http://www.aicto.org

Architecture
for
Cyber-Physical Systems-of-Systems

Ringvorlesung TUD WS 2017/18
Prof. Dr. Frank J. Furrer

Version 1.0

Prof. h.c. Dr. Frank J. Furrer

2015 (July 1): **Professor h.c.** of the Computer Science Department of the Technical University of Dresden (TUD)

2013/14: **Lehrbeauftragter** TUD Dresden

1975 -2011: **Industry-career** in industrial control systems and in system/software architecture for very large IT systems

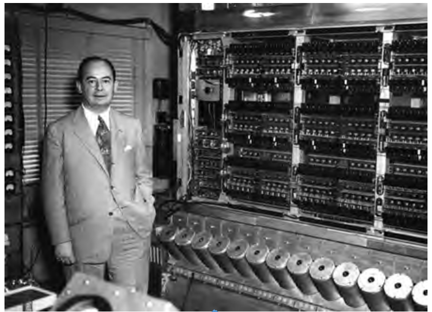
1974: **Ph.D.EE** (Dr. sc. techn. ETHZ) from the Swiss Federal Institute of Technology, Zurich (ETH-Z)

1970: **MS** in Electrical Engineering 1970 from the Swiss Federal Institute of Technology, Zurich (ETH-Z)

1945 (January 27): Born in Switzerland (Zurich)

frank.j.furrer@bluewin.ch
frank.furrer@mailbox.tu-dresden.de

1951: John von Neumann at the Princeton Institute for Advanced Studies [IAS Computer]

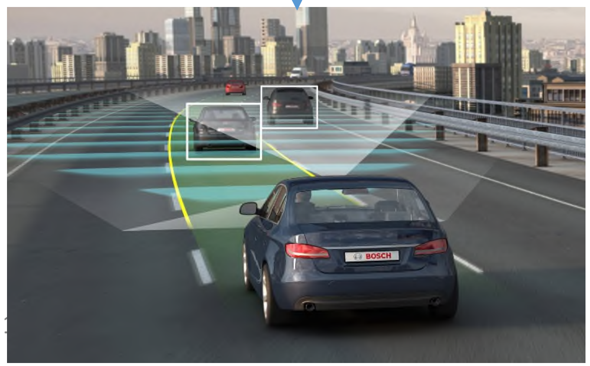


http://www.rutherfordjournal.org

Enterprise Computing

Computing for Control

e-Business



Cyber-Physical Systems

Content:

Part 1: Definitions

- Introduction
- Cyber-Physical Systems (CPS)
- Cyber-Physical Systems-of-Systems (CPSoS)

Part 2: CPSoS Opportunities & Risks

- CPSoS Opportunities
- CPSoS Risks

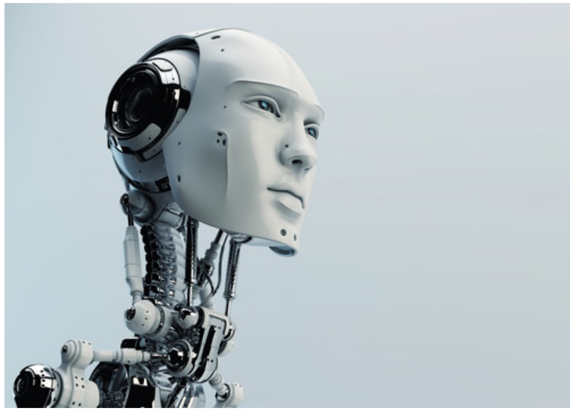
Part 3: CPSoS Architecture for Dependability

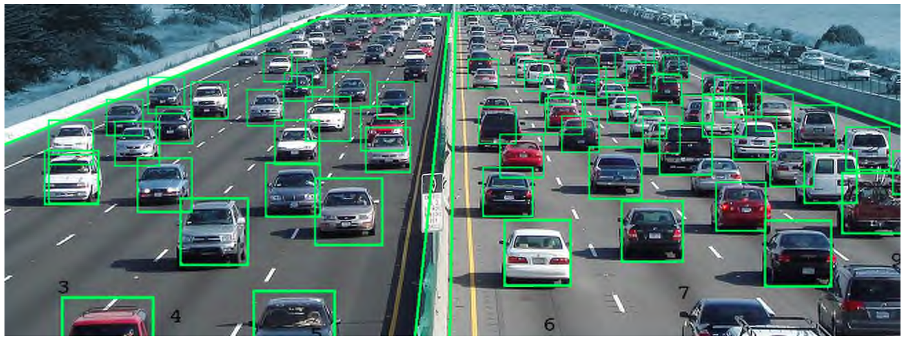
- CPSoS Architecture Framework
- CPSoS Dependability Engineering

Recommended Reading

Introduction

Most of today’s interesting systems are Cyber-Physical Systems-of-Systems (CPSoS)





Most of today's interesting systems are Cyber-Physical Systems-of-Systems (CPSoS)

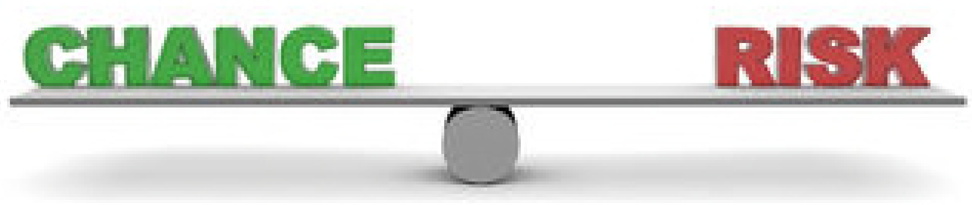
They give us phantastic **chances**

… but also introduce significant **risk**

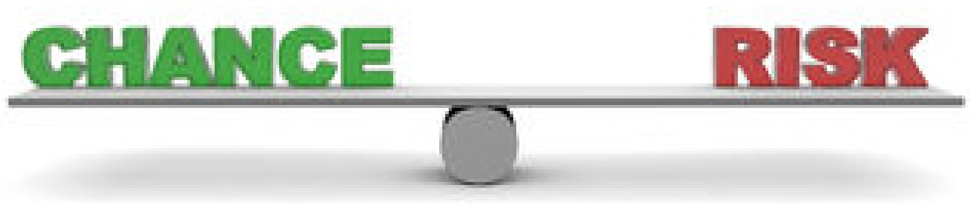
Most of today's interesting systems are Cyber-Physical Systems-of-Systems (CPSoS)

They give us phantastic **chances**

… but also introduce significant **risk**



As systems engineers
we must **ensure**
a fair balance between chances and risks



As systems engineers
we must **ensure**
a fair balance between chances and risks

Today much of the functionality is
implemented in **software**

We must engineer
and implement
Dependable Software

Dependable Software

The strong foundation for Dependable Software are an adequate **system architecture** and proven architecture principles

Architecture for **dependable** cyber-physical systems-of-systems

Cyber-Physical Systems (CPS)

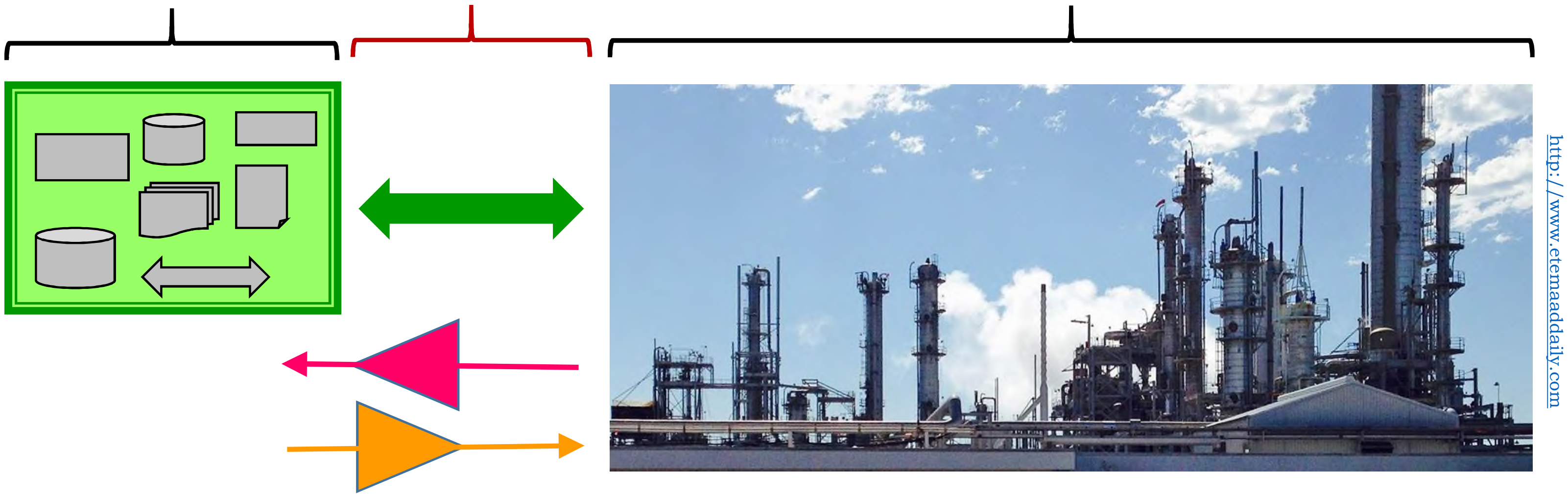
Cyber-Physical System (CPS)

DEFINITIONS

A **cyber-physical system** (CPS) consists of a computing device interacting with the physical world in a feedback loop

[Rajeev Alur, 2015]

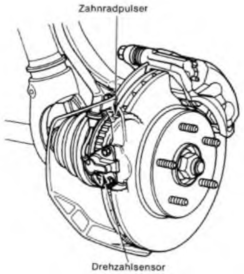
Cyber-part **Interaction** **Physical part**



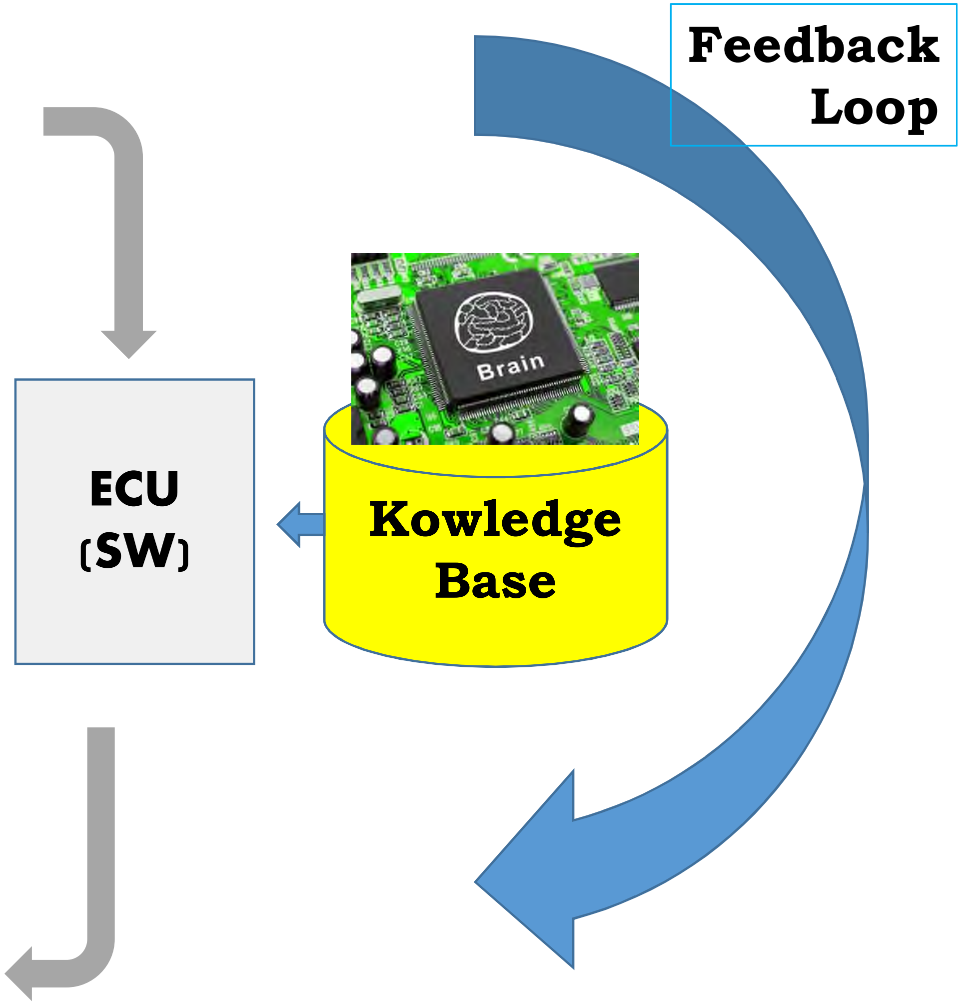
Sensors: Read plant information

Actuators: Control plant

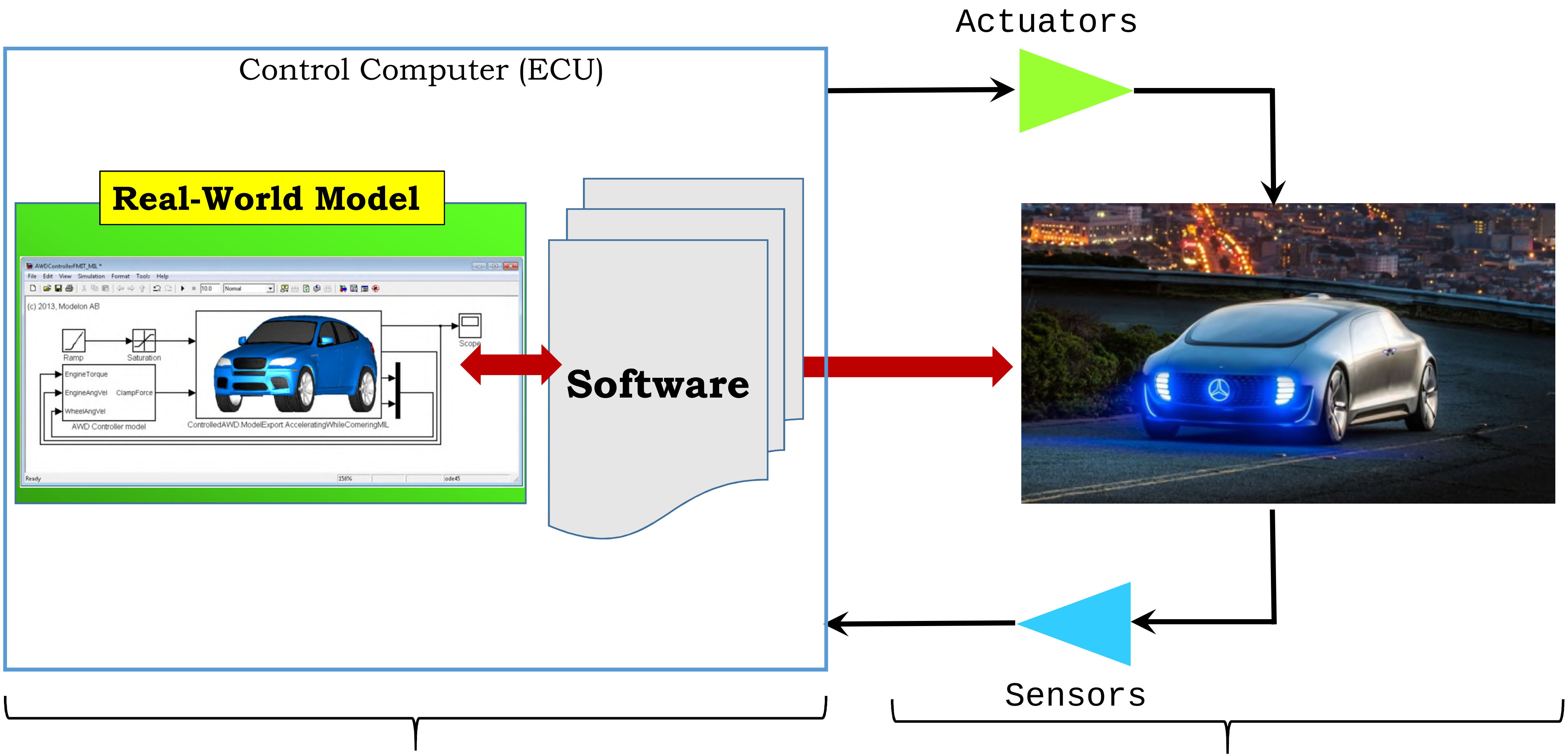
CPS-Example: **ESC**



http://www.polizeiticker.ch



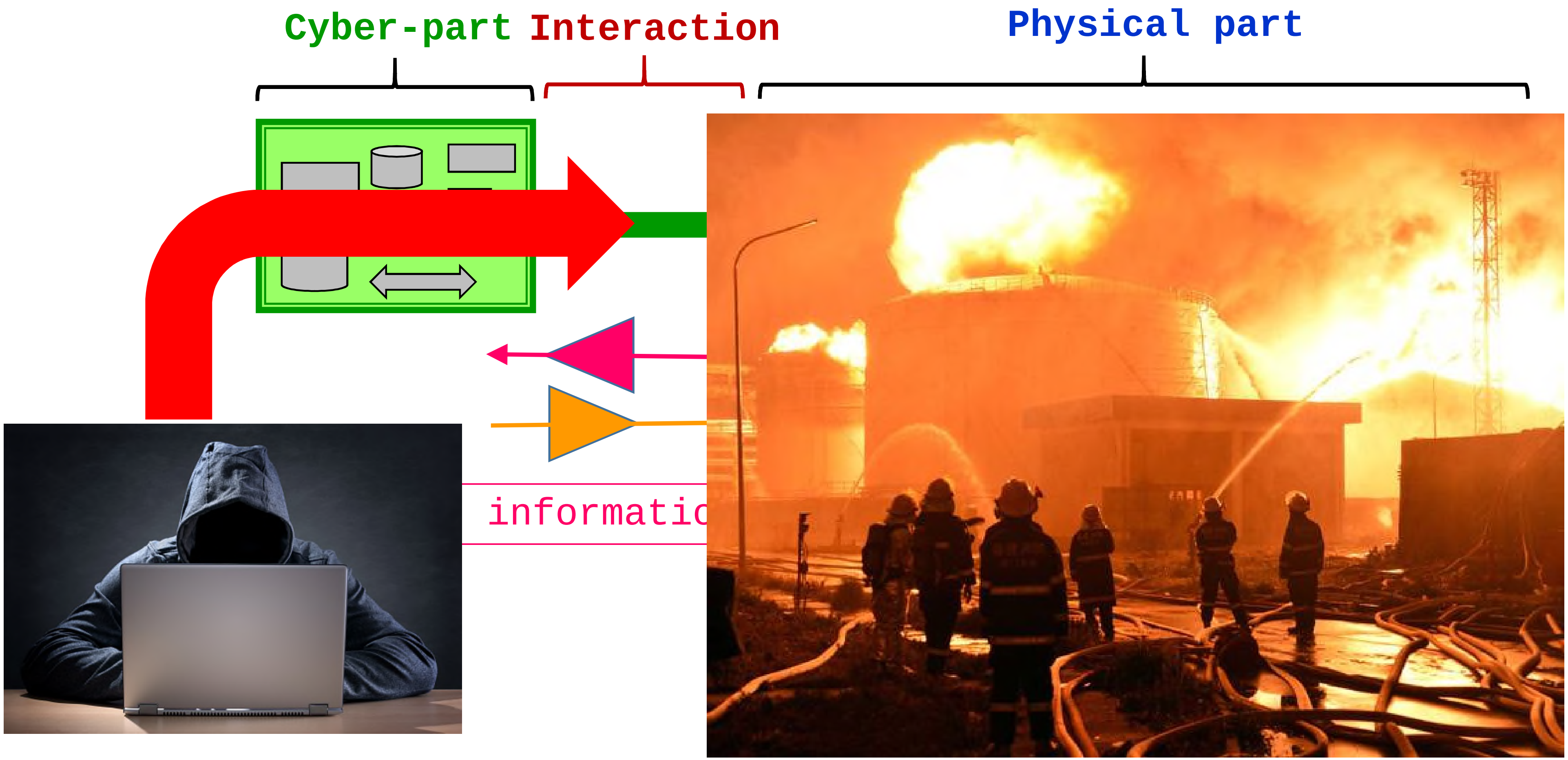
Architecture for Cyber-Physical Systems of Systems (CPSoS)



