system adaptation

Reasoning and planning adaptation of an ensemble of systems

Hardware resource sharing

Flexible and transferable embedded functionality dispatch

Performance characterization from abstract functionality

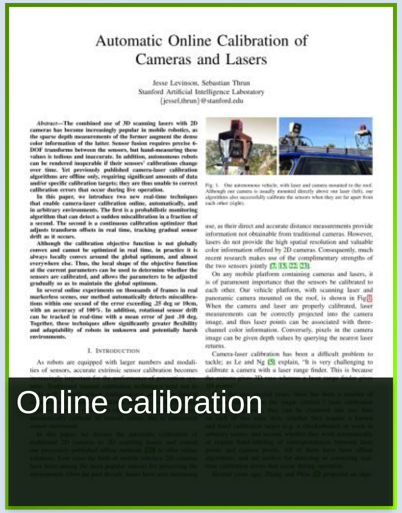
Functionality sharing

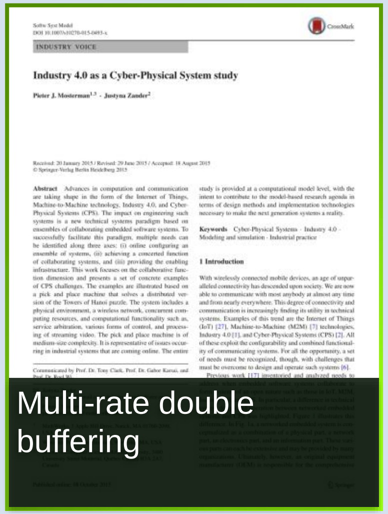
Collaborative

Reliably configure an ensemble online to exploit exogenous functionality

Contract out endogenous resources and balance use of

Purpose functionality to create novel system features post





Runtime system adaptation

Reasoning and planning adaptation of an ensemble of systems

Hardware resource sharing

Flexible and transferable embedded functionality dispatch

Performance characterization from abstract functionality

Functionality sharing

Multi-use functionality post-deployment

Feature interaction

Design

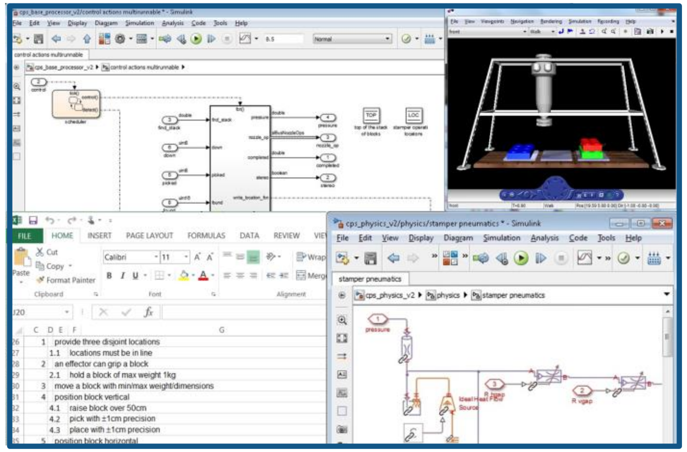
Confidently design systems as part of a reliable ensemble

