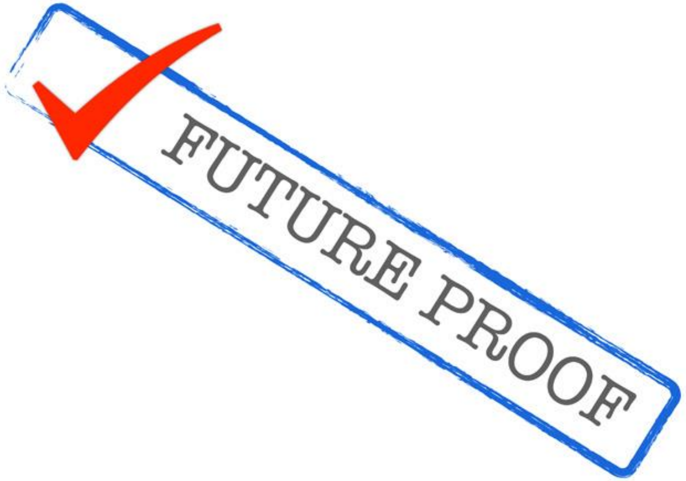


https://static1.squarespace.com

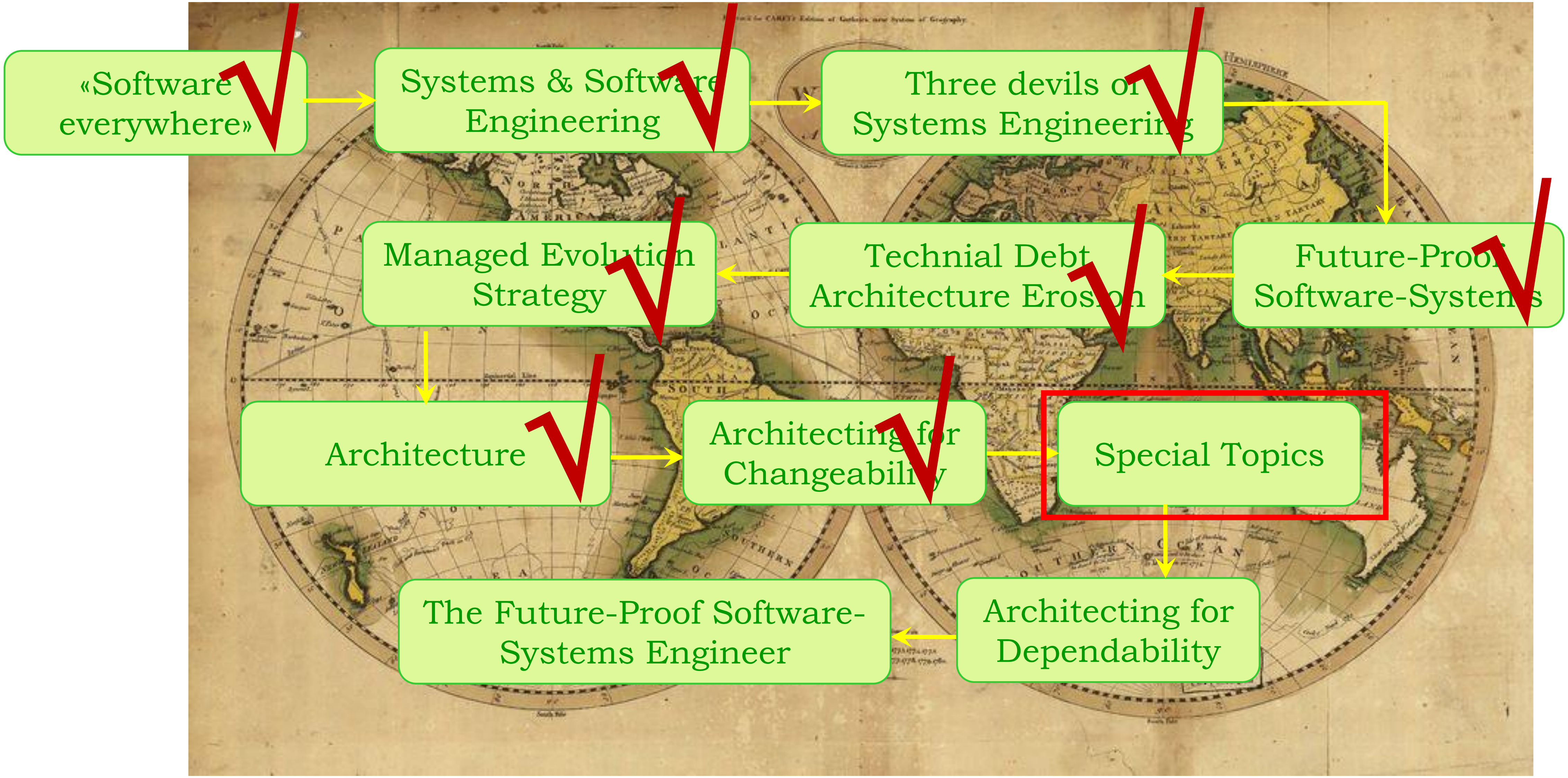
Future-Proof Software-Systems (FPSS)

Part 3C: Special Topics (1)

Lecture WS 2019/20: Prof. Dr. Frank J. Furrer

Version 0.2/08.12.2019

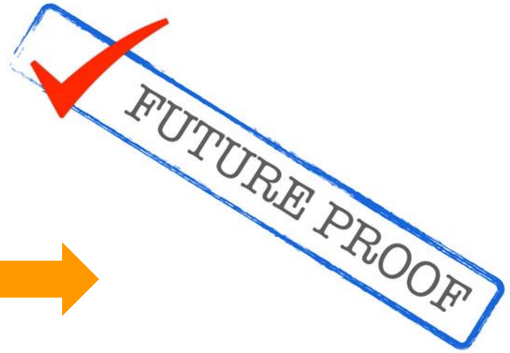
Our journey:


«Software everywhere»
Systems & Software Engineering
Three devils of Systems Engineering
Future-Proof Software-Systems
Technial Debt Architecture Erosion
Managed Evolution Strategy
Architecture
Architecting for Changeability
Special Topics
Architecting for Dependability
The Future-Proof Software-Systems Engineer

Special Topics = Specific IT-Architecture Areas related to IT-Systems

Part 3C

- Cyber-Physical Systems (CPS)
- Systems-of-Systems (SoS)
- Cyber-Physical Systems-of-Systems (CPSoS)
- Cloud Computing
- Microservices
- Agile Manifesto and Future-Proof Software-Systems ?
- Domain Software Engineering
- Legacy System Migration/Modernization
- Software Product Lines



Future-Proof Software-Systems

Cyber-Physical Systems (CPS)

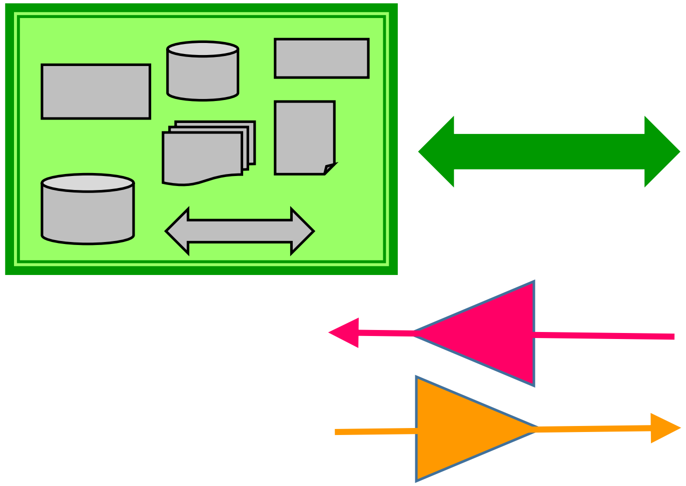
Cyber-Physical System (CPS)