http://www.modelon.com

http://cdn1.alphr.com

Risk: Cyber-Physical Attacks

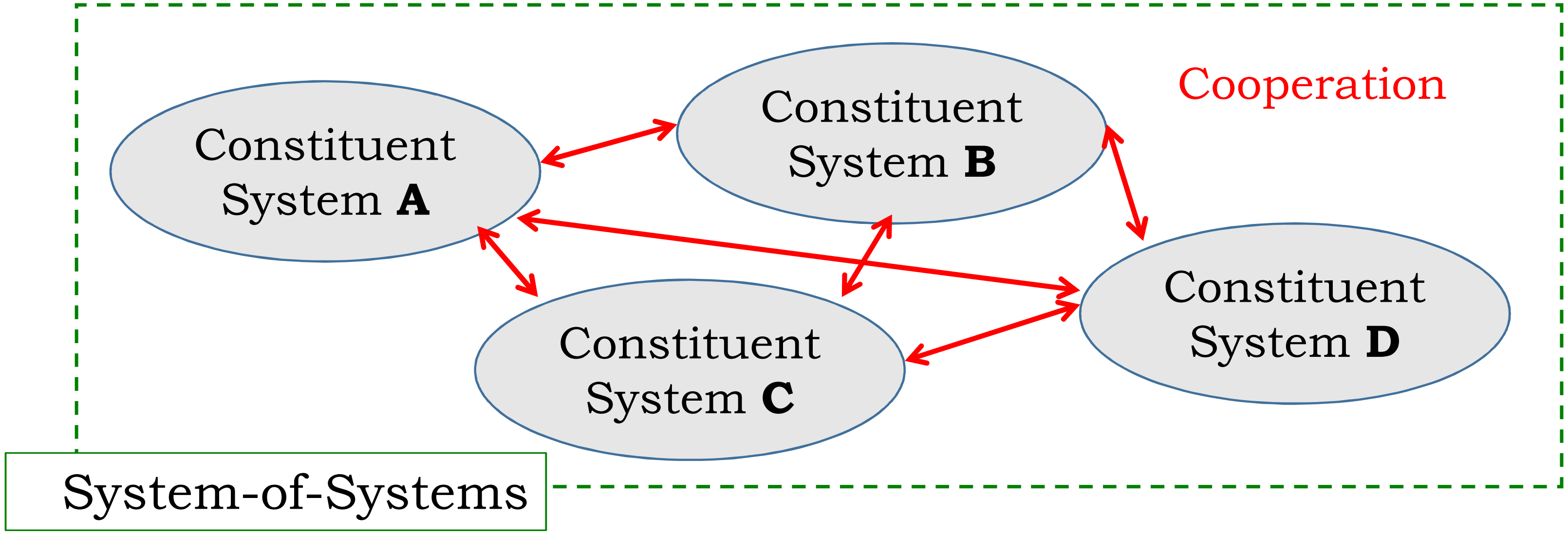


Cyber-Physical Systems-of-Systems (CPSoS)

Architecture for Cyber-Physical Systems of Systems (CPSoS)

Cyber-Physical Systems (CPS) √

Cyber-Physical Systems-of-Systems (CPSoS)



http://www.fotocommunity.de

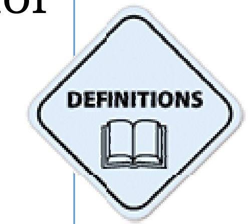
New capability, **emergent property**

System-of-Systems (SoS)

A **system of systems (SoS)** brings together a set of *cooperating systems* for a task that none of the systems can accomplish on its own (= emergent property).

Each constituent system keeps its own management, goals, and resources while coordinating within the SoS and adapting to meet SoS goals.

ISO/IEC/IEEE 15288 Annex G



New capability: emergent *property* & emergent *behaviour*

System A
- Properties «A»
- Behaviour «A»

System B
- Properties «B»
- Behaviour «B»

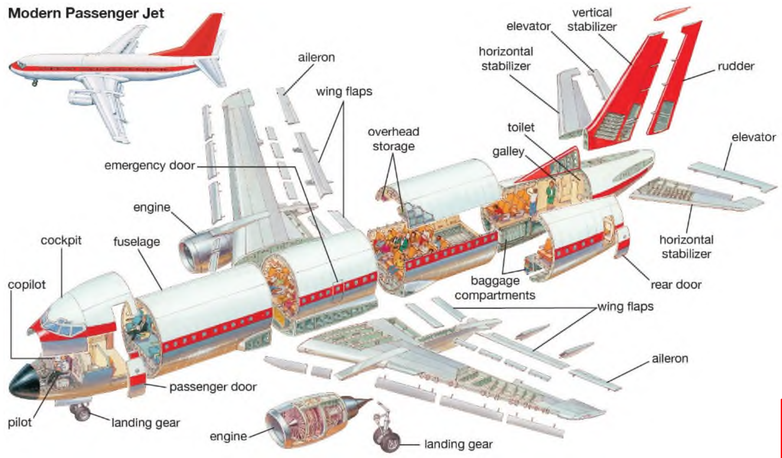
…

System G
- Properties «G»
- Behaviour «G»

Agreed Collaboration: **SoS Goal**

New, desired property or behaviour
NOT
available from the individual constituent systems

Example 1: Emerging Properties: „flying“



http://cashmancuneo.net

<u>Constituent systems (CS) of an aircraft</u>:

- engines
- body
- wings
- cockpit
- etc.

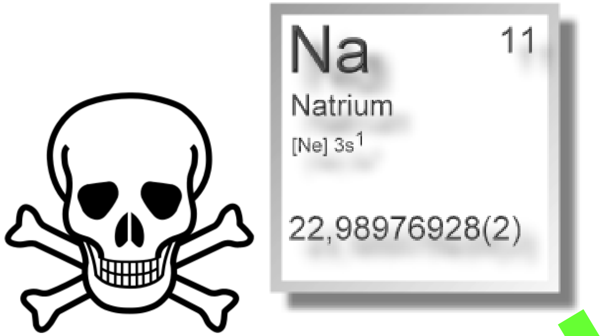
… none of the constituent systems is able to fly !

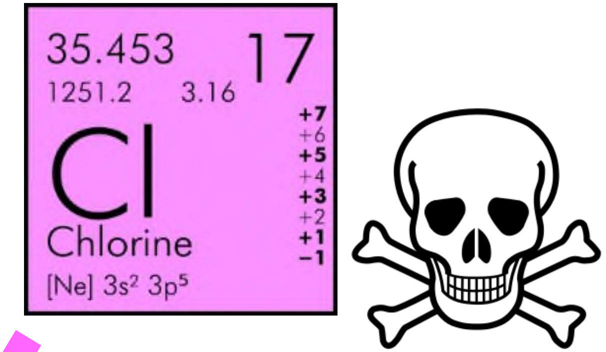
Assemble the essential constituent systems:

Emerging property: the assembly (= airplane) is able to **fly** !

http://en.wikipedia.org/wiki/Airbus_A320_family

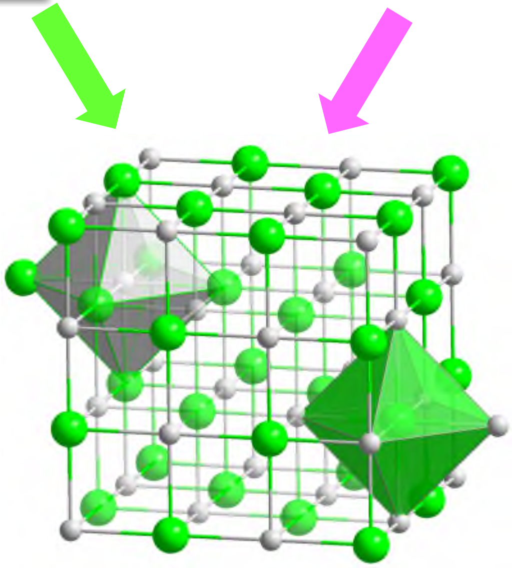
Example 2: Emergent Properties (NaCl)





Constituent systems:
Na
Cl

https://upload.wikimedia.org



Na+ Cl-

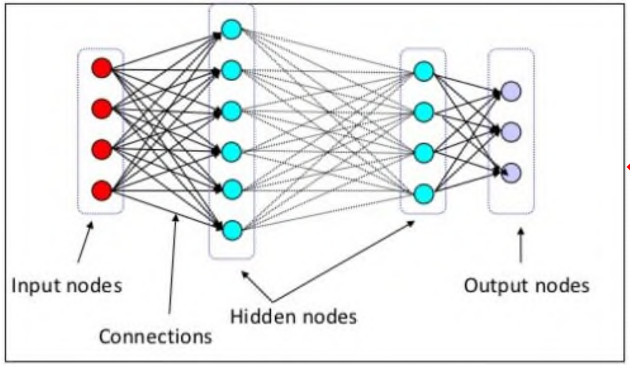
Table Salt
(= Natriumchloride NaCl)

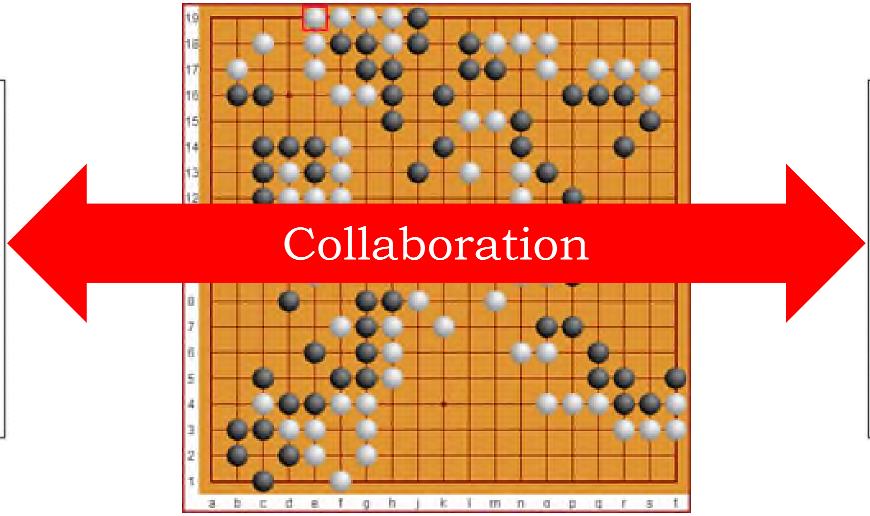
https://de.wikipedia.org

Example 3: Emerging properties: „AlphaGo Zero“

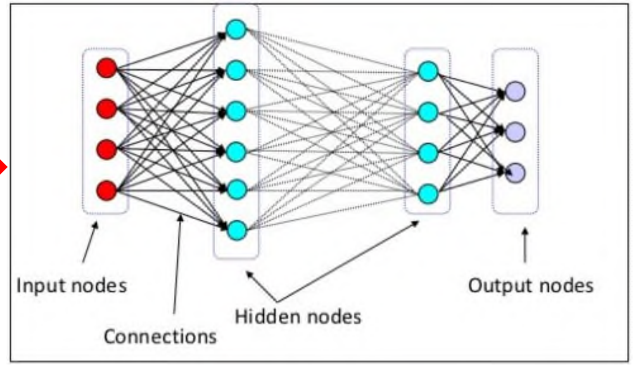
AlphaGo's team published an article in the journal Nature on 19 October 2017, introducing **AlphaGo Zero**, a version created without using data from human games, and stronger than any previous version

Constituent System **A**



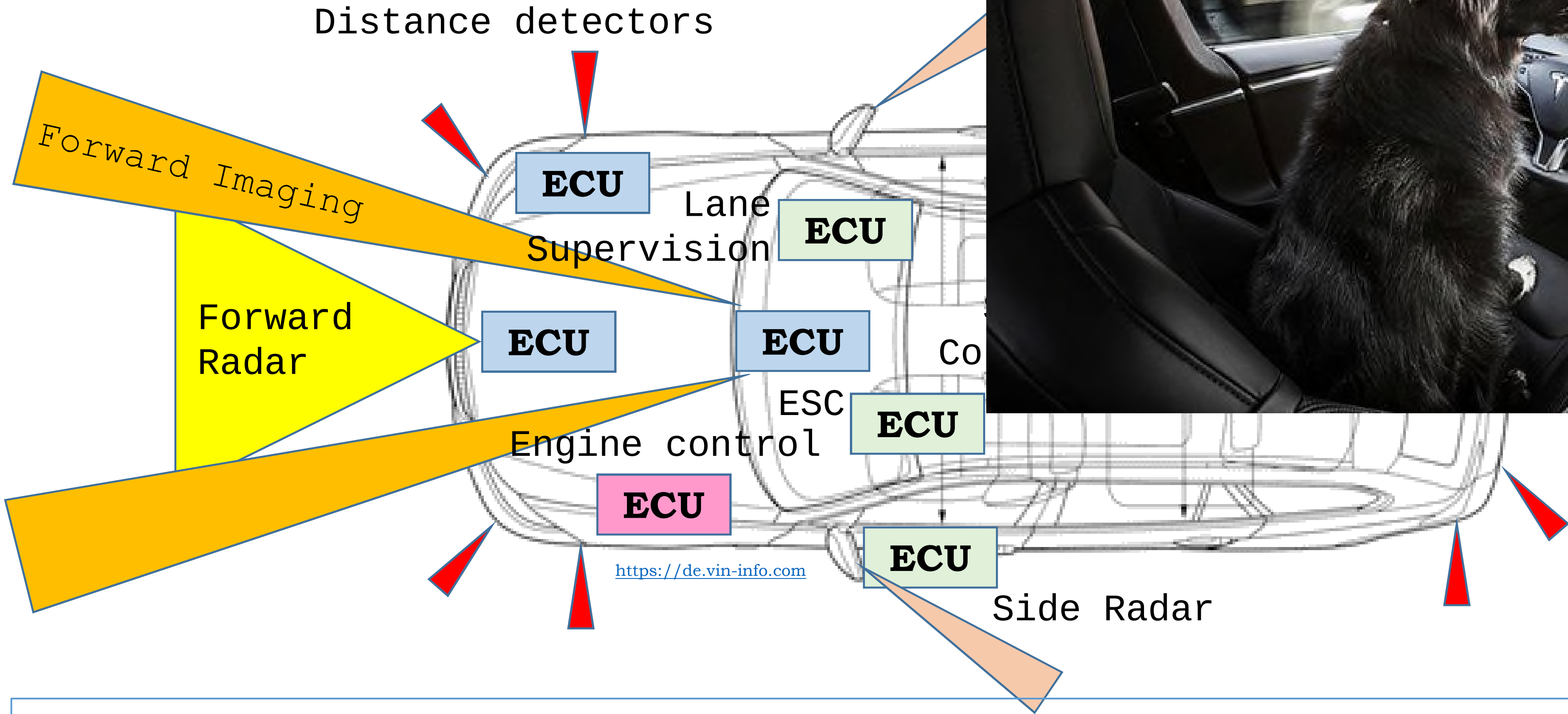


Constituent System **B**



AlphaGo Zero is so powerful because it is "*no longer constrained by the limits of human knowledge*“ [Demis Hassabis, 2017]

Example 4: Emerging Property «Autonomou



https://hips.hearstapps.com

A modern high-end car contains more than 100 networked ECUs (Electronic Control Units)

Example 5: Emerging Property «Runway Accident»

http://avioners.net/2011/03/airbus-a319-approach-to-landing.html

Constituent systems:

- Airplane (DC-8)
- Airport (Runway)

October 8, 1979: Swiss Air Flight 316 overran the Athens runway – 14 deaths

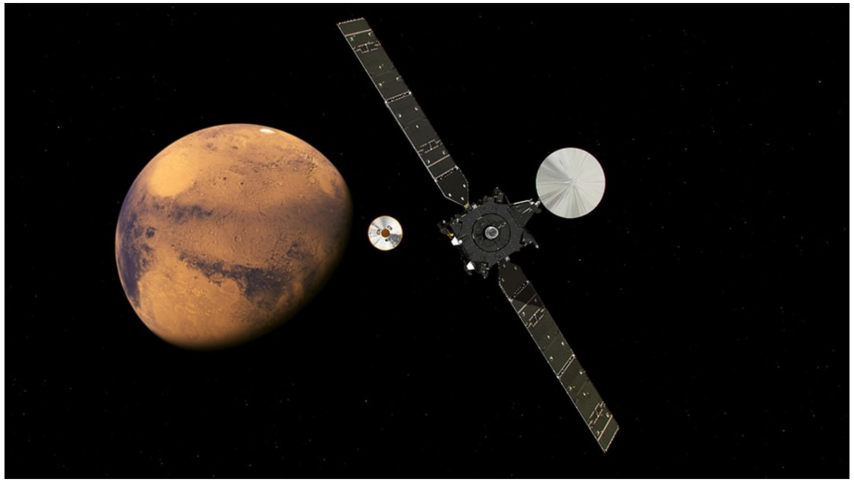
Cause: „Interface“ between the runway and the airplane