Virtual system integration

Systematically design optimal behavior of system ensembles

Emerging behavior design

Collaborate between stakeholders throughout the system life cycle



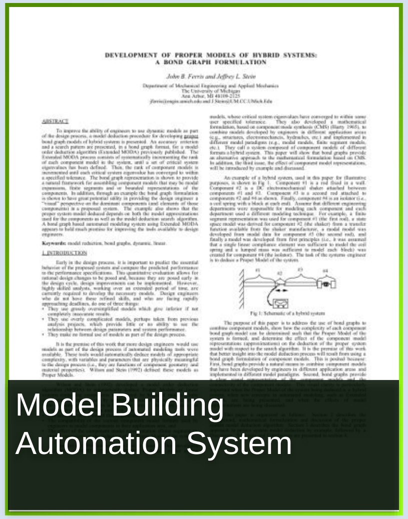
Design artifact sharing

Design

Confidently design systems as part of a reliable ensemble

Systematically design optimal behavior of system ensembles

Collaborate between stakeholders throughout the


DEVELOPMENT OF PROPER MODELS OF HYBRID SYSTEMS: A BOND GRAPH FORMULATION

Model Building Automation System


Counterexample-Guided Abstraction Refinement *

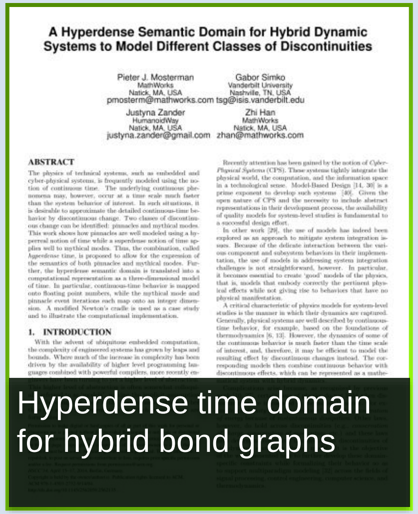
Counterexample guided abstraction refinement


Computer-Automated Multiparadigm Modeling in Control Systems Technology

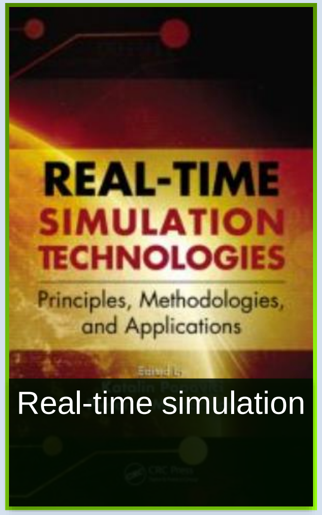
Multiparadigm modeling


An Overview of Hybrid Simulation Phenomena and Their Support by Simulation Packages

Hybrid dynamic systems


A Hyperdense Semantic Domain for Hybrid Dynamic Systems to Model Different Classes of Discontinuities

Hyperdense time domain for hybrid bond graphs



Real-time simulation

Virtual system integration

Proper models in design

System-level design and analysis by using models

Connectivity among models, software, and hardware

Emerging behavior design

Design artifact sharing

Design

Confidently design systems as part of a reliable ensemble

Systematically design optimal behavior of system ensembles

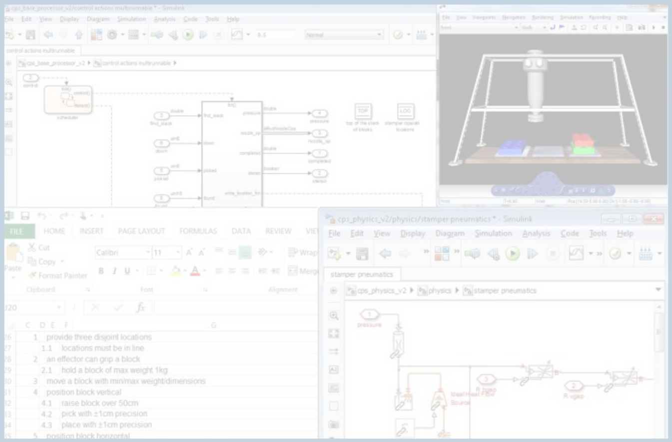
Collaborate between stakeholders throughout the system life cycle


Technical Report MSR-TR-2005-28, February, 2005
Towards Service-Oriented Networked Embedded Computing
Jie Liu and Feng Zhao
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA

Service oriented sensor programming


Machine ballets don't need conductors
Towards scheduling-based service choreographies in a real-time SOA for industrial automation