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

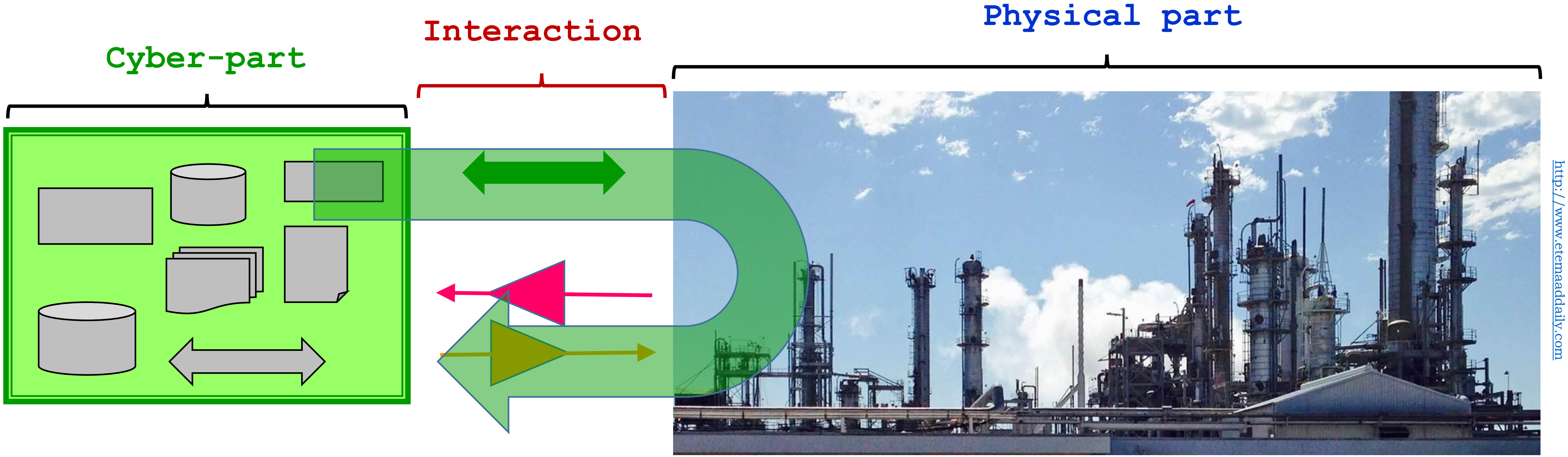



http://www.etemaaddaily.com

Sensors: Read plant information

Actuators: Control plant

Cyber-Physical System

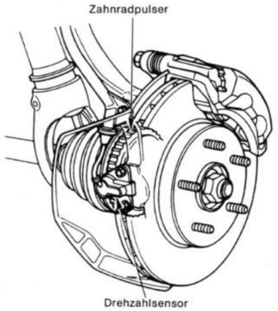


Software Control Loop

Sensors: Read plant information

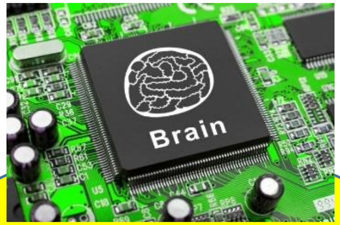
Actuators: Control plant

CPS-Example: **ESC**



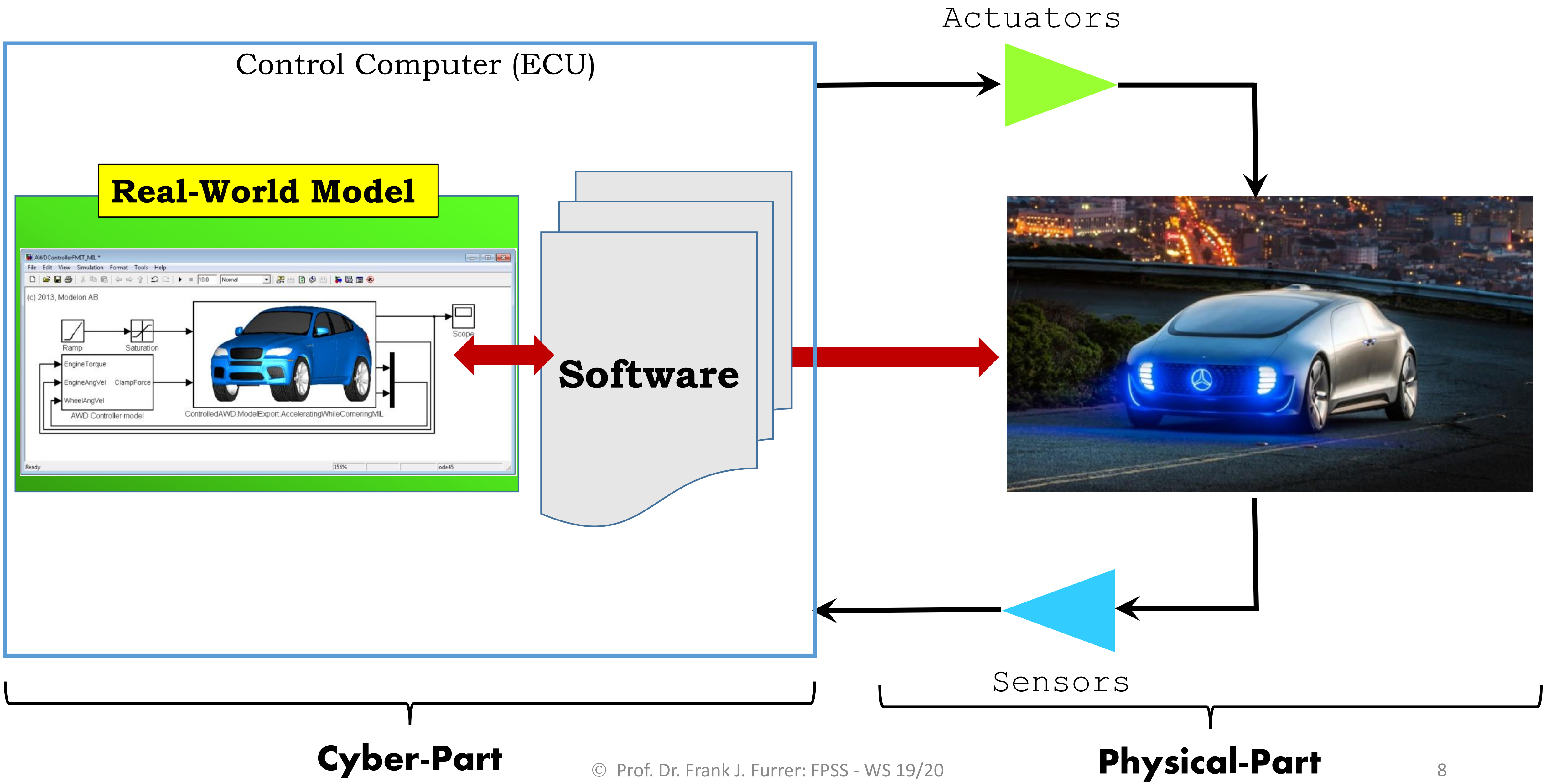
http://www.polizeiticker.ch

ECU (SW)