- Landing when braking action is less then good
- Crew mistakes

Example 6: Emerging Property «Mars Lander Crash»

19. October 2016:

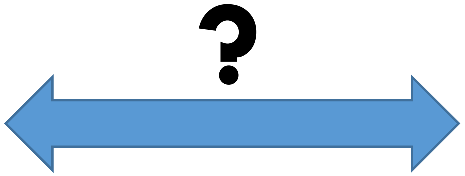
The Mars Lander «**Schiaparelli**» crashes to the ground



http://www.aargauerzeitung.ch

Radar-Altimeter

https://de.wikipedia.org



http://www.militaryaerospace.com

Navigation Computer

Software **Interoperability** problem between Radar-Altimeter and Navigation Computer in the Lander

SoS Emergent Behaviour **Classification**

<table>
<tr><th>Emergence</th><th>Desirable/positive</th><th>Undesirable/negative</th></tr>
<tr><td>Expected emergent behavior</td><td>Reason for building the SoS (SoS objective)
</td><td>Mitigate by appropriate design measures, such as threat/risk analysis and countermeasures

</td></tr>
<tr><td>Unexpected emergent behavior</td><td>Sometimes (however, quite rarely) an SoS shows unexpected, beneficial behaviour
</td><td><u>Un</u>expected & <u>un</u>desirable negative emergent behavior is one of the critical risks of most SoS

</td></tr>
</table>

Mark W. Maier:
SoS Criteria

Monolithic systems ⇔ Systems-of-systems:

1. Operational Independence of the Elements
2. Managerial Independence of the Elements
3. Evolutionary Development
4. Emergent Behaviour
5. Geographic Distribution

[5 Maier criteria, 1998]

Governance

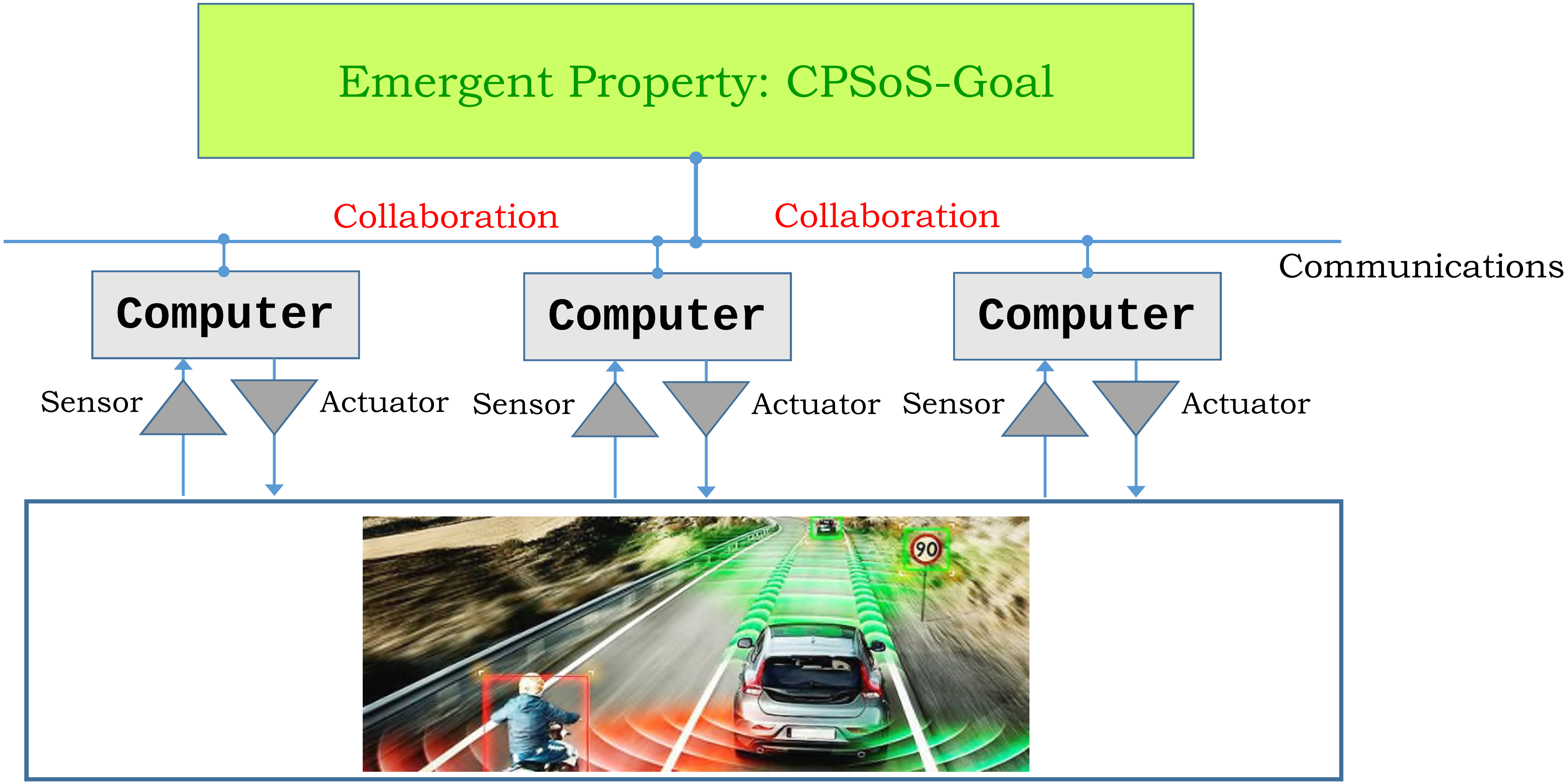
Total = more than the sum of its parts

Collaboration

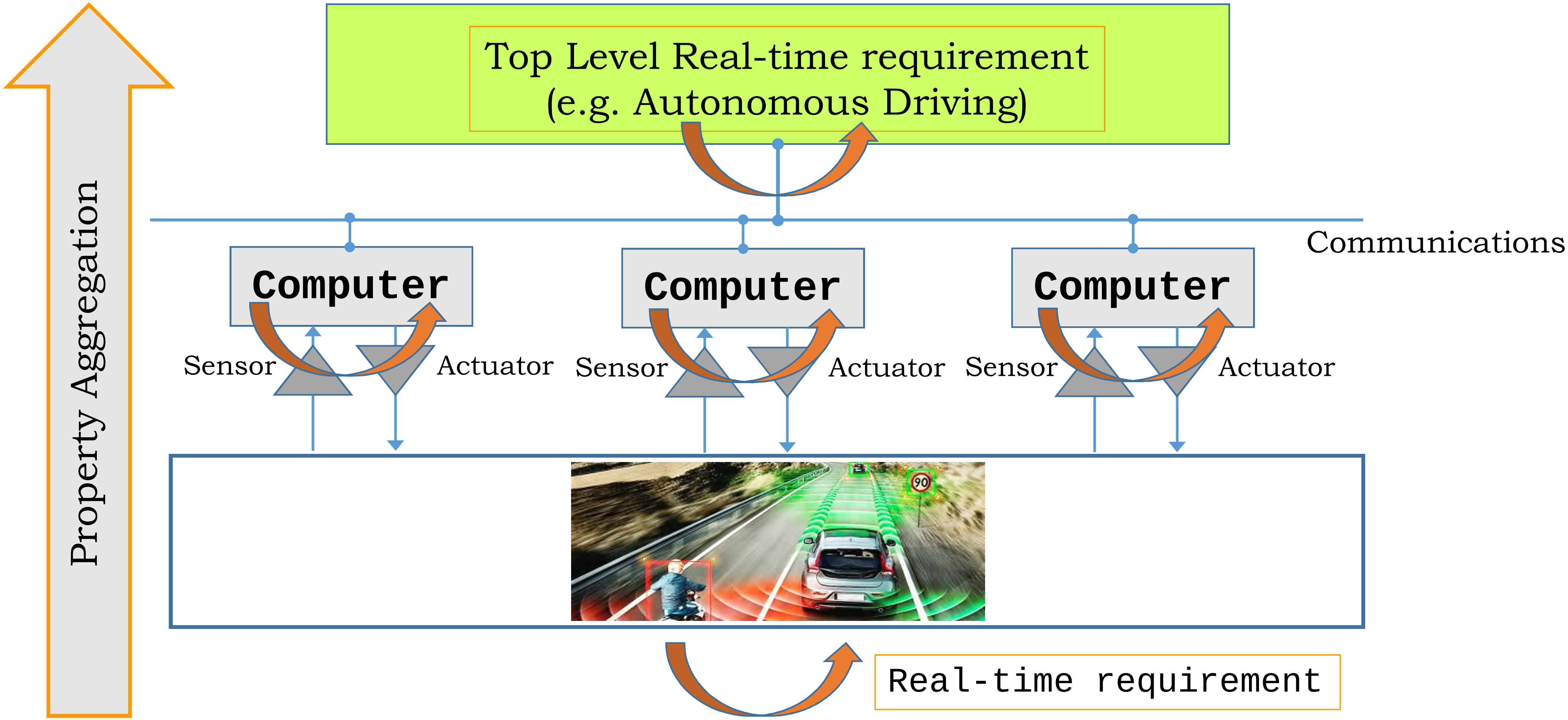
Collaboration Contract

- functional
- operational
- commercial
- legal

Cyber-Physical System-of-Systems (CPSoS)



Architecture for Cyber-Physical Systems of Systems (CPSoS)



CPSoS Example: Roborace [Unmanned Automobile Racing]

Fully electric cars, V_{max} = 300 km/h

https://www.crowdact.com

24 mechanically identical cars / 12 teams / F1-race circuits

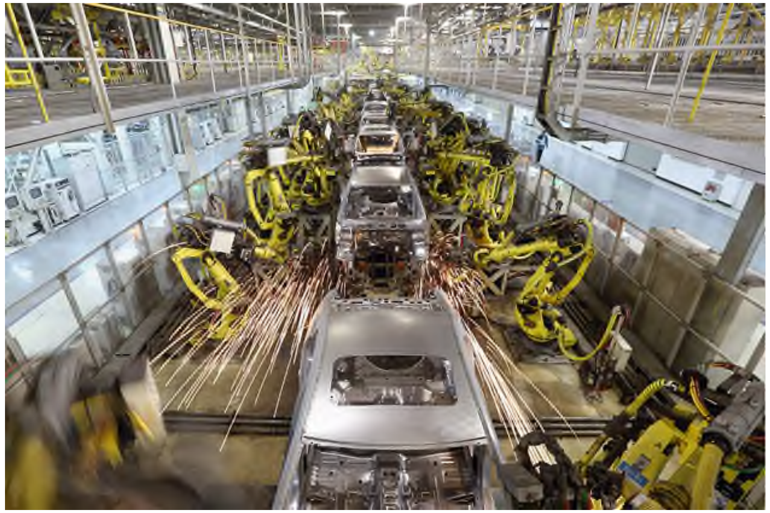
NO drivers

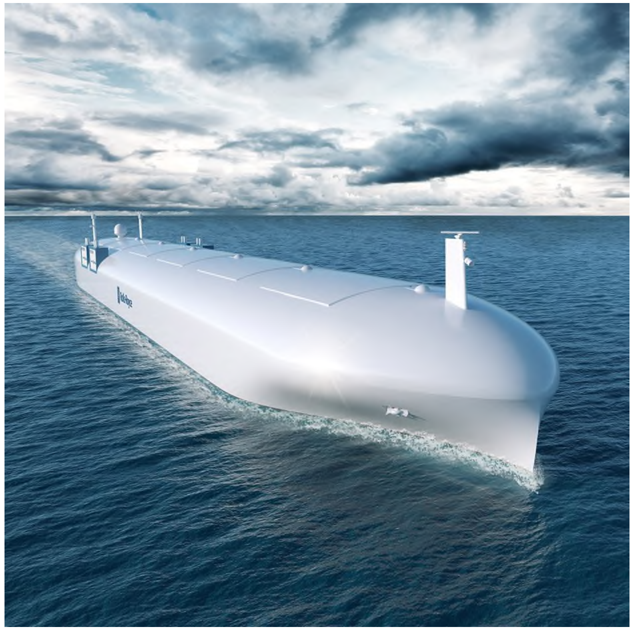
https://wi-images.condecdn.net

Winner: Cognitive and autonomic CPSoS-SW (24-Teraflops-Computers on-board)

CPSoS Opportunities

CPSoS enable tremendous opportunities – *today and tomorrow*



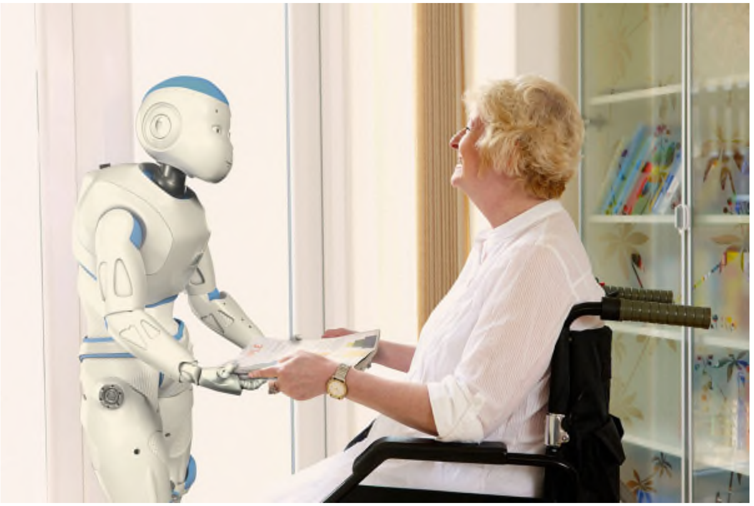


CPSoS allow us to:

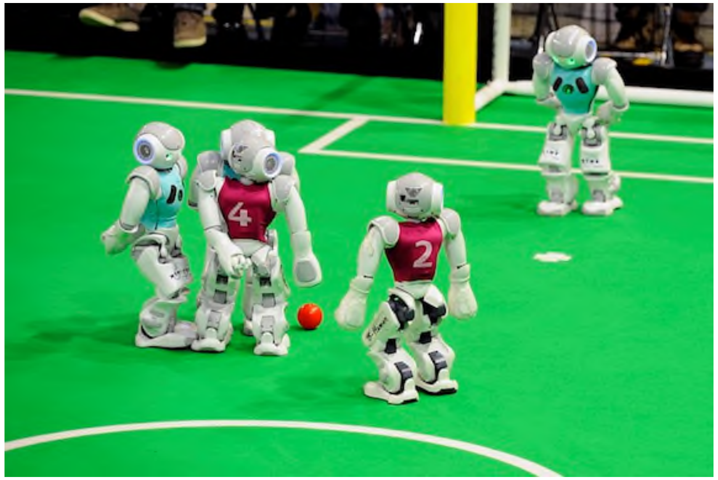
- Build and manage *highly complex* systems
- Build systems which interact and control *real world* systems
- Build and manage systems which could not be managed by people
- Make *optimum usage* of natural and technical resources (Energy, pollution, traffic space, electromagnetic spectrum, water, …
- Increase the *safety* and *security* of technical infrastructure
- Use the support of *artificial intelligence* in real-world situations
- … and more

Humans will have to learn to **cooperate** with CPSoS's:

http://medicalfuturist.com



http://public.media.smithsonianmag.com



https://placester.com

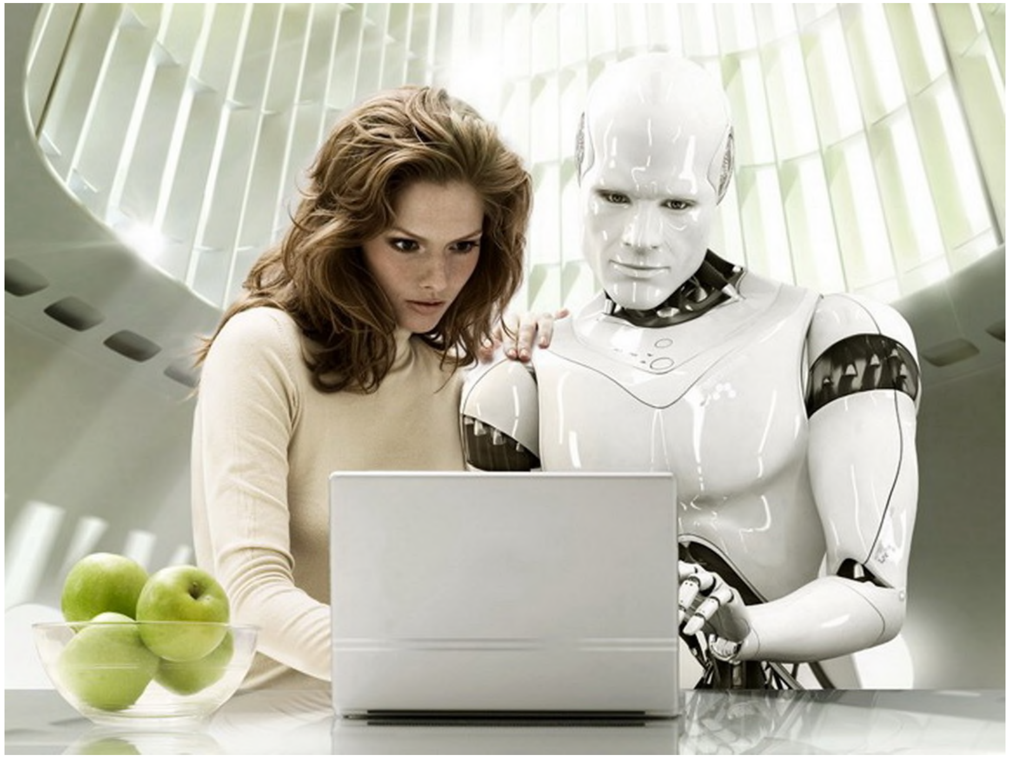
https://i.ytimg.com

https://www.hrw.org

https://thelibertarianrepublic.com

CPSoS's – in various forms – will become **partners** for humans:

http://sickusablog.com



http://files.fbstatic.com

CPSoS Risks

CPSoS Risks

Technical Risks

Blackout

https://s-i.huffpost.com

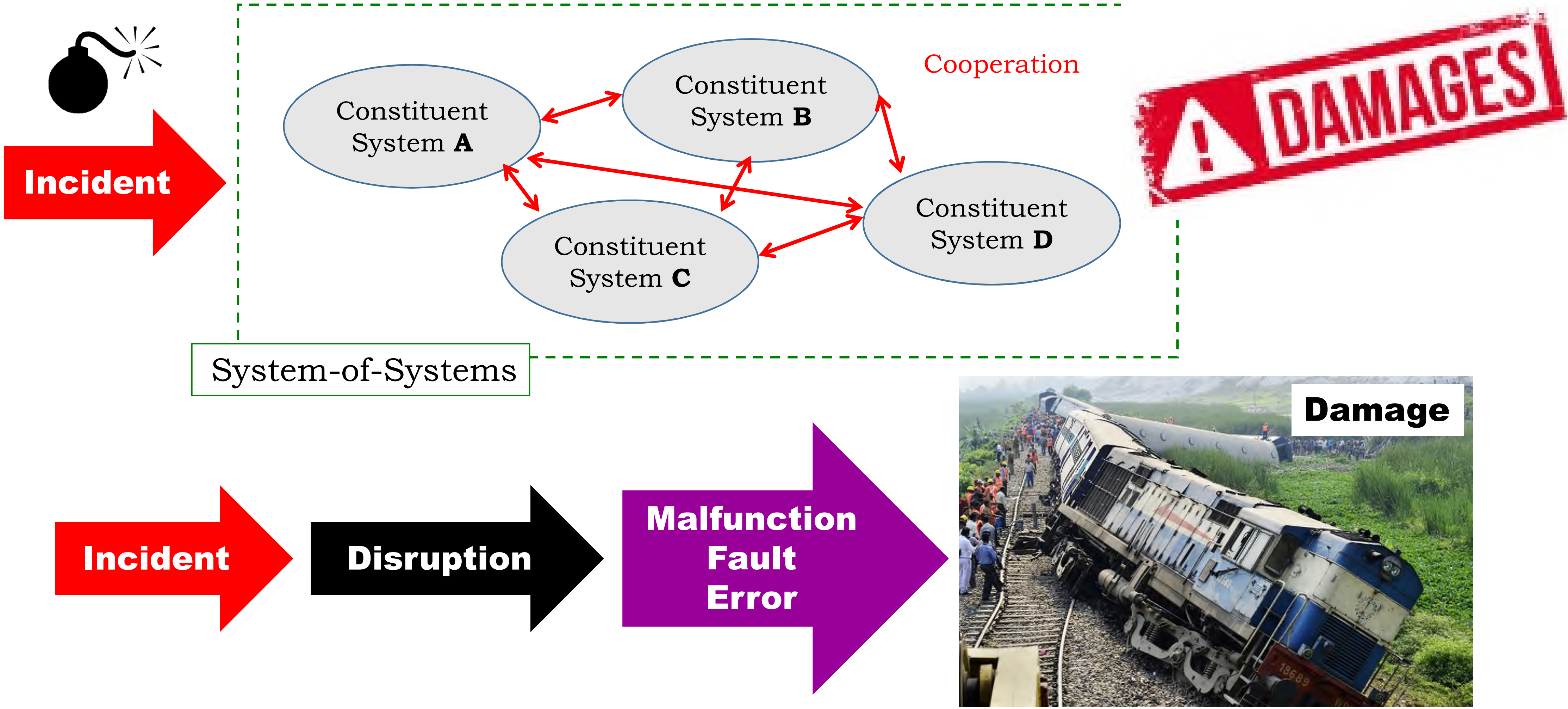
Risks for Society

- Unemployment
- Identity theft
- etc.

http://www.vocfm.co.za

https://specials-images.forbesimg.com

Architecture for Cyber-Physical Systems of Systems (CPSoS)



CPSoS Risk Example 1: Unauthorized Access

theguardian

Thu 31 Aug '17: Hacking risk leads to recall of 500,000 pacemakers due to patient death fears

The FDA says that the *embedded software vulnerability* allows an unauthorised user to access a device using commercially available equipment and reprogram it.

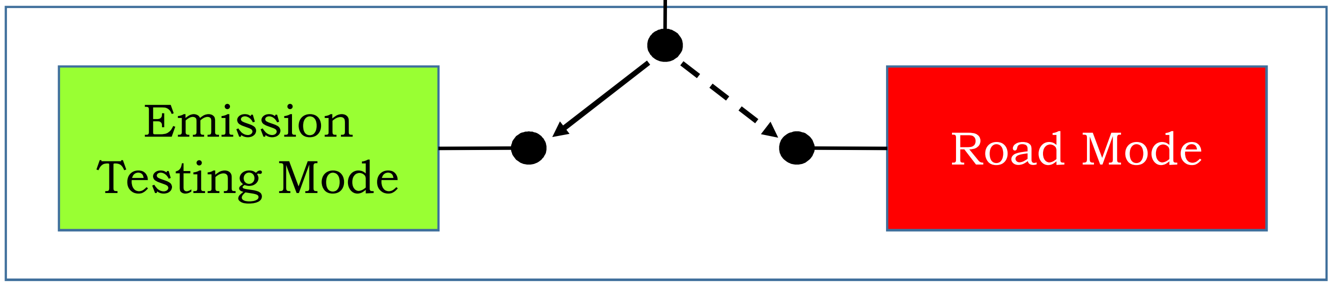
The hackers could then deliberately run the battery flat, or conduct "administration of inappropriate pacing". Both could, in the worst case, result in the death of an affected patient.

https://www.theguardian.com/technology/2017/aug/31/hacking-risk-recall-pacemakers-patient-death-fears-fda-firmware-update

CPSoS Risk Example 2: Emission Cheating

https://assets.bwbx.io

http://www.xpats.com



CPSoS Risk Example 3: NFC intervention

naked **security** by SOPHOS

Android NFC hack lets subway riders evade fares

Near **F**ield **C**ommunications

24 Sep 2012:
Subway riders in the New Jersey and San Francisco transit systems can use near-field communication (NFC) Android smartphones to endlessly *replenish their fare cards for free*, security researchers demonstrated Thursday at the EUSecWest security conference in Amsterdam

https://nakedsecurity.sophos.com/2012/09/24/android-nfc-hack-lets-subway-riders-evade-fares/

CPSoS Risk Example 4: Car hacking

TREND MICRO

16.8.2017: A Deep Flaw in Your Car Lets Hackers Shut Down Safety Features

http://images.guff.com

"*You could disable the air bags, the anti-lock brakes, or the door locks, and steal the car*" says Federico Maggi. Maggi says the attack is stealthier than previous attempts, foiling even the few intrusion detection systems some companies have promoted as a way to head off car hacking threats. "It's practically impossible to detect at the moment with current technology" he says.

https://link.springer.com/chapter/10.1007/978-3-319-60876-1_9

CPSoS Risk Example 5: Water poisoning threat

Water Online

30.3.2016: Hackers Infiltrate Water Plant, Modify Chemical Levels

https://d.ibtimes.co.uk

Hackers infiltrated the control system at a *water treatment plant* and managed to *manipulate the level of chemicals* being used at the facility

The fallout from the hack was not as bad as it could have been. The water company reversed chemical and flow changes before any customers became ill

https://www.wateronline.com/doc/hackers-infiltrate-water-plant-modify-chemical-levels-0001

CPSoS Risk Example 6: Airplane Hacking

WIRED

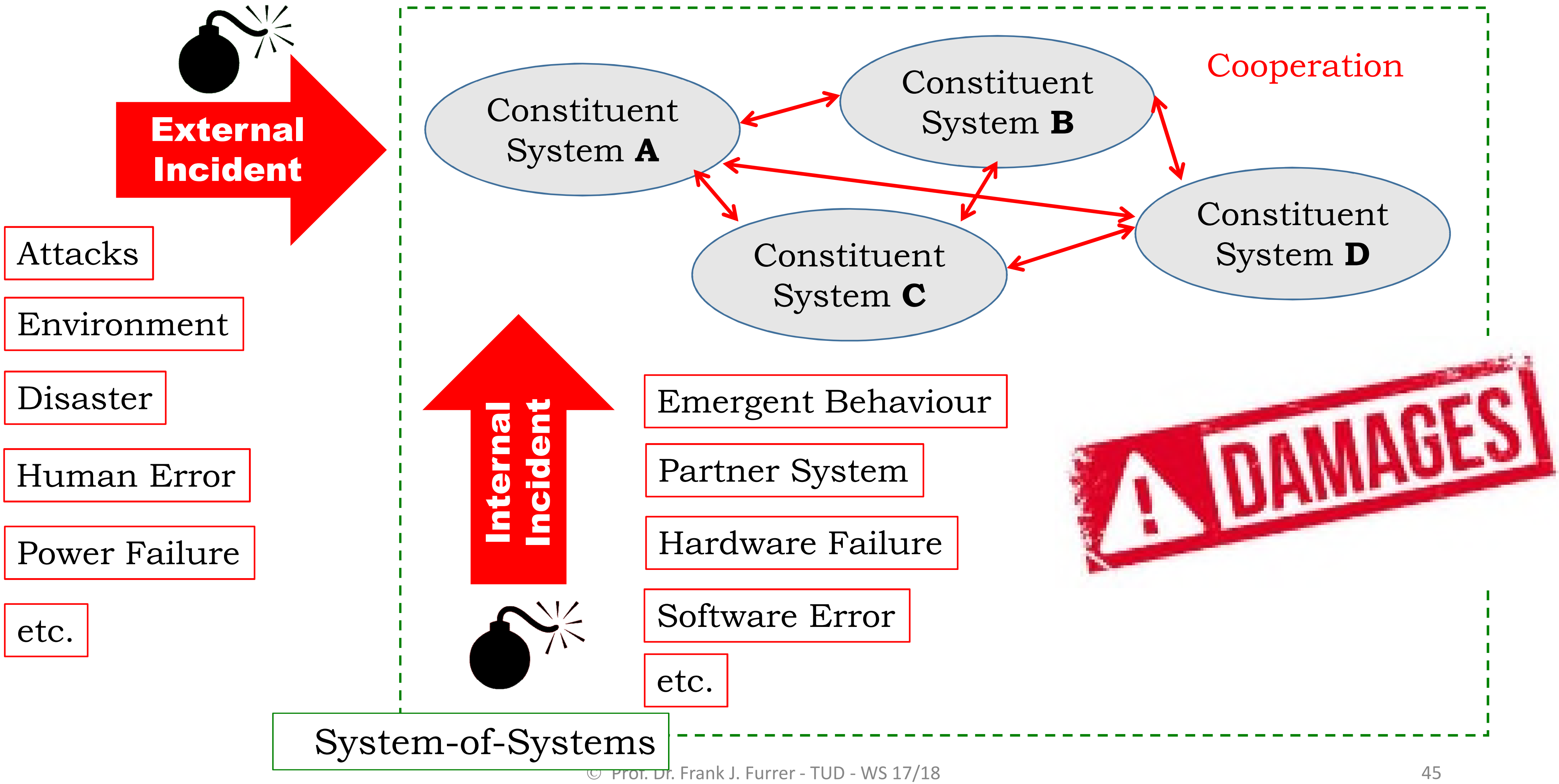
http://i.dailymail.co.uk

https://www.ianvisits.co.uk

15.4.2015:

Boeing 787 Dreamliner jets, as well as Airbus A350 and A380 aircraft, have Wi-Fi passenger networks that use the same network as the avionics systems of the planes, raising the possibility that a hacker could hijack the navigation system or **commandeer the plane** *through the in-plane network*, according to the US Government Accountability Office, which released a report about the planes today.

https://www.wired.com/2015/04/hackers-commandeer-new-planes-passenger-wi-fi/


External Incident
Attacks
Environment
Disaster
Human Error
Power Failure
etc.
Constituent System A
Constituent System B
Constituent System C
Constituent System D
Cooperation
Internal Incident
Emergent Behaviour
Partner System
Hardware Failure
Software Error
etc.
DAMAGES
System-of-Systems

Dependable (resilient) Architecture

Dependability Principles & Patterns

Risk Management in Development & Operation

Monitoring & Intervention

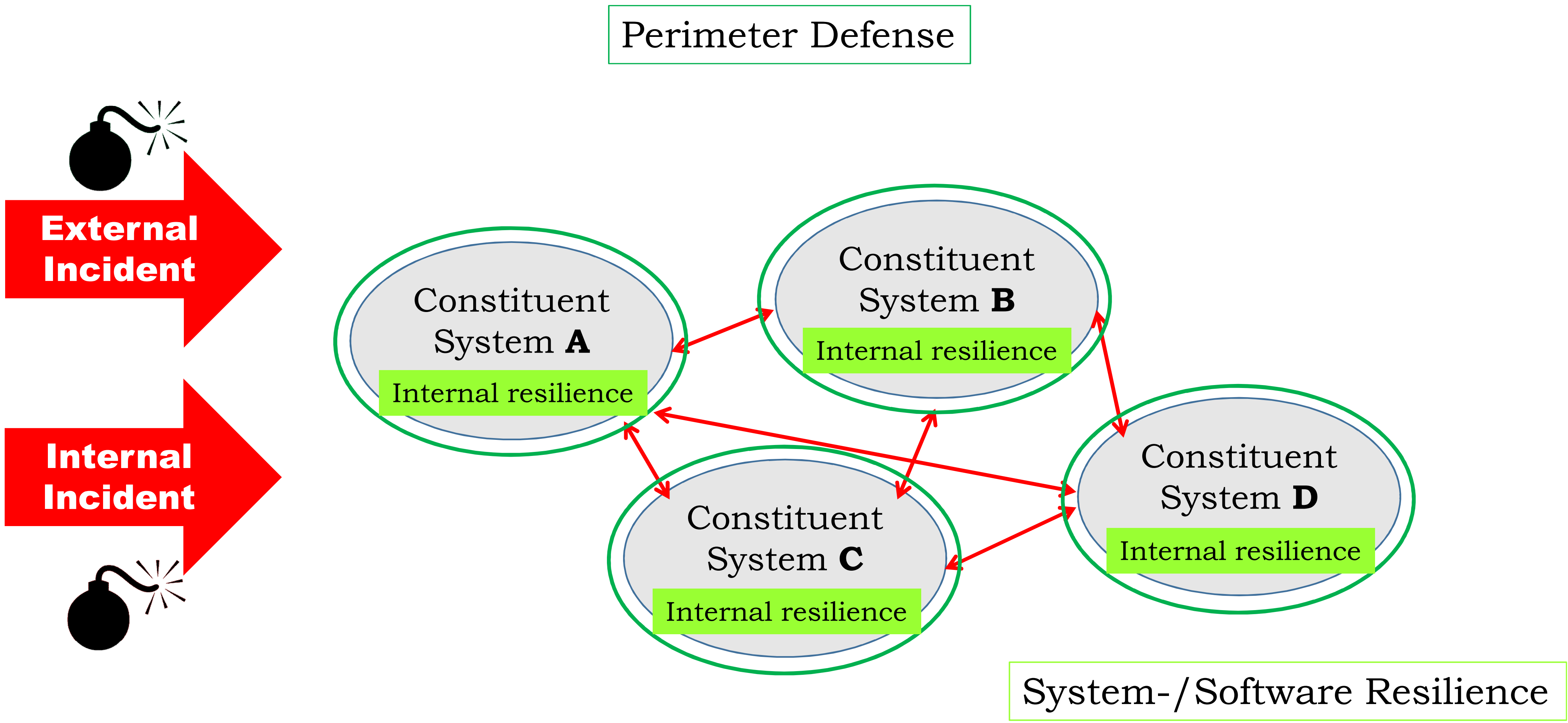
Weight depending on application area:

http://i.dailymail.co.uk

Dependability =

- Safety
- Security
- Integrity
- Availability
- Confidentiality
- Trustworthiness
- etc.

Architecture for Cyber-Physical Systems of Systems (CPSoS)



Risk Management

a) Classical

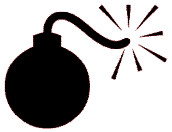
Principles & Patterns

Threat & Incident **Analysis** → Risk & Damage **Assessment** → Countermeasure & Protection Mechanisms **Definition** → Countermeasure & Protection Mechanisms **Implementation**

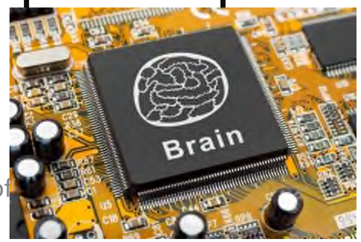
b) Autonomous Defense

Principles & Patterns

Incident → **M**onitor → **A**nalyze → **P**lan → **E**xecute



Knowledge



CPSoS Architecture Framework

Canterbury Cathedral

https://upload.wikimedia.org

Architecture guarantees:

Functionality, structural stability, and beauty

Quality Properties

Architecture for Cyber-Physical Systems of Systems (CPSoS)

Cyber-Physical Systems-of-Systems

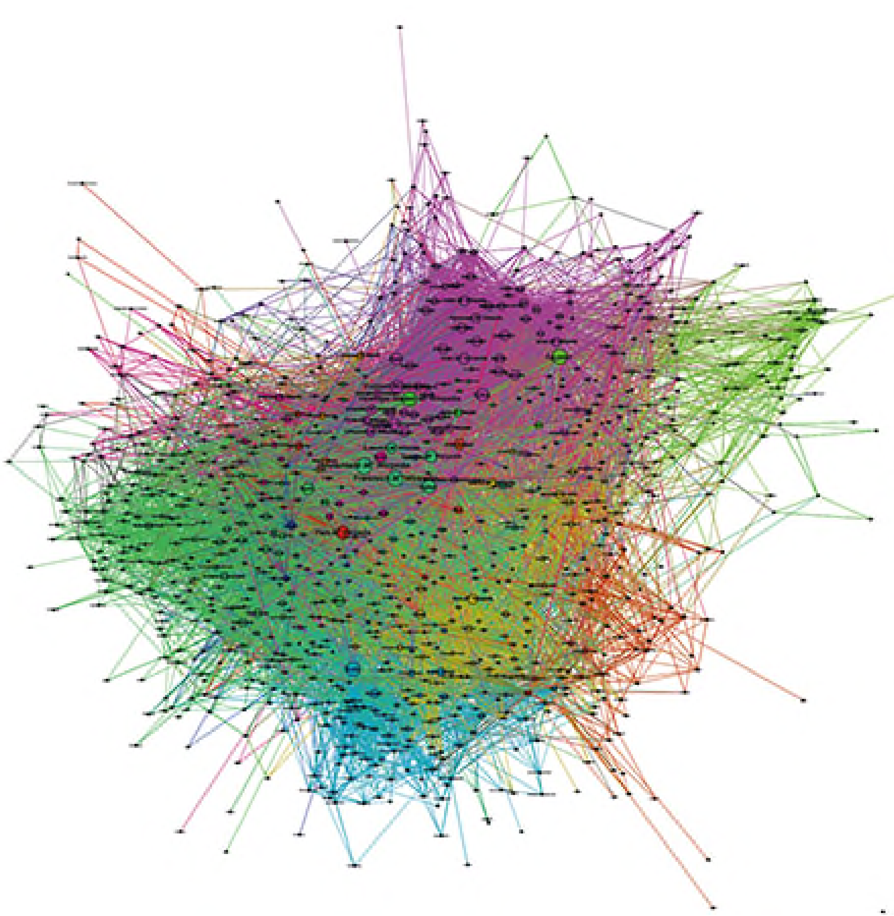
«The system does what it should, and does NOT what it should NOT do»

Architecture is the ***foundation*** for:

Functionality, dependability, and other quality properties

What is the key challenge with CPSoS ?