Service orchestration



Virtual system integration

Proper models in design

System-level design and analysis by using models

Connectivity among models, software, and hardware

Emerging behavior design

Collaborative planning, guidance, and control

Design artifact sharing

Design

Confidently design systems as part of a reliable ensemble

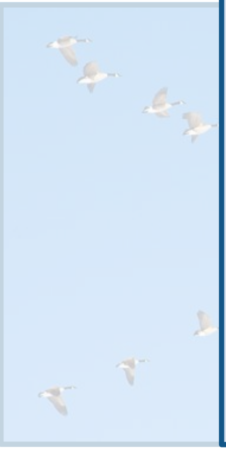
Virtual system integration

Proper models in design

System-level design and analysis by using models

Connectivity among models, software, and hardware

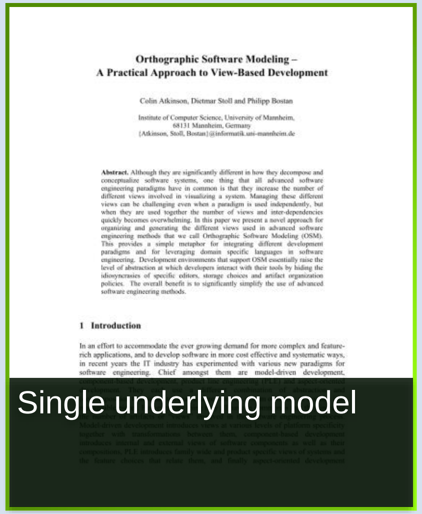
Systematically design optimal behavior of system ensembles

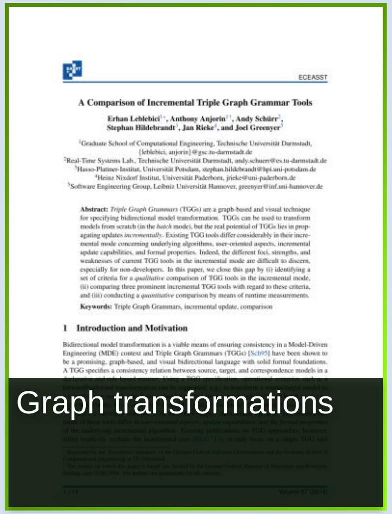


Emerging behavior design

Collaborative planning, guidance, and control

Collaborate between stakeholders throughout the





Design artifact sharing

Tool coupling among disparate organizations

Support manifold views and tools in design

Softw Syst Model
DOI 10.1007/s10270-015-0469-x

INDUSTRY VOICE

Cyber-physical systems challenges: a needs analysis for collaborating embedded software systems

Pieter J. Mosterman[1] · **Justyna Zander**[2]

© Springer-Verlag Berlin Heidelberg 2015

Abstract Embedding computing power in a physical environment has provided the functional flexibility and performance necessary in modern products such as automobiles, aircraft, smartphones, and more. As product features came to increasingly rely on software, a network infrastructure helped factor out common hardware and offered sharing functionality for further innovation. A logical consequence was the need for system integration. Even in the case of a single original end manufacturer who is responsible for the final product, system integration is quite a challenge. More recently, there have been systems coming online that must perform system integration even after deployment—that is, during operation. This has given rise to the cyber-physical systems (CPS) paradigm. In this paper, select key enablers for a new type of system integration are discussed. The needs and challenges for designing and operating CPS are identified along with corresponding technologies to address the challenges and their potential impact. The intent is to contribute to a model-based research agenda in terms of design methods, implementation technologies, and organization challenges necessary to bring the next-generation systems online.

Communicated by Tony Clark, Gabor Karsai, and Roel J. Wieringa.

✉ Justyna Zander
justyna.zander@gmail.com; dr.justyna.zander@ieee.org

Keywords Cyber-physical systems · Computation · Embedded systems · Challenges · Internet of Things · Modeling and simulation

1 Motivation

Engineered systems rely on ingenuity and technology to implement a desired functionality, examples of which include aircraft, automobiles, power plants, smartphones, robots, washers and dryers, pacemakers, and more. Embedded systems are engineered systems that implement functionality by employing computational technologies. The embedded nature allows the computational elements to interact directly (i) with a physical computing platform that it executes on and (ii) with its physical surroundings. In other words, computational logic may obtain input from sensors that measure physical quantities, execute physical instructions of a computing platform to compute output from this input, and provide the output to actuators that effect change in physical quantities and affect the physical behavior.