Kowledge Base

Feedback Loop



http://www.modelon.com

http://cdn1.alphr.com

https://www.siemens.com

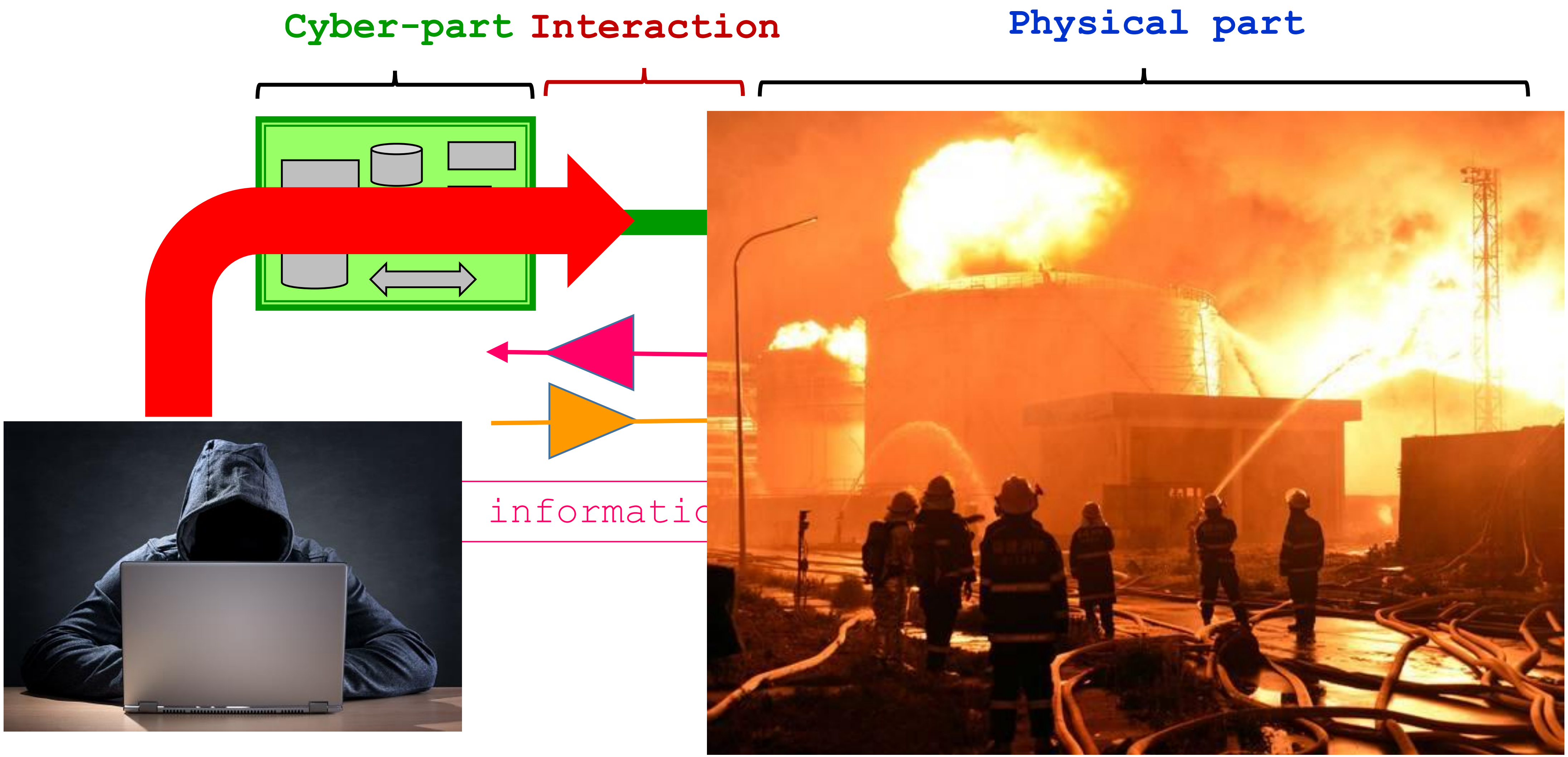
Cyber-physical systems control more and more of our physical devices (Cars, planes, trains, …)

https://tr3.cbsistatic.com

⇒ **Dependability** becomes the key issue

Dependable **architecture** (resilience)	Dependable **software** (trustworthiness)	Dependable **implementation** (correctness)

Risk: Cyber-Physical Attacks



Safety

CPS-Example: **Safety Risk**

Toyota "Unintended Acceleration" has killed 89 People

https://www.cbsnews.com / May 25, 2010

https://www.carscoops.com

Toyota Is Fined $1.2 Billion for Concealing Safety Defects

March 19, 2014

https://m.eet.com/media/1114769/0411esdbarr02.jpg

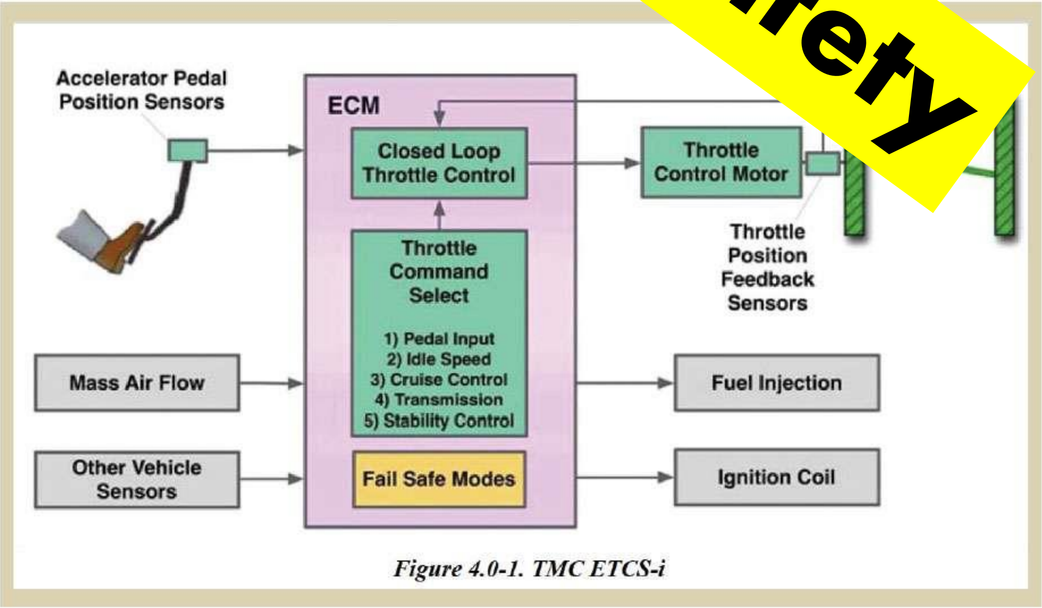
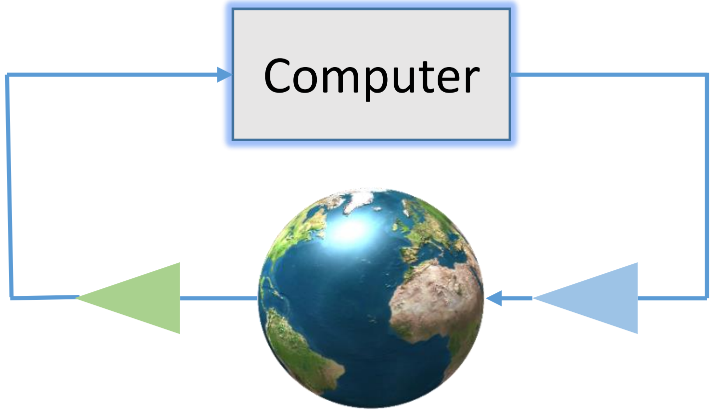