COMPLEXITY !



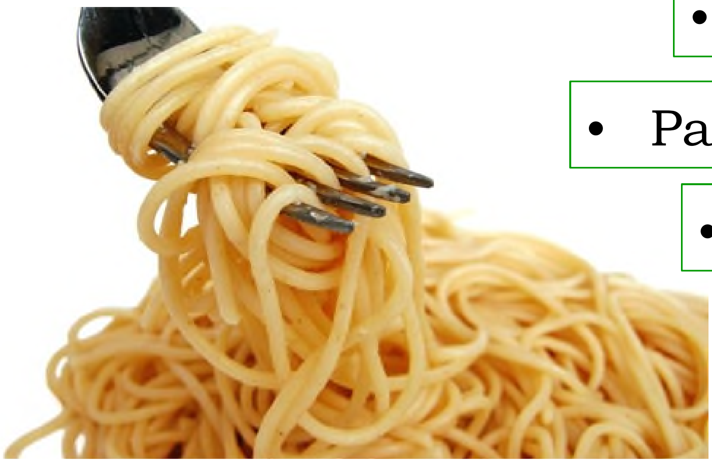
http://www.atelier-schubert.com

- Large number of parts
- Many, rich interactions
- Complicated functionality
- Emergent behaviour
- High dynamicity
- Fragmented governance
- Uncoordinated evolution
- etc.

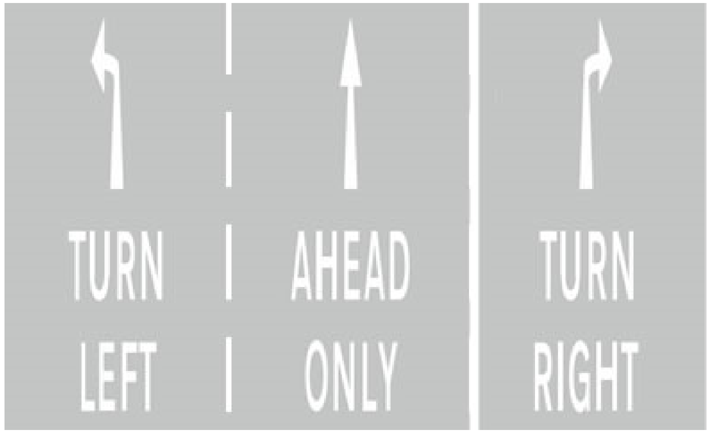
What is the key challenge with CPSoS's ?

Main CPSoS architecture tasks:

- Separation of concerns
- Partitioning and encapsulation
- Interface management



http://www.proun-game.com

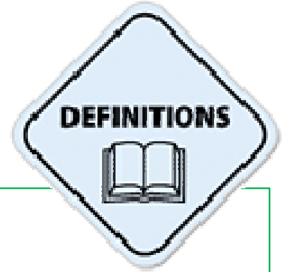


https://www.drivingtesttips.biz

http://www.chambers.com.au

Definition: **Architecture Framework**

Long-lived, industrially or commercially relevant IT-system

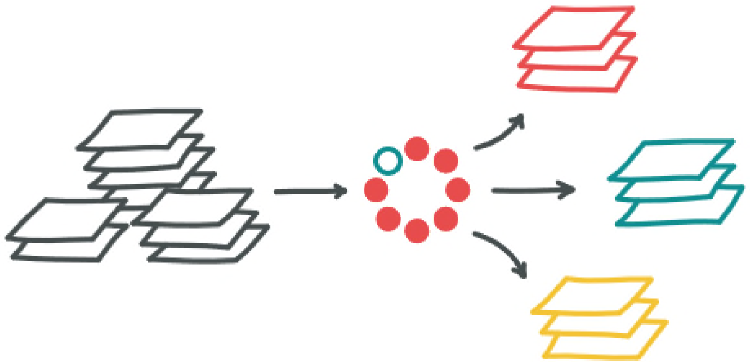


Architecture Framework =

A conceptual framework for structuring and separating the functionality and the quality properties of IT-systems to enable partitioning, verification, and life-cycle management.

Objective:

Separate and partition the dimensions of an IT-system in order to organize and manage both complexity and the stakeholders



https://miuros.com

Definition: **Architecture Framework**

Functional Architecture Decomposition

Functionality

Data

Models, etc.

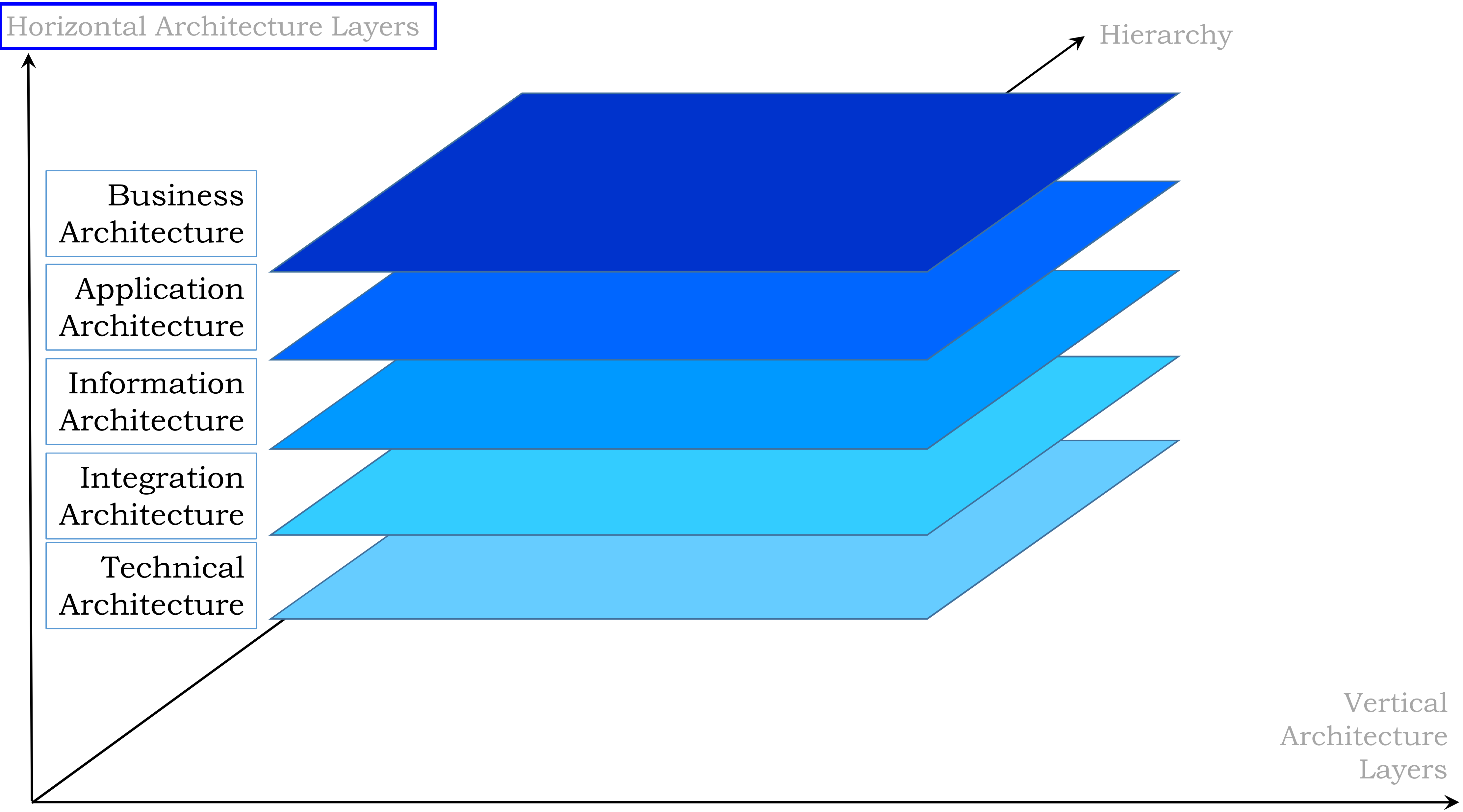
Principles Patterns

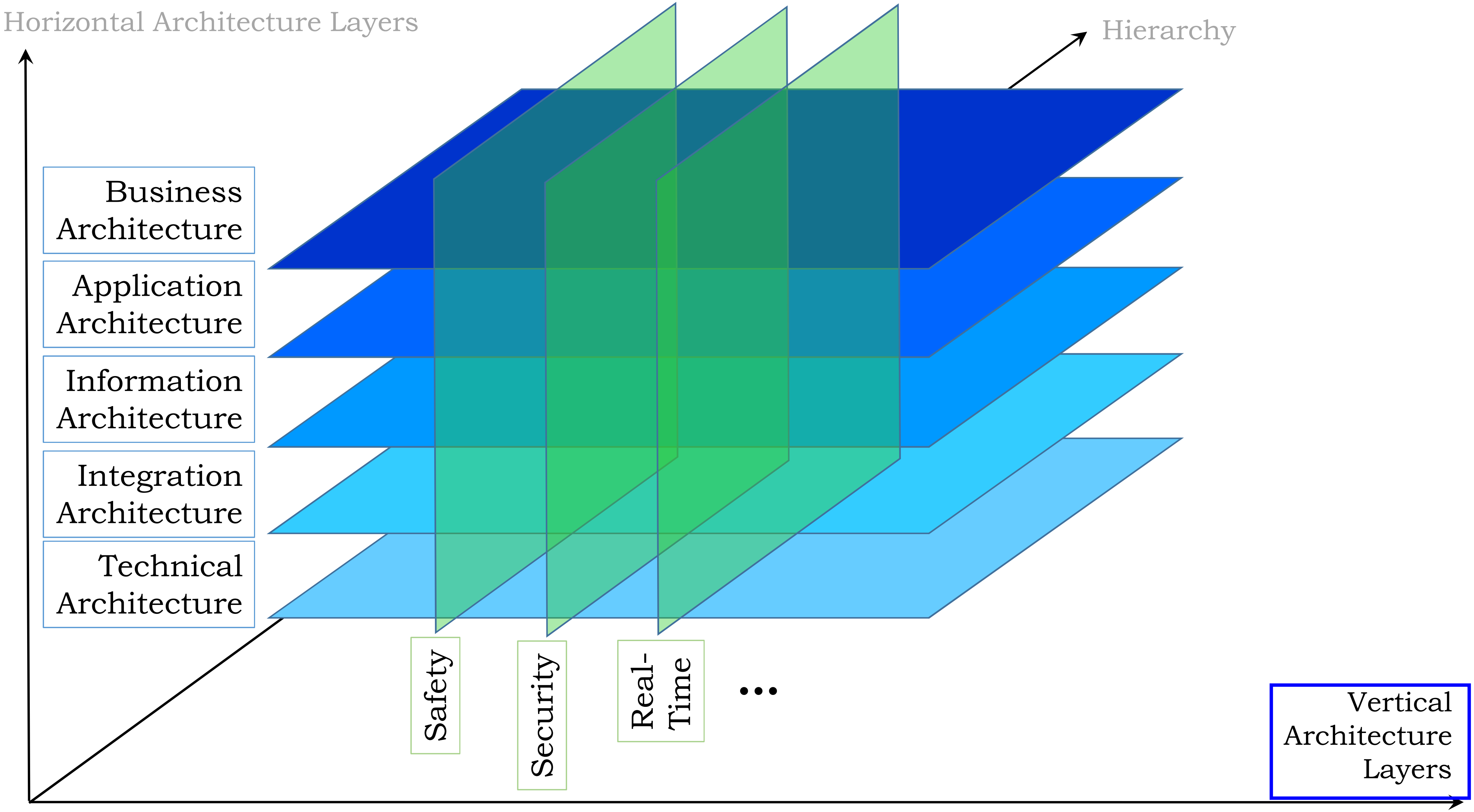
Hierarchical Decomposition

Quality Properties Decomposition

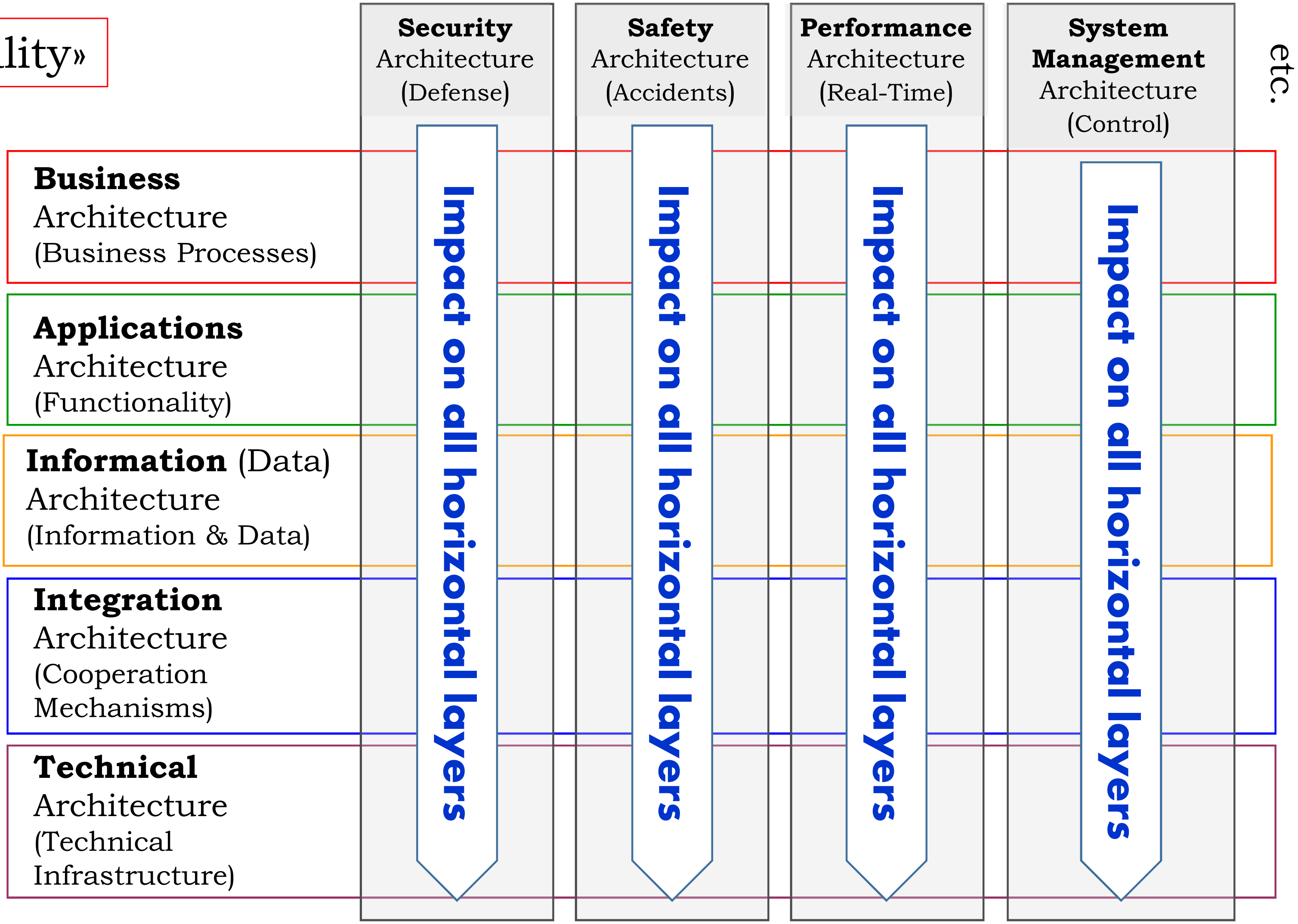
- **Separation of Concerns**
- **Order: Partitioning & Encapsulation**
- **Interface boundaries**