The intent of this paper is to explore the maturation of embedded systems and the evolution of the concept of cyber-physical systems (CPS). A result of this exploration is the identification of challenges specific to systems of a CPS nature. The perspective reflects upon an industry vantage point. Focus is on models for solving industry-relevant challenges when developing next-generation software systems. While the material is intended to be accessible to the

Pieter J. Mosterman and Justyna Zander, “Cyber-physical systems challenges: a needs analysis for collaborating embedded software systems,” in *Software & Systems Modeling*, Springer Berlin/Heidelberg, ISSN 1619-1366, vol. 15, nr. 1, pp. 5-16, 2016

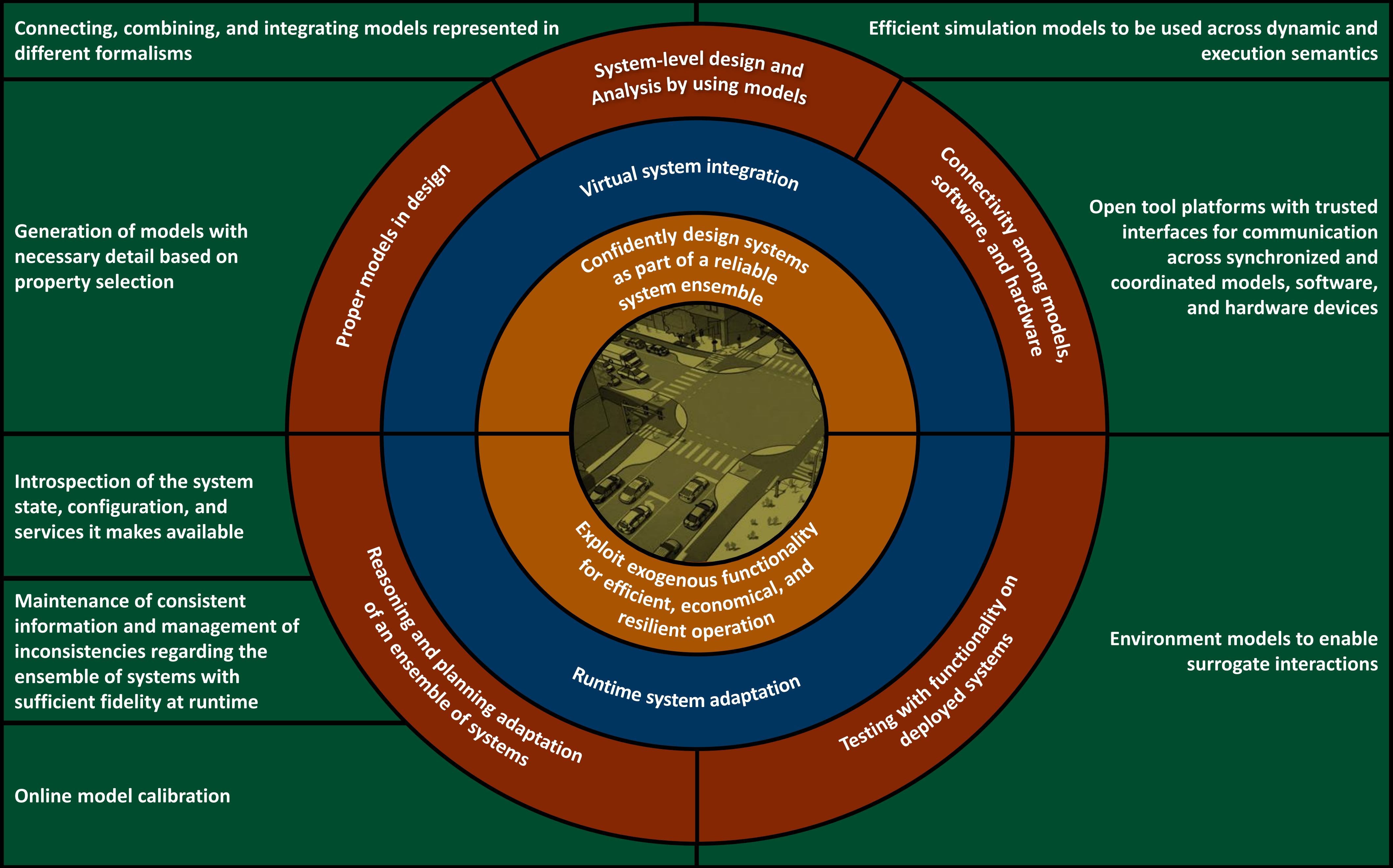
- A smart emergency response system
- A feature classification
- Evolving architectures
- Future value drivers