Figure 4.0-1 on page 13 of NASA Report shows the ETCS-i.[5]

All of these issues involved unrealistic timing delays in the multiprocessing, asynchronous software control flow. NASA also spent time simulating possible race conditions due to worrisome "recursively nested interrupt masking"



Fault, Failure

Attack, Intrusion

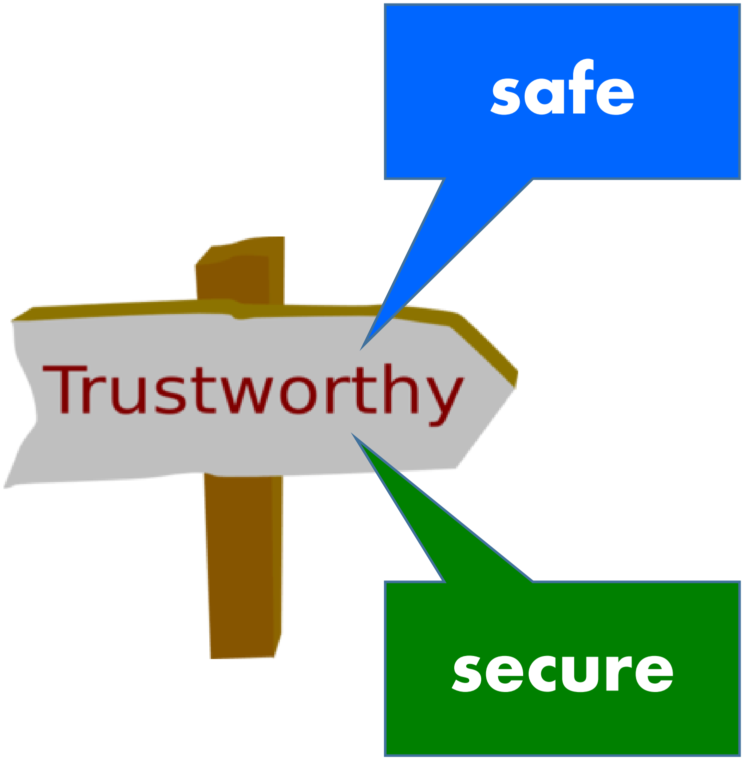
Risk

= Inherent **property** of cyber-physical systems

Risk Management

= Decisive part of systems engineering

Trustworthy Cyber Physical Systems



http://images.dailykos.com

https://www.jigsaw.ie

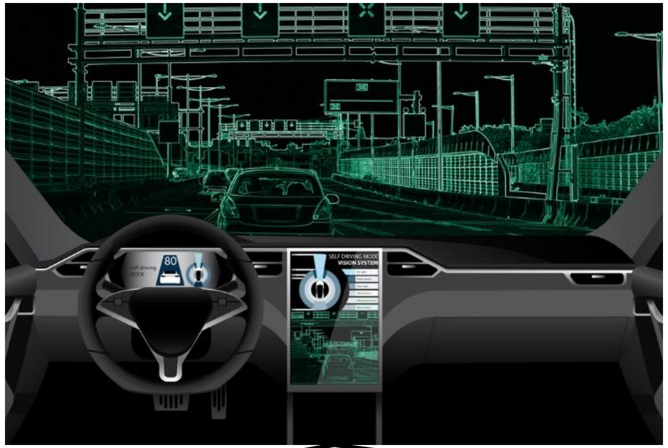
Trustworthiness of a system is the *expectation*
that the system will do what it should do,
and not do what it should not do,
both in expected and in unexpected conditions

http://www.anzor.com.au

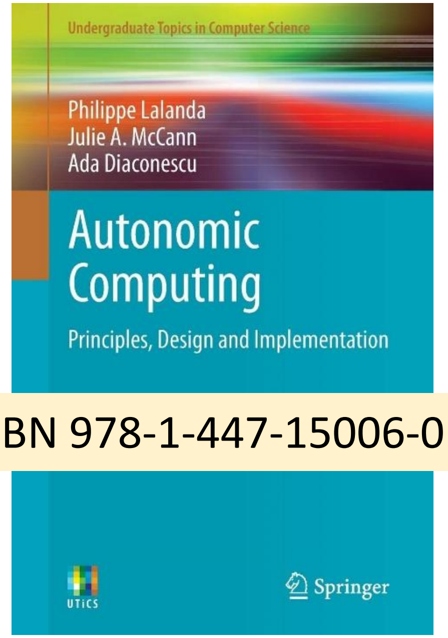
Dependability

Dependability Architecture Principles (Part 4)

Algorithmic/ Autonomic CPS



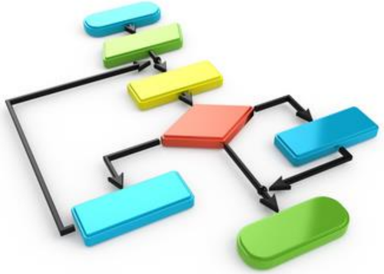
Cyber-Physical System



ISBN 978-1-447-15006-0

Algorithmic Computing

Autonomic Computing



http://www.bobology.com

The «programmer» must think of all *possible* cases and decisions *beforehand*

ANALYZE → PLAN

MONITOR

Knowledge & Artificial Intelligence

EXECUTE

Input

Output

Machine Learning

Machine reasoning

Machine Decisions

Example: Flight Assistance System

https://www.aitrends.com

Computer

Machine Decision

Right? Wrong?