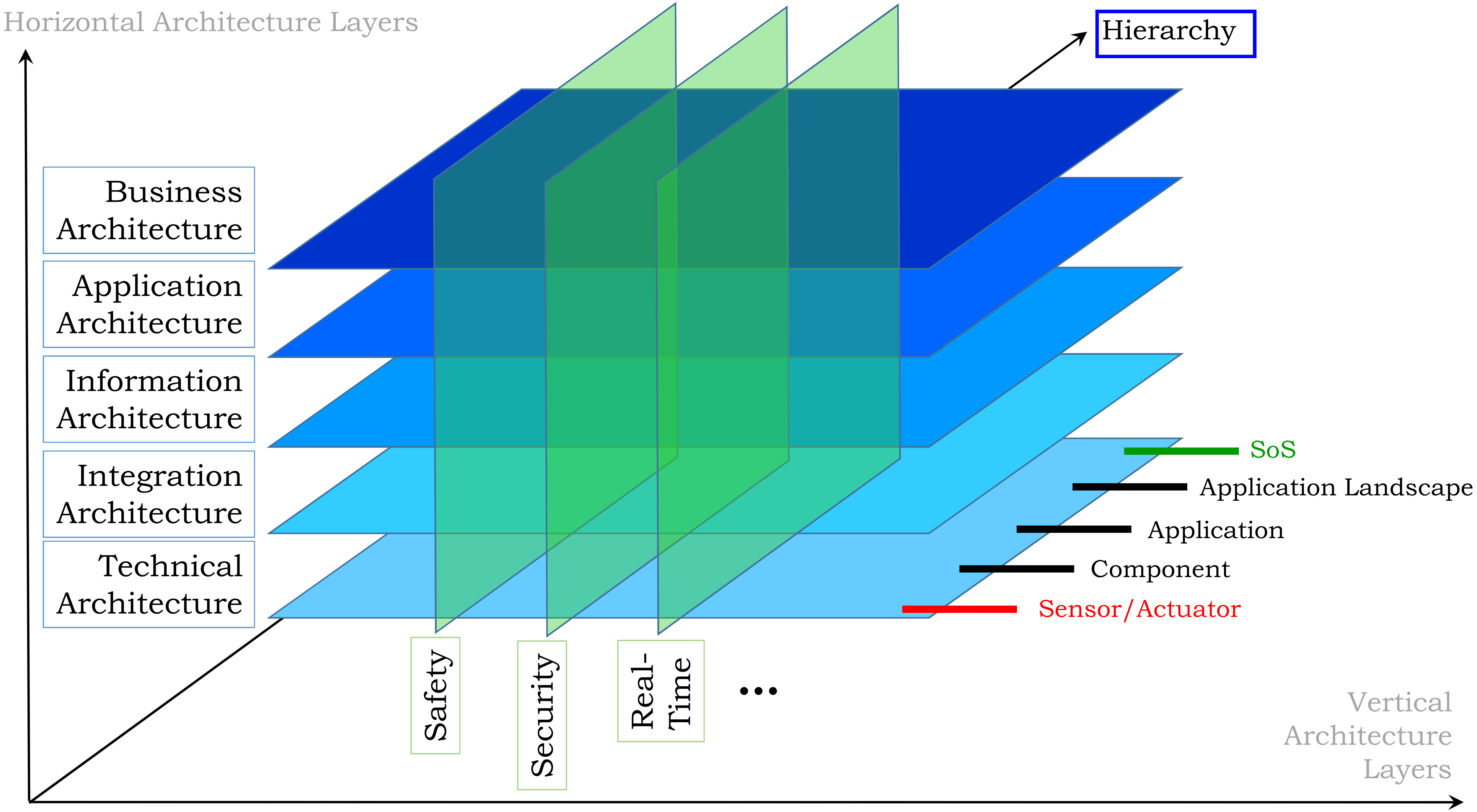
https://i.pinimg.com

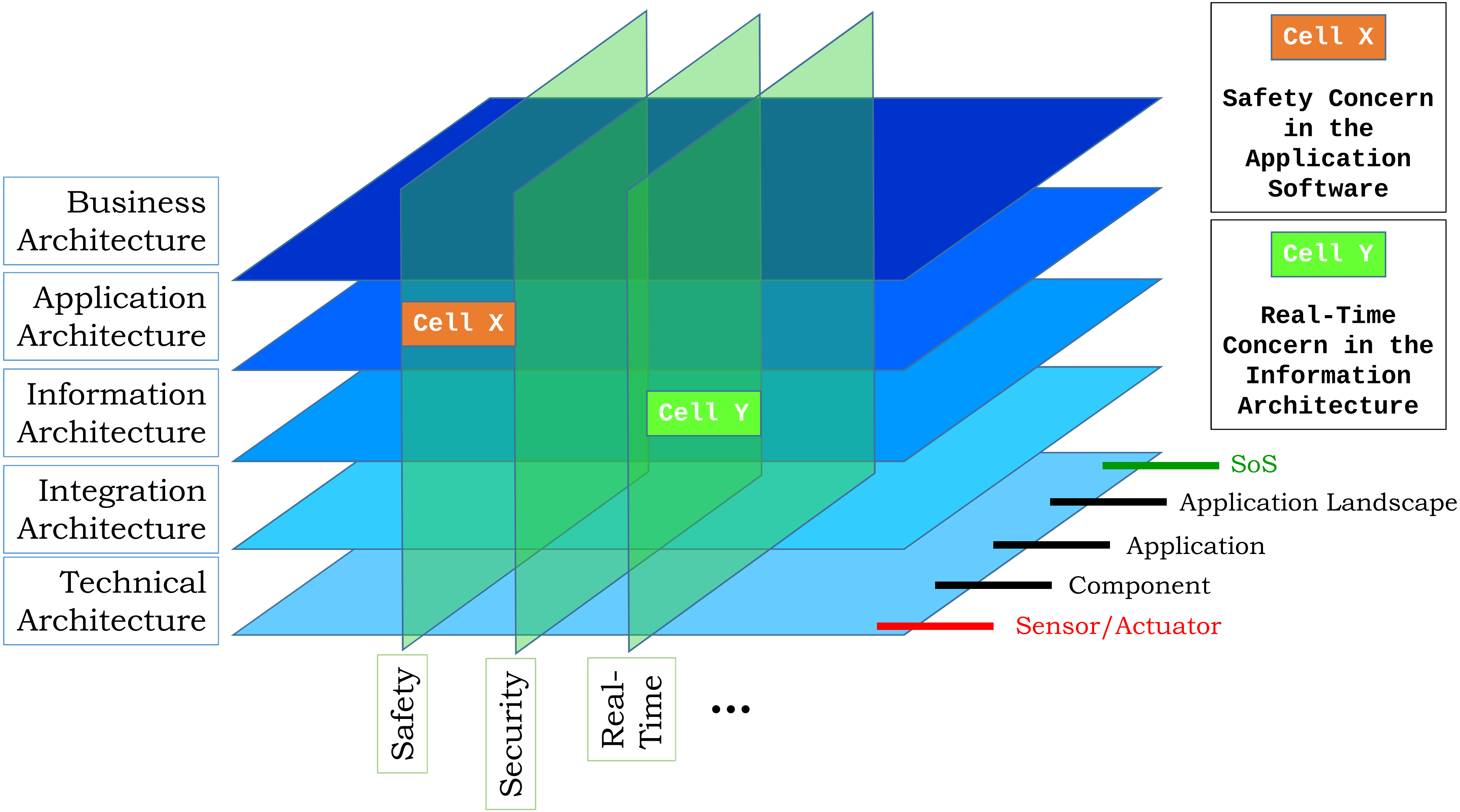

Horizontal Architecture Layers
Hierarchy
Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
Vertical Architecture Layers


Horizontal Architecture Layers
Hierarchy
Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
Safety
Security
Real-Time
...
Vertical Architecture Layers

«Canon of Orthogonality»




Horizontal Architecture Layers
Hierarchy
Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
SoS
Application Landscape
Application
Component
Sensor/Actuator
Safety
Security
Real-Time
...
Vertical Architecture Layers



Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
Cell X
Cell Y
Safety
Security
Real-Time
...
Cell X
Safety Concern in the Application Software
Cell Y
Real-Time Concern in the Information Architecture
SoS
Application Landscape
Application
Component
Sensor/Actuator

Cell X = **Safety Concern in the Application Software**

Architecture Framework Cells =

Allow assignment, structuring, and separating of the functionality and of the quality properties of IT-systems to enable partitioning and life-cycle management.

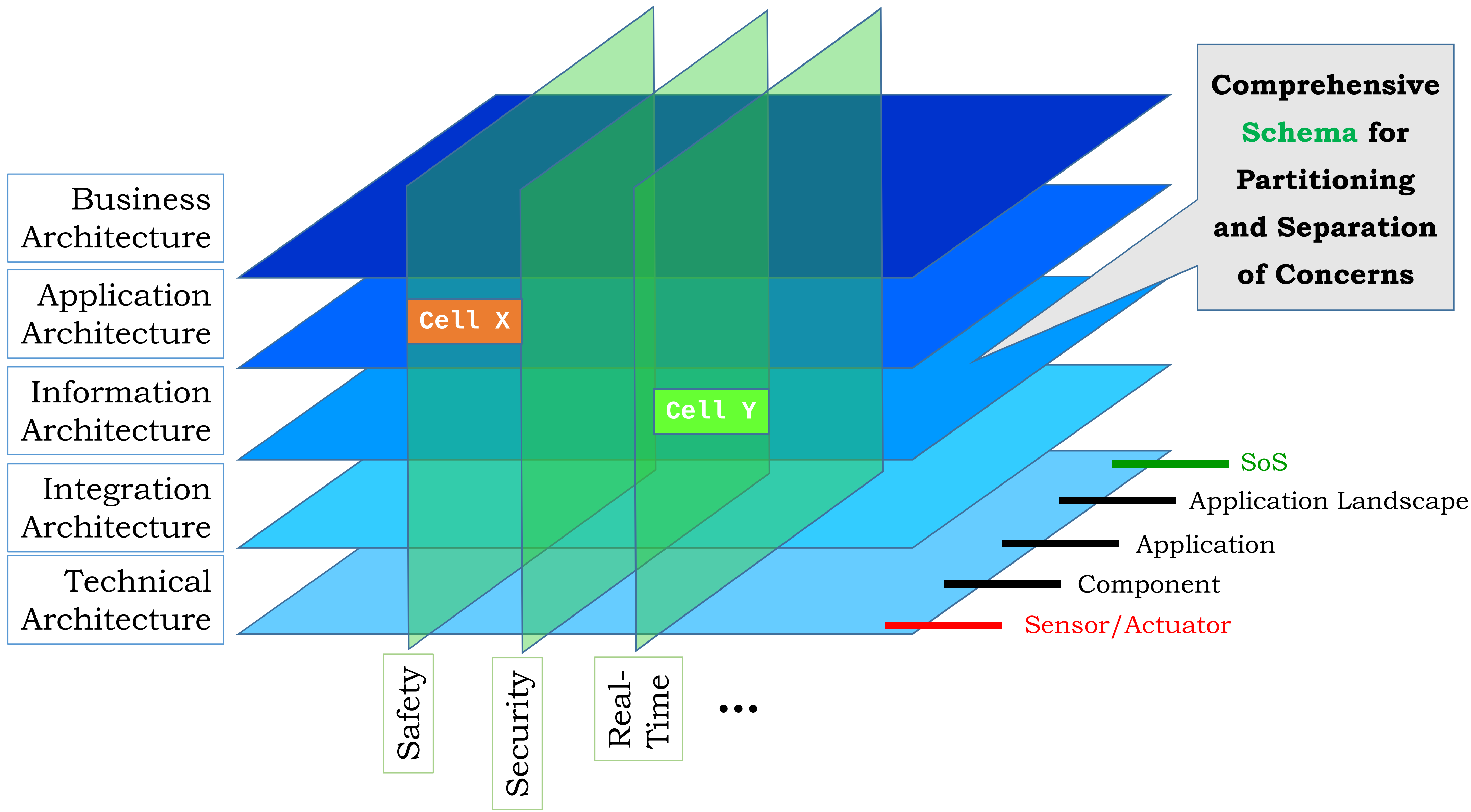
⇒ **Formulation of Powerful Set of Architecture Principles**, e.g.:

NEVER implement security functionality in the applications software

… but only allow calls to the security functionality

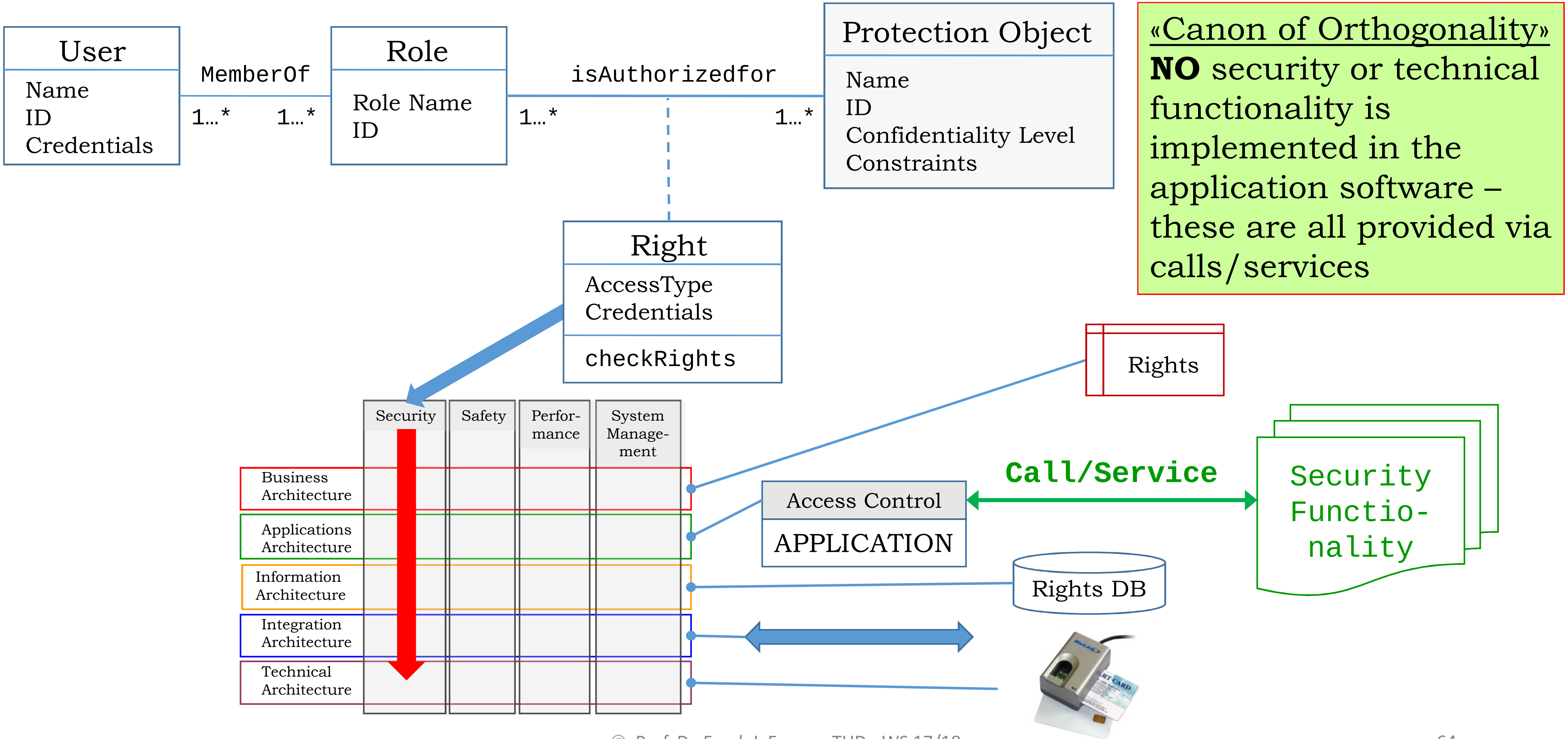
«Canon of Orthogonality»

https://www.npmis.com


Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
Cell X
Cell Y
Comprehensive Schema for Partitioning and Separation of Concerns
SoS
Application Landscape
Application
Component
Sensor/Actuator
Safety
Security
Real-Time
...

Example: Access Control (Security Architecture)

<u>«Canon of Orthogonality»</u>
NO security or technical functionality is implemented in the application software – these are all provided via calls/services



CPSoS Risk Example 6: Airplane Hacking

https://www.ianvisits.co.uk

Highly dangerous **mix** of concerns:

- Passenger WI-FI NW
- Avionics control NW

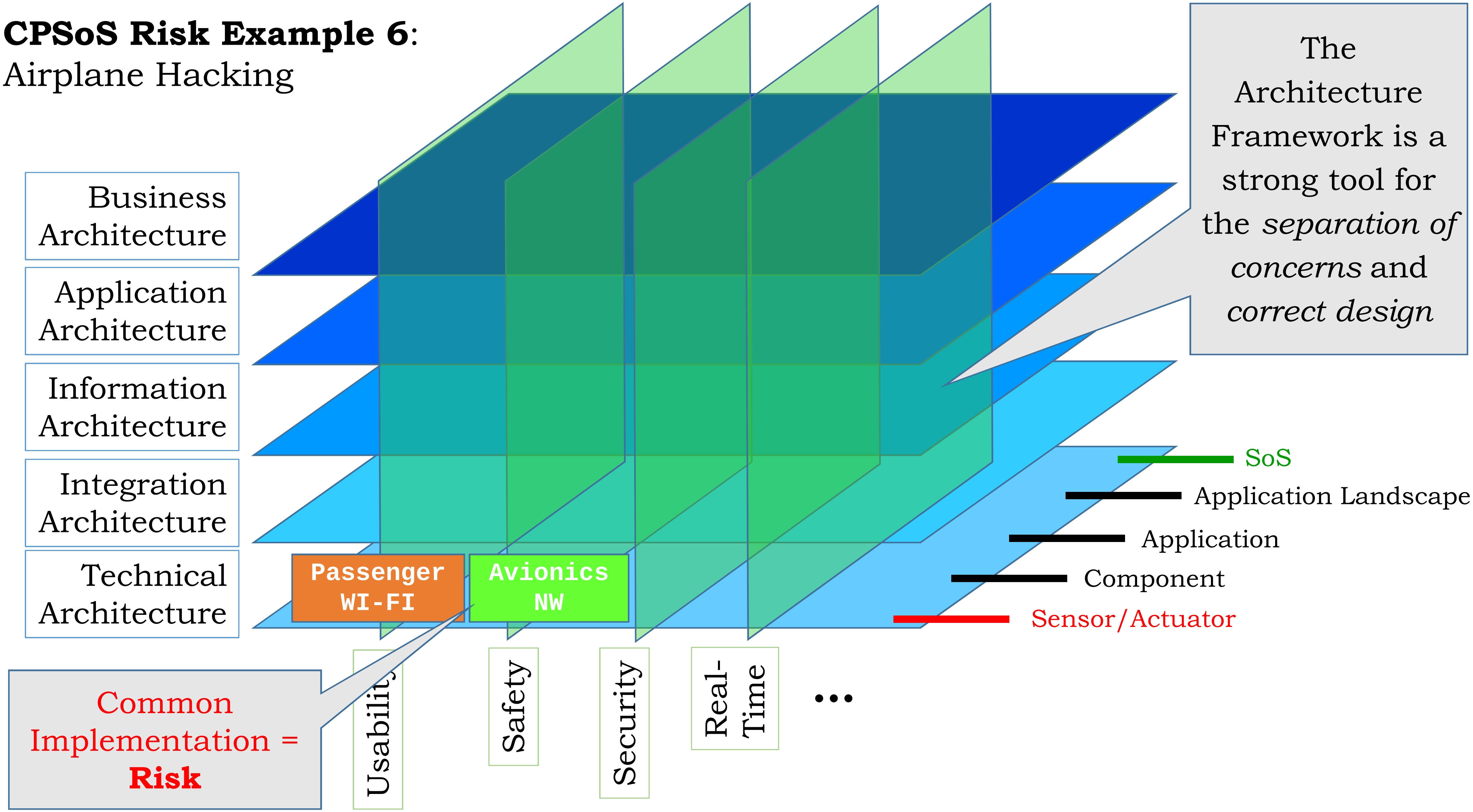
on the same technical infrastructure

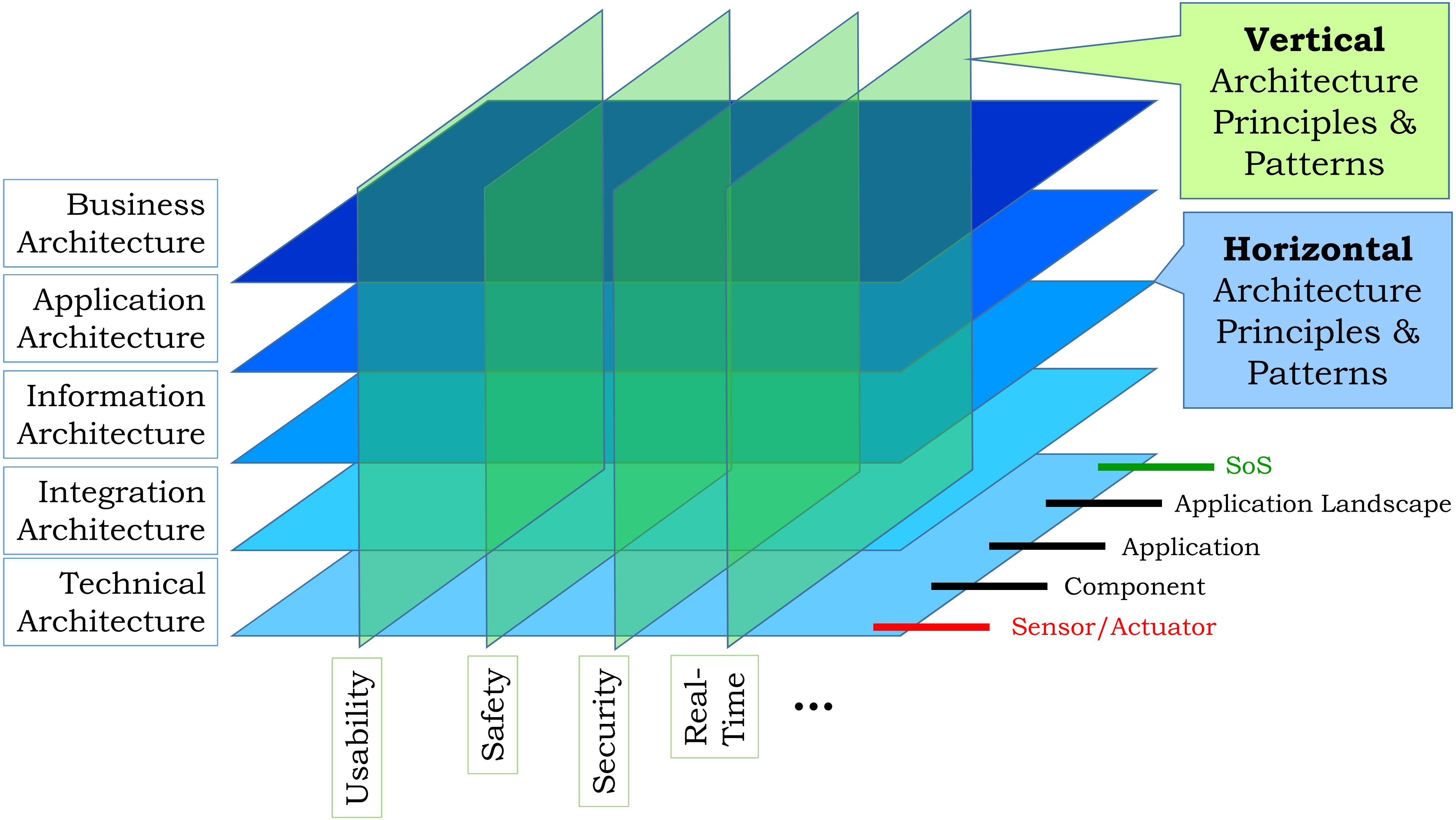
15.4.2015:

Boeing 787 Dreamliner jets, as well as Airbus A350 and A380 aircraft, have Wi-Fi ***passenger networks that use the same network as the avionics systems of the planes***, raising the possibility that a hacker could hijack the navigation system or commandeer the plane through the in-plane network, according to the US Government Accountability Office, which released a report about the planes today.

https://www.wired.com/2015/04/hackers-commandeer-new-planes-passenger-wi-fi/

CPSoS Risk Example 6:
Airplane Hacking




Business Architecture
Application Architecture
Information Architecture
Integration Architecture
Technical Architecture
Vertical Architecture Principles & Patterns
Horizontal Architecture Principles & Patterns
SoS
Application Landscape
Application
Component
Sensor/Actuator
Usability
Safety
Security
Real-Time
...

Architecture Principles and Patterns

Architecture Principles:

Fundamental insights – formulated as *enforcable rules* – how a good software-system should be built [⇐ «Eternal Truths»]

Architecture Patterns:

Proven, *generic solutions* to clearly specified architectural problems which can be adapted to the task at hand

Example: *Horizontal* **Architecture Principle** «Architecture Layer Isolation» (1/3)

Business Architecture **Layer**
(Business Processes)

Isolation

Applications Architecture **Layer**
(Functionality)

Isolation

Information (Data) Architecture **Layer**
(Information & Data)

Isolation

Integration Architecture **Layer**
(Cooperation Mechanisms)

Isolation

Technical Architecture **Layer**
(Technical Infrastructure)

A1

[1]
Always use standardized, technology-independent, and product-independent mechanisms for transfer of data and control between layers

Example: *Horizontal* **Architecture Principle** «Architecture Layer Isolation» (2/3)

Business Architecture **Layer**
(Business Processes)

Isolation

Applications Architecture **Layer**
(Functionality)

Isolation

Information (Data) Architecture **Layer**
(Information & Data)

Isolation

Integration Architecture **Layer**
(Cooperation Mechanisms)

Isolation

Technical Architecture **Layer**
(Technical Infrastructure)

Breaking Layers

Direct access – **bypassing** the standardized, technology-independent mechanisms

Result:

- Technology dependence
- Vendor lock-in
- No standards-compliance

http://media.istockphoto.com

Example: *Horizontal* **Architecture Principle** «Architecture Layer Isolation» (3/3)

Business Architecture **Layer**
(Business Processes)

Isolation

Applications Architecture **Layer**
(Functionality)

Isolation

Information (Data) Architecture **Layer**
(Information & Data)

Isolation

Integration Architecture **Layer**
(Cooperation Mechanisms)

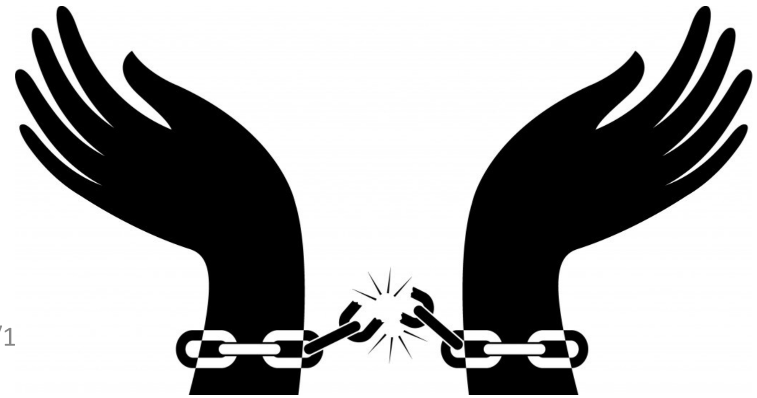
Isolation

Technical Architecture **Layer**
(Technical Infrastructure)

NOT Breaking Layers

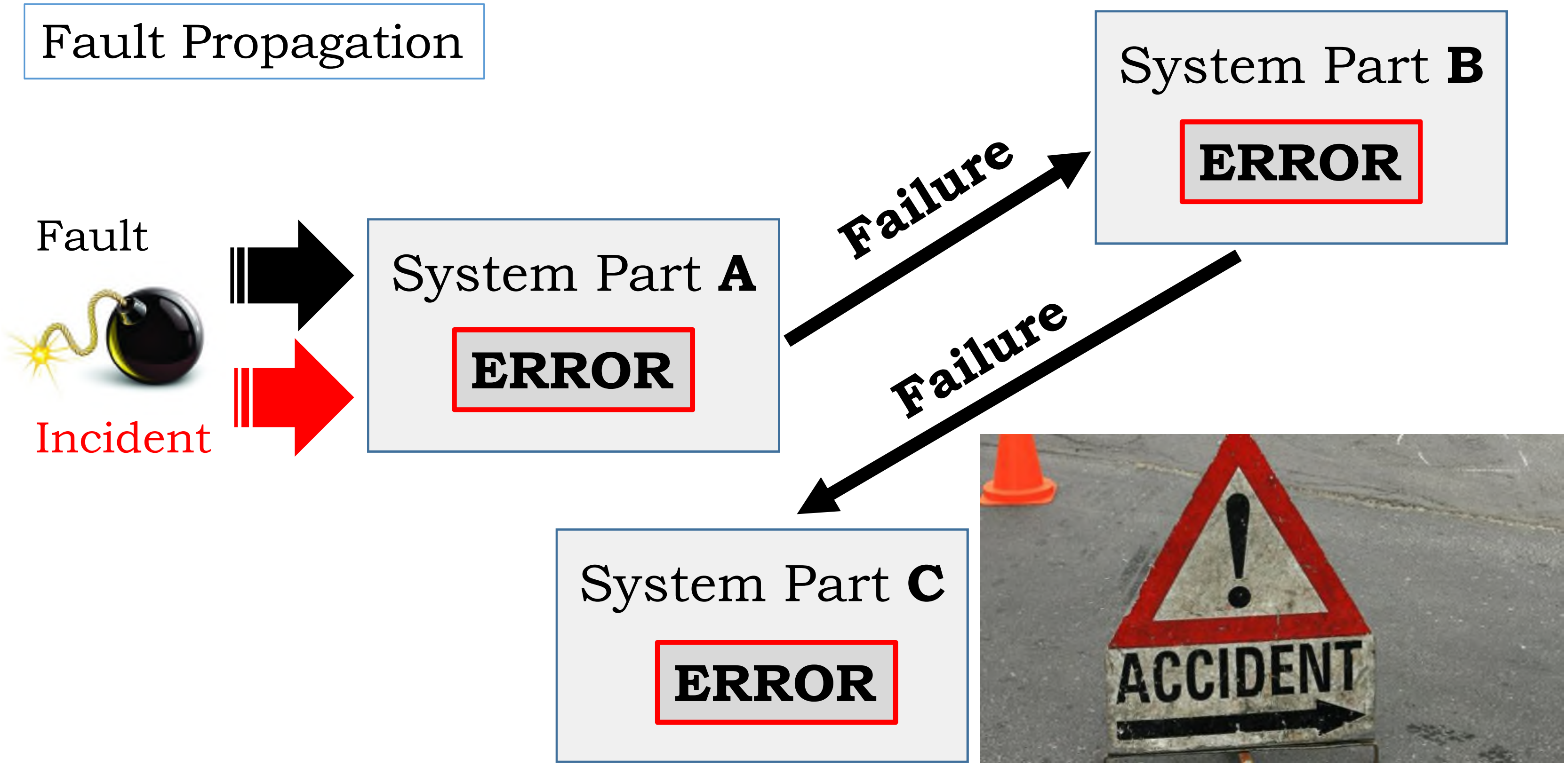
Industry-standard, technology-independent mechanism

Result:
- Technology **in**dependence
- Vendor **in**depence
- Full standards-compliance



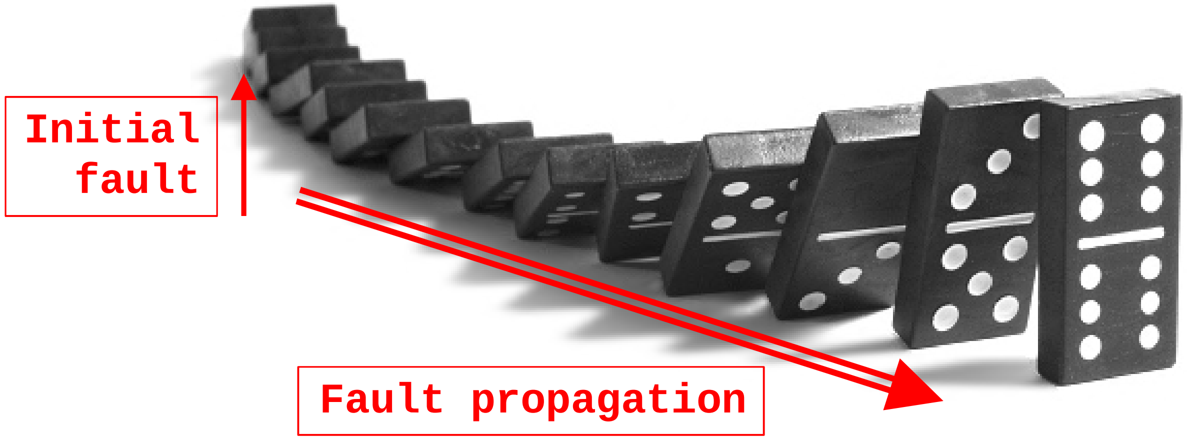
https://i1.wp.com

Example: *Vertical* **Architecture Principle** «Fault Containment» (1/3)



The consequences of a *fault* – the ensuing *error* – can **propagate** either by an erroneous message or by an erroneous output action of the faulty part

Example: *Vertical* **Architecture Principle** «Fault Containment» (2/3)

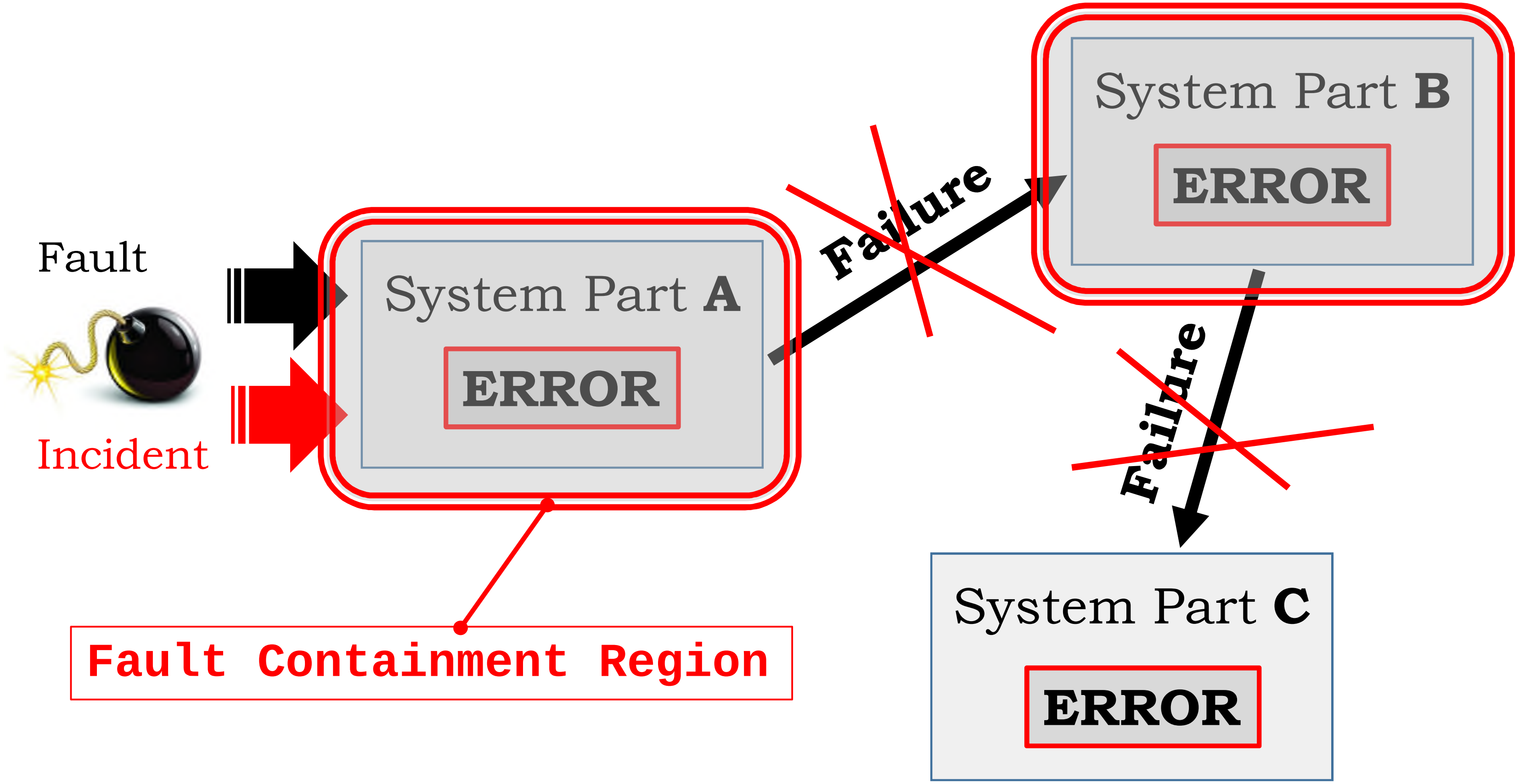


Fault propagation can consecutively affect system parts (⇒ Domino effect)

The result may be severe malfunctions or the loss of the system

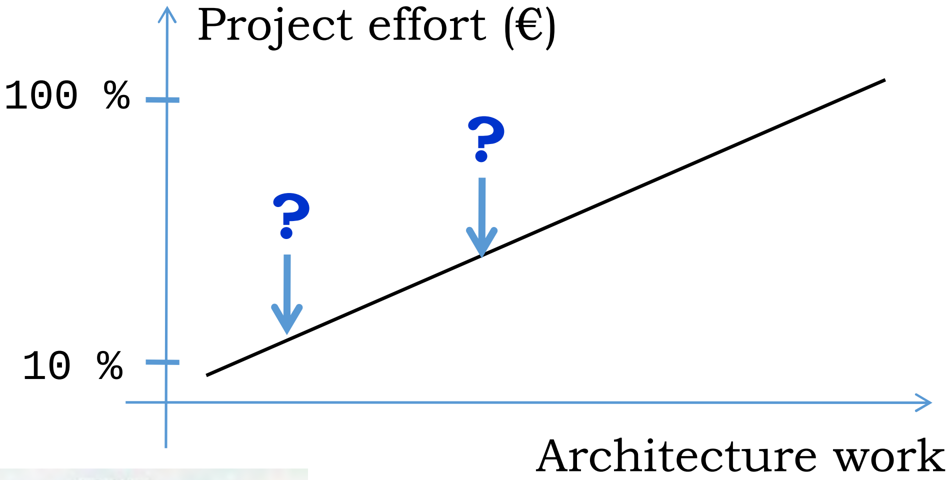
Fault propagation is difficult to predict

Example: *Vertical* **Architecture Principle** «Fault Containment» (3/3)

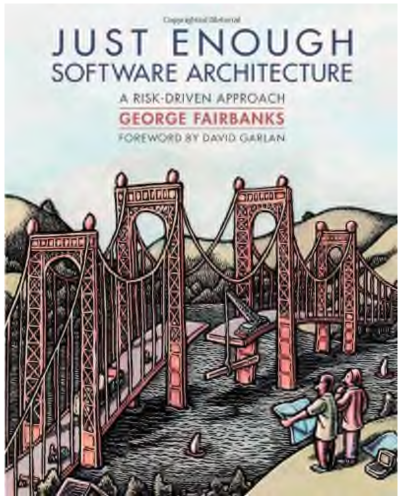


Build **error** propagation boundaries around each system part

How much Architecture is enough ?



http://i.imgur.com



G. Fairbanks / ISBN 978-0-9846181-0-1

<u>Answer</u>:

- System creation/extensions with **high risk** need much architecture work
- System creation/extensions with **low risk** need little architecture work

(George Fairbanks - ISBN 978-0-9846181-0-1, 2010)

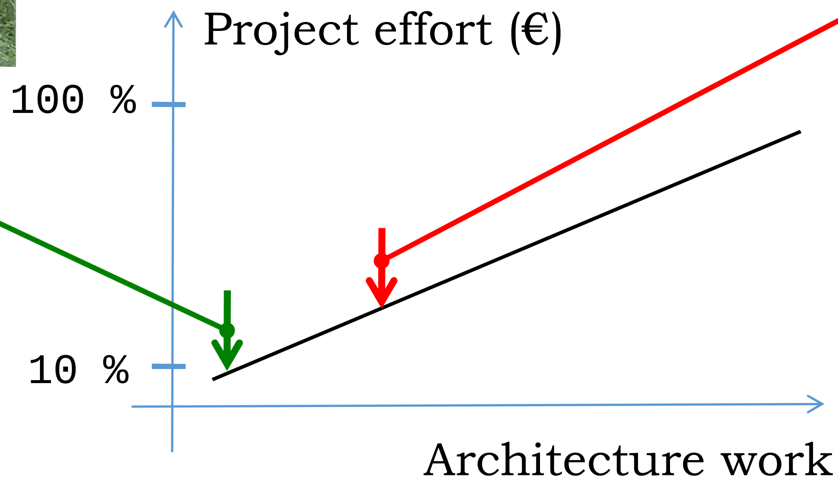
How much Architecture is enough ?