Conclusions

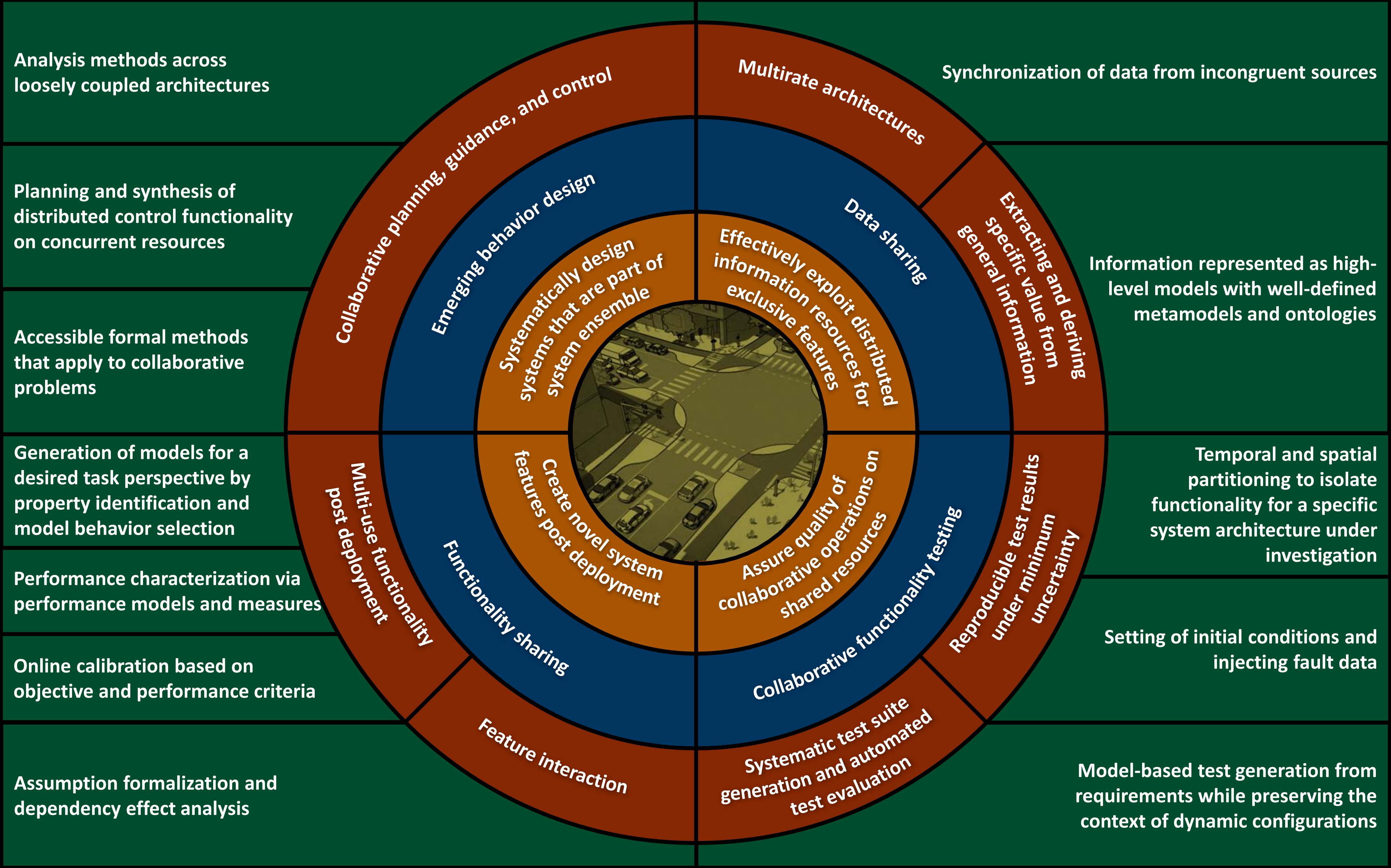
- World is becoming machines that
  - Adapt to the environment
  - Make decisions autonomously
  - Are connected
  - Work together
- Metcalfe is supplanting Moore as value driver
- Key M&S application areas
  - Performance
  - Concurrency
  - Physics
- We need a stupendous range of technologies combined
  - Do not be an individualist!
- MathWorks touches on most any of these technologies
- Come seek us out to discuss!
  - Opportunities
  - Solution needs
  - Starting points