Safe? Unsafe?

Timely? Late?

Ethical?

Legally OK? Law violation?

Socially acceptable?

Transparent?

Sensors

http://www.yalescientific.org

Actuators

Example: Boeing 737Max8 Anti-Stall Control

https://www.abc.net.au

Lion Air Flight 610: On 29 October 2018, the Boeing 737 MAX 8 crashed into the Java Sea 12 minutes after takeoff, killing all 189 passengers and crew

https://www.aerotelegraph.com

Ethiopian Airlines Flight 302: Six minutes after takeoff, the plane crashed near the town of Bishoftu, Ethiopia, killing all 157 people aboard.

Both planes crashed **nose-down**

What happened?

The 737 MAX was equipped with new, more fuel-efficient engines

https://leehamnews.com

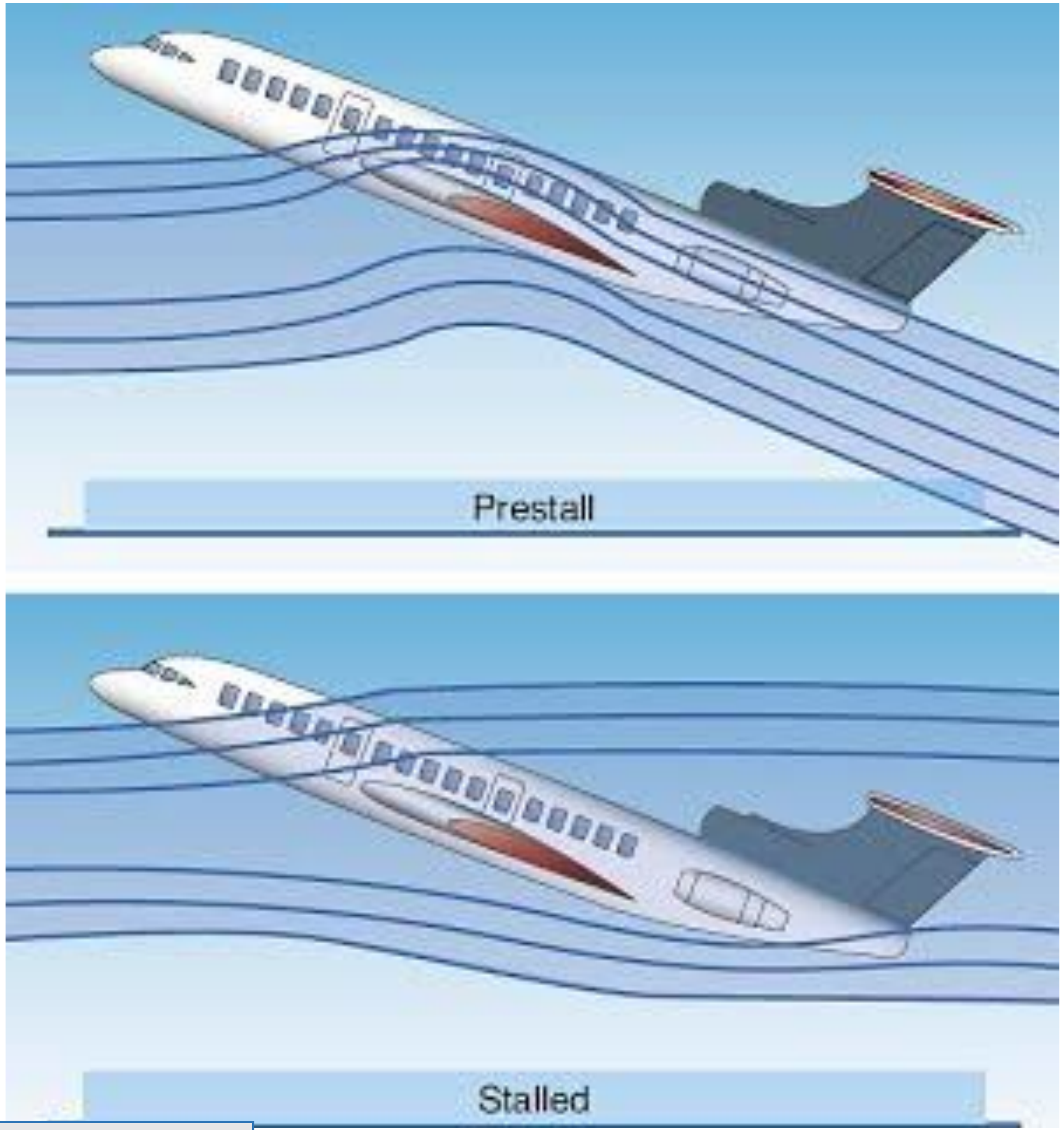
The larger engines augmented the risk of **stalling**