Low Risk

http://www.dimensionsinfo.com/dimensions-of-a-dog-house

High Risk

http://www.skyscrapernews.org



Outlook

https://reisekino.files.wordpress.com

The Message

Cyber-Physical Systems-of-Systems (CPSoS) offer great **opportunities**

Many of today's interesting systems are CPSoS

However, CPSoS also generate significant **risks**

We have seen serious examples of accidents

Cyber-Physical Systems-of-Systems (CPSoS) must be built and operated with **high dependability**

An adequate **architecture** is the *foundation* of correct functionality, high dependability, and other quality properties

Building dependable CPSoS:
- *Separation of concerns*
- *Partitioning and encapsulation*
- *Architecture principles & patterns for dependability*

Operating dependable CPSoS:
- *Monitoring*
- *Run-time dependability mechanisms*
- *Continuous risk assessment & management*

http://mobi.vub.ac.be

Cyber-Physical Systems-of-Systems are an *important part of our future*

… but we need to build and operate them in a **dependable** way

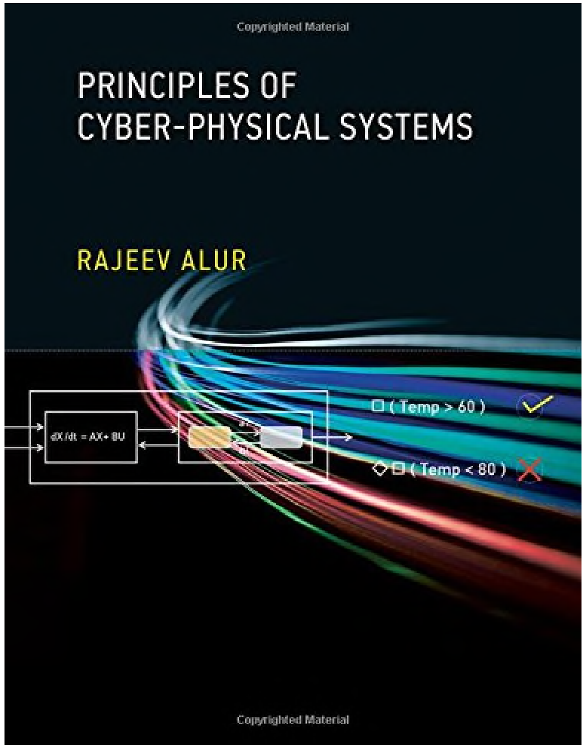
If we continue to develop our technology without wisdom or prudence, our servant may prove to be our executioner

Omar N. Bradley (U.S. Army General, Chairman of the Joint Chiefs of Staff [1949])

http://www.1zoom.me/en/wallpaper

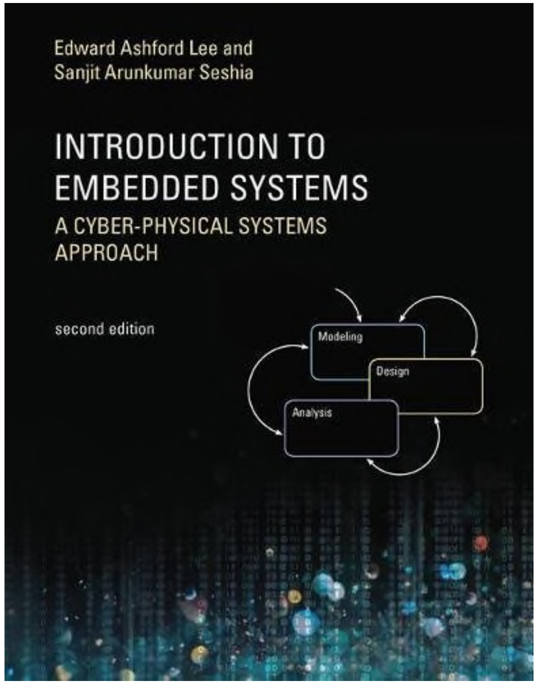
Recommended Reading

Recommended Reading: **CPS**



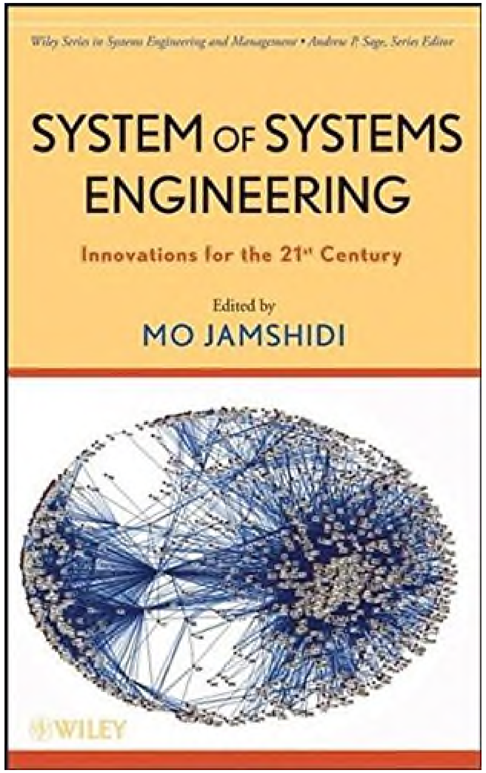
Rajeev Alur:
Principles of Cyber-Physical Systems
MIT Press, Cambridge, MA, USA, 2015. ISBN 978-0-262-02911-7

Recommended Reading: **CPS**



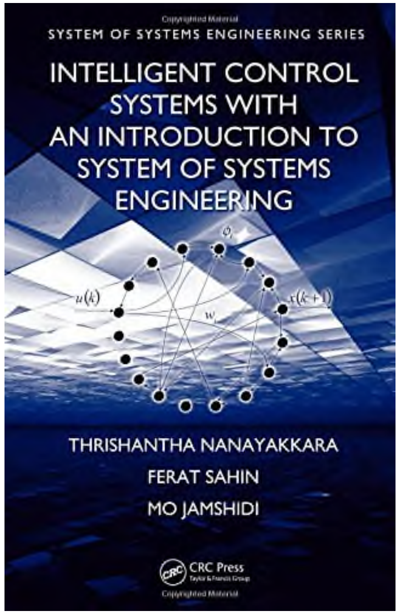
Edward A. Lee, Sanjit A. Seshia:
Introduction to Embedded Systems: *A Cyber-Physical Systems Approach*
MIT Press, Cambridge, MA, USA, 2nd edition 2017. ISBN 978-0-262-53381-2

Recommended Reading: **SoS**



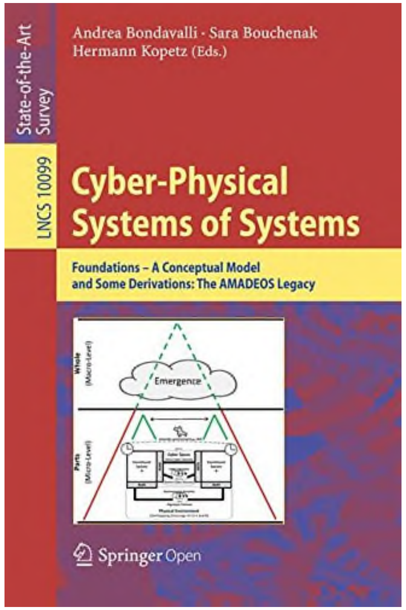
Mohammad Jamshidi:
System of Systems Engineering
Wiley Inc., USA, 2008. ISBN 978-0-470-19590-1

Recommended Reading: **SoS**



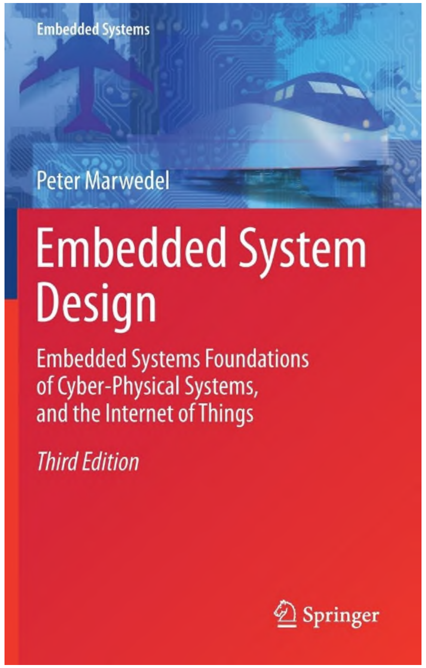
Thrishantha Nanayakkara, Ferat Sahin, Mo Jamshidi:
Intelligent Control Systems with an Introduction to Systems of Systems Engineering
CRC Press (Taylor & Francis), USA, 2009.
ISBN 978-1-4200-7924-1

Recommended Reading: **CPSoS**



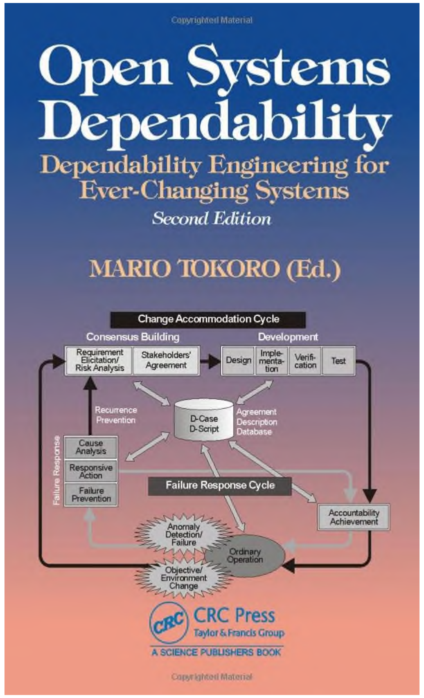
Andrea Bondavalli, Sara Bouchenak, Hermann Kopetz (Editors):
Cyber-Physical Systems of Systems: *Foundations - A Conceptual Model and some Derivations*
Springer-Verlag, Germany, 2016. ISBN 978-3-319-47589-9

Recommended Reading: **CPSoS**



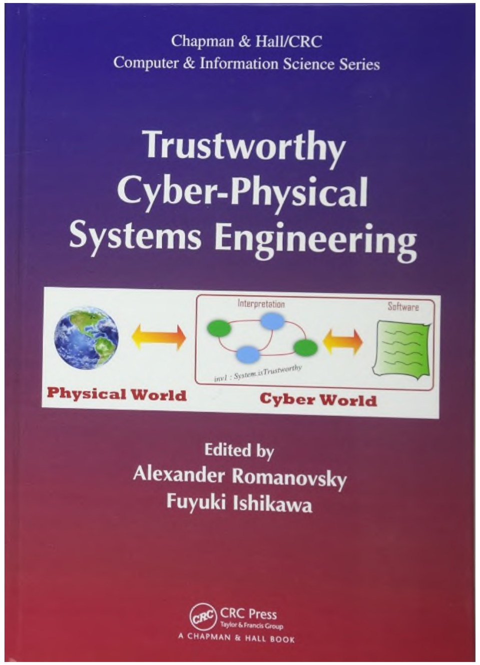
Peter Marwedel:
Embedded System Design: *Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things*
Springer-Verlag, Germany, 3^{rd} edition, 2018. ISBN 978-3-319-56043-4

Recommended Reading: **Dependable SW**



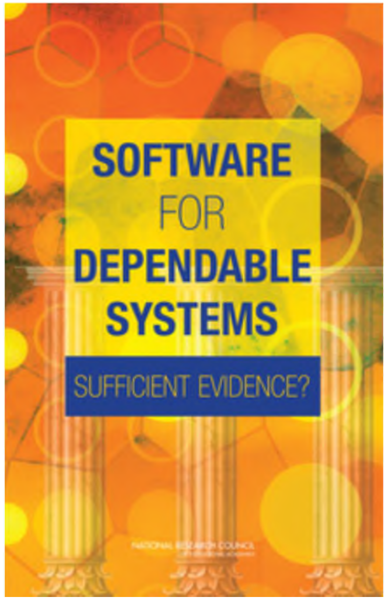
Mario Tokoro (Editor):
Open Systems Dependability: *Dependability Engineering for Ever-Changing Systems*
CRC Press, Taylor & Francis Inc., USA, 2^{nd} edition 2015. ISBN 978-1-4987-3628-2

Recommended Reading: **Dependable SW**



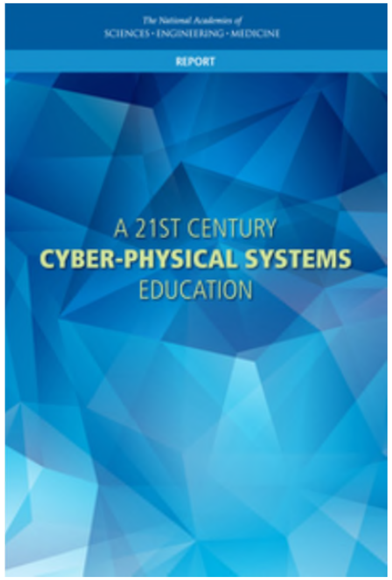
Alexander Romanovsky, Fuyuki Ishikawa (Editors):
Trustworthy Cyber-Physical Systems Engineering
CRC Press, Taylor & Francis Inc., USA, 2016. ISBN 978-1-4987-4245-0

Recommended Reading: **Dependable SW**



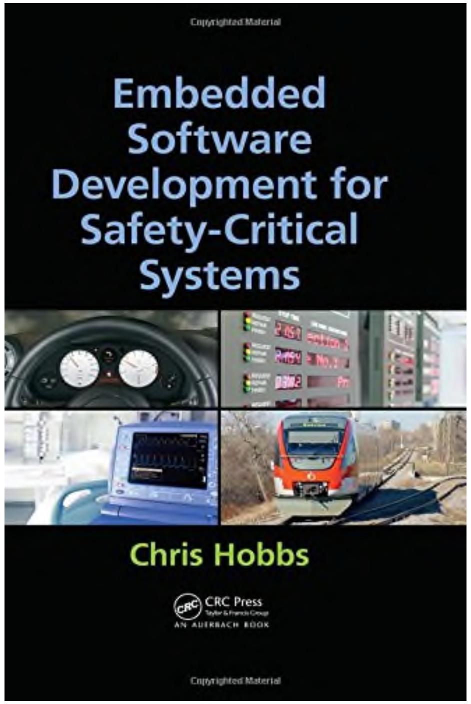
Daniel Jackson, Martyn Thomas, Lynette I. Millett (Editors):
Software for Dependable Systems – *Sufficient Evidence?*
The US National Academic Press, Washington, 2007. ISBN 978-0-309-38450-6. Available at http://nap.edu/11923

Recommended Reading: **Dependable SW**



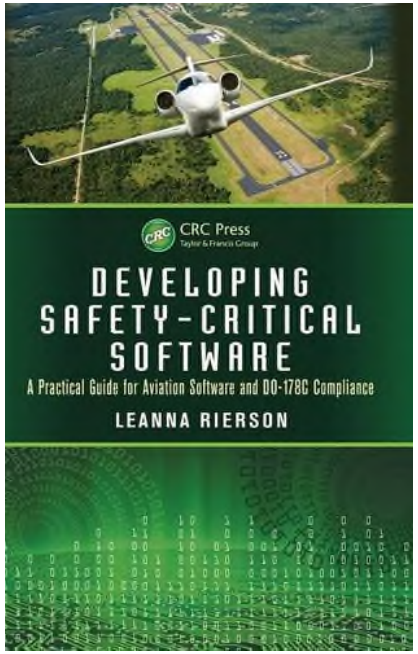
Committee on 21st Century Cyber-Physical Systems Education:
21st Century Cyber-Physical Systems Education
The US National Academic Press, Washington, 2016. ISBN 978-0-309-45163-5. Available at http://nap.edu/23686

Recommended Reading: **Dependable SW**



Chris Hobbs:
Embedded Software Development for Safety-Critical Systems
Taylor & Francis Inc., USA, 2015. ISBN 978-1-4987-2670-2

Recommended Reading: **Dependable SW**



Leanna Rierson:
Developing Safety-Critical Software: *A Practical Guide for Aviation Software and DO-178C Compliance*
Taylor & Francis Inc., USA, 2013. ISBN 978-1-4398-1368-3


WORKS END
THANK YOU