MathWorks®

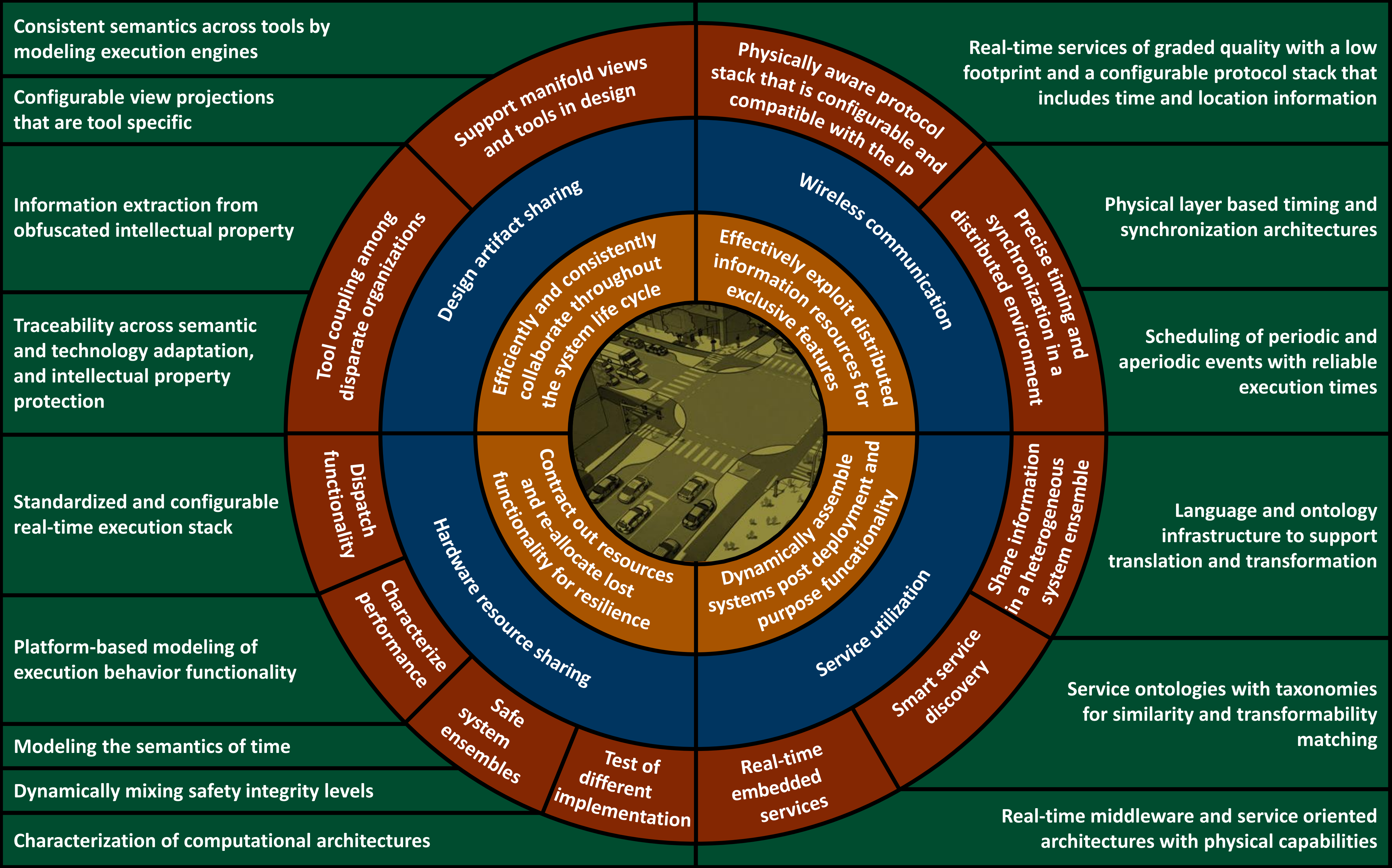
Accelerating the pace of engineering and science