MCAS takes readings from sensors to determine how much the plane's nose is pointing up or down. If the software detects the nose is pointing up at a dangerous angle it has the ability to automatically push the nose down in an attempt to **stop the plane stalling**

https://www.theguardian.com

Airflow
⇓
Lift

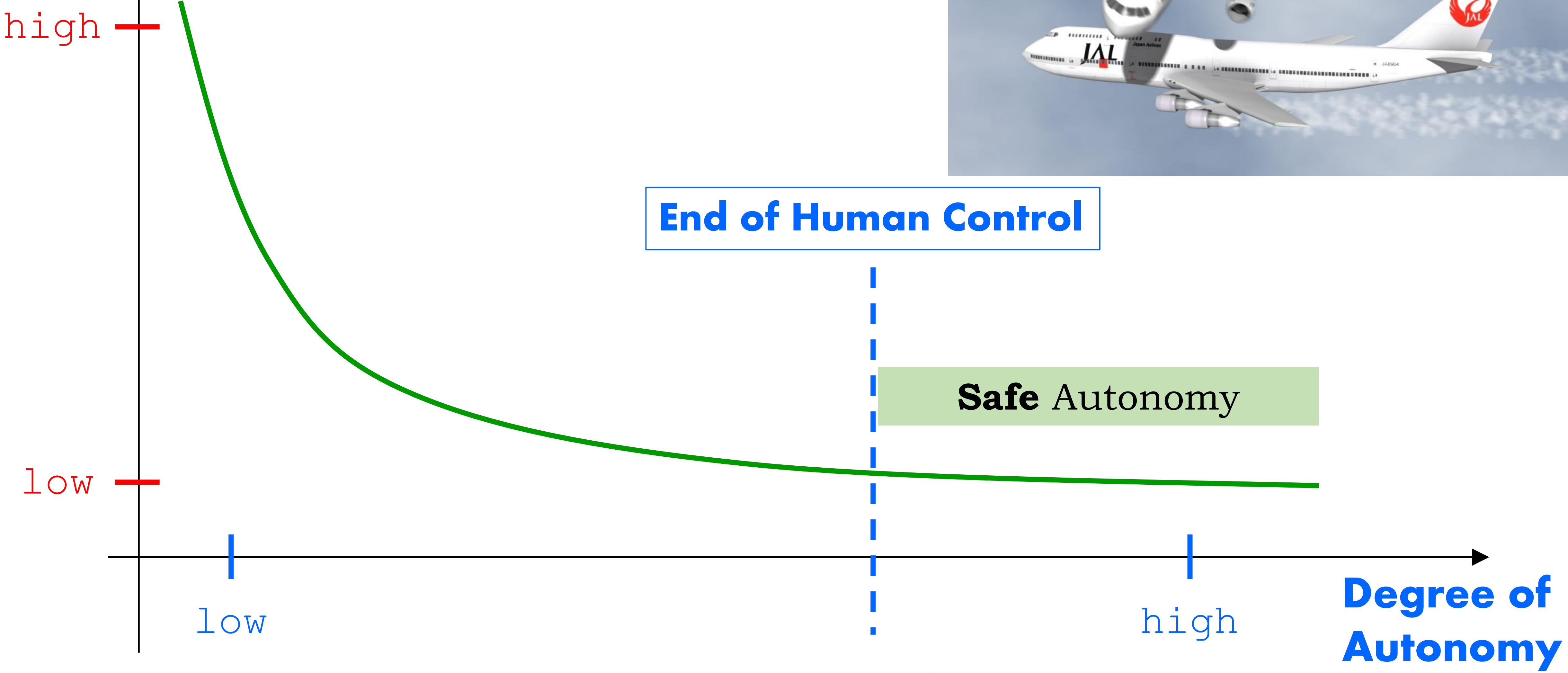
Lift **Loss**
⇓
Stalling



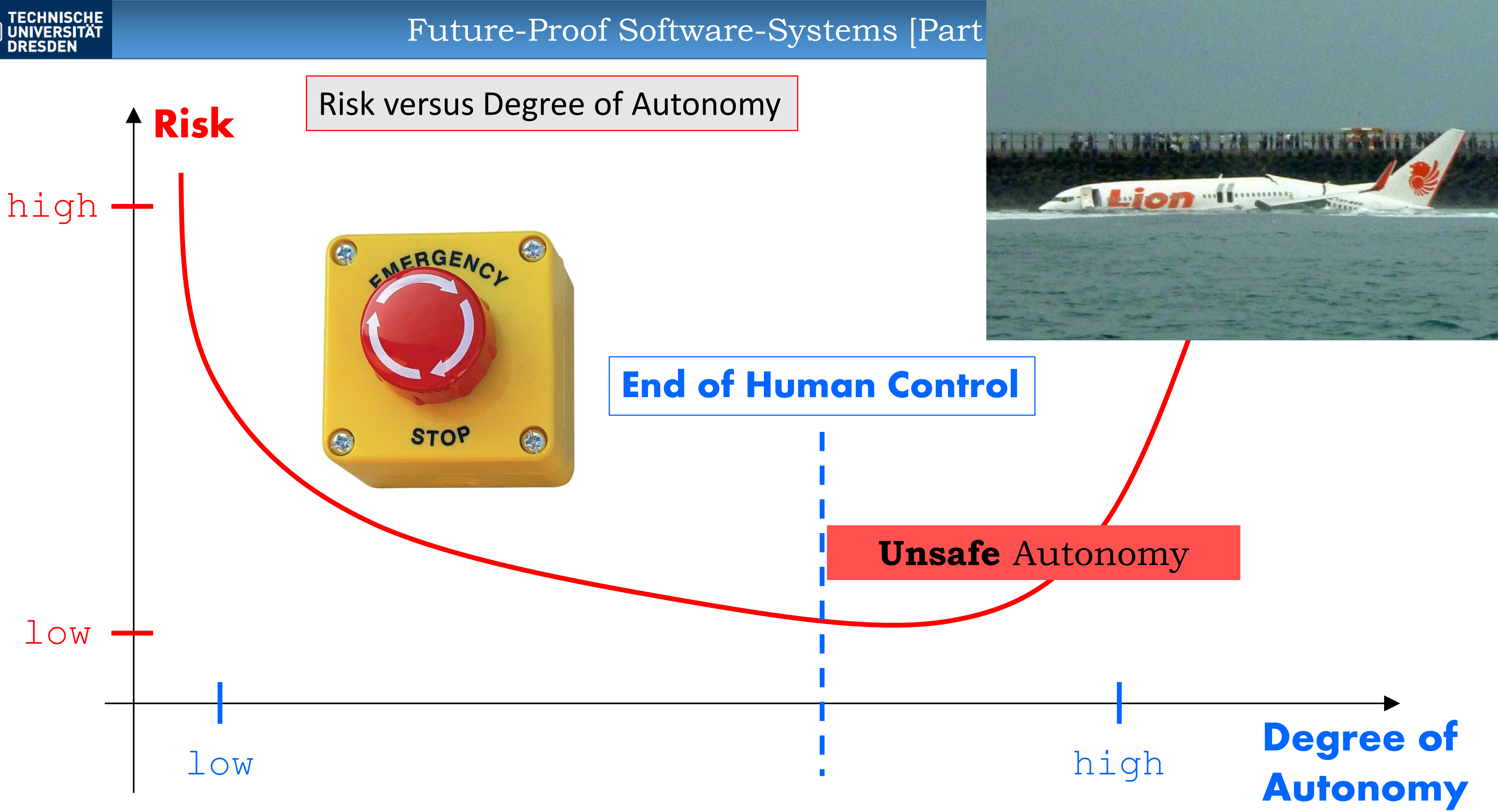
https://en.wikipedia.org

Typical CPS

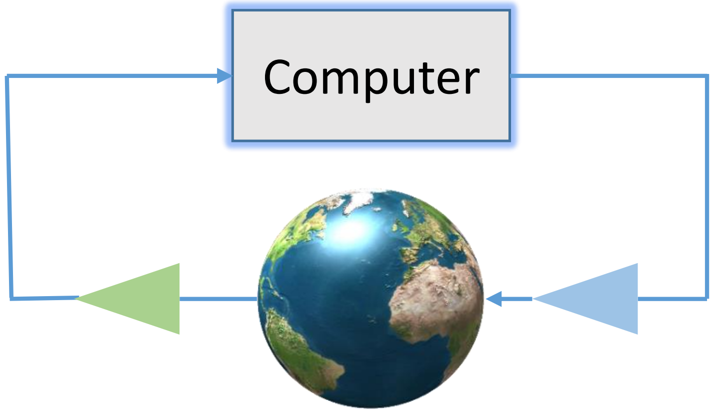
Risk versus Degree of Autonomy

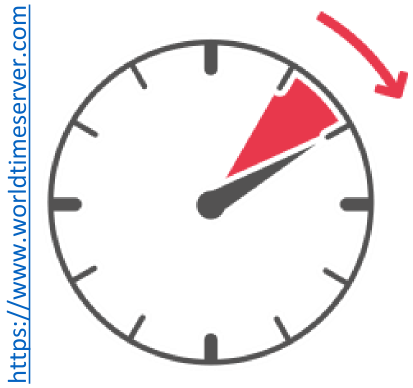