Connecting, combining, and integrating models represented in different formalisms
Efficient simulation models to be used across dynamic and execution semantics
System-level design and Analysis by using models
Proper models in design
Connectivity among models, software, and hardware
Virtual system integration
Confidently design systems as part of a reliable system ensemble
Generation of models with necessary detail based on property selection
Open tool platforms with trusted interfaces for communication across synchronized and coordinated models, software, and hardware devices
Introspection of the system state, configuration, and services it makes available
Maintenance of consistent information and management of inconsistencies regarding the ensemble of systems with sufficient fidelity at runtime
Online model calibration
Exploit exogenous functionality for efficient, economical, and resilient operation
Runtime system adaptation
Reasoning and planning adaptation of an ensemble of systems
Testing with functionality on deployed systems
Environment models to enable surrogate interactions



Analysis methods across loosely coupled architectures
Planning and synthesis of distributed control functionality on concurrent resources
Accessible formal methods that apply to collaborative problems
Generation of models for a desired task perspective by property identification and model behavior selection
Performance characterization via performance models and measures
Online calibration based on objective and performance criteria
Assumption formalization and dependency effect analysis
Synchronization of data from incongruent sources
Information represented as high-level models with well-defined metamodels and ontologies
Temporal and spatial partitioning to isolate functionality for a specific system architecture under investigation
Setting of initial conditions and injecting fault data
Model-based test generation from requirements while preserving the context of dynamic configurations
Collaborative planning, guidance, and control
Multirate architectures
Extracting and deriving specific value from general information
Reproducible test results under minimum uncertainty
Systematic test suite generation and automated test evaluation
Feature interaction
Multi-use functionality post deployment
Emerging behavior design
Data sharing
Collaborative functionality testing
Functionality sharing
Systematically design systems that are part of system ensemble
Effectively exploit distributed information resources for exclusive features
Assure quality of collaborative operations on shared resources
Create novel system features post deployment


Consistent semantics across tools by modeling execution engines
Configurable view projections that are tool specific
Information extraction from obfuscated intellectual property
Traceability across semantic and technology adaptation, and intellectual property protection
Standardized and configurable real-time execution stack
Platform-based modeling of execution behavior functionality
Modeling the semantics of time
Dynamically mixing safety integrity levels
Characterization of computational architectures
Real-time services of graded quality with a low footprint and a configurable protocol stack that includes time and location information
Physical layer based timing and synchronization architectures
Scheduling of periodic and aperiodic events with reliable execution times
Language and ontology infrastructure to support translation and transformation
Service ontologies with taxonomies for similarity and transformability matching
Real-time middleware and service oriented architectures with physical capabilities
Support manifold views and tools in design
Tool coupling among disparate organizations
Dispatch functionality
Characterize performance
Safe system ensembles
Test of different implementation
Physically aware protocol stack that is configurable and compatible with the IP
Precise timing and synchronization in a distributed environment
Share information in a heterogeneous system ensemble
Smart service discovery
Real-time embedded services
Design artifact sharing
Wireless communication
Hardware resource sharing
Service utilization
Efficiently and consistently collaborate throughout the system life cycle
Effectively exploit distributed information resources for exclusive features
Contract out resources and re-allocate lost functionality for resilience
Dynamically assemble systems post deployment and purpose funcationality