Risk versus Degree of Autonomy



Importance of Time



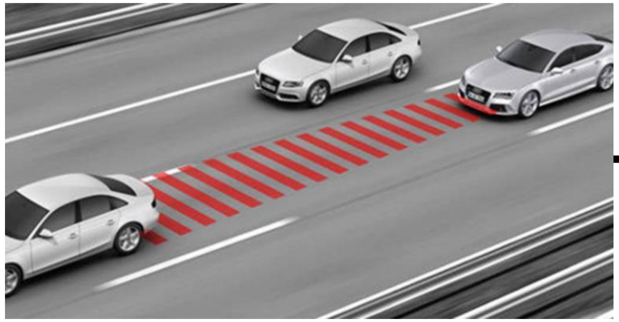


Cyber-Physical Systems are often **real-time systems**:

- Their reaction time input ⇒ processing ⇒ output must be **guaranteed**
- The timing conditions must be fulfilled at all times

Response Time

Start **Event**
[Receipt of Message]



https://www.electronicdesign.com

Start **Event**
[Radar Sensor Input]

Software Processing

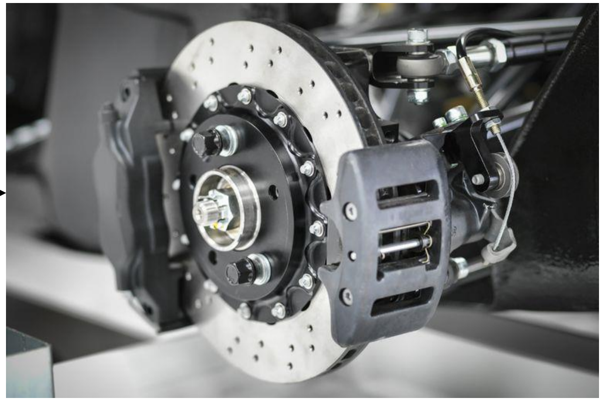
```
void CMymfc29EView::OnUpdateClockoleCreatealarm(CCmdUI* pCmdUI)
{
    pCmdUI->Enable(m_clock.GetInterfacePtr() != NULL);
}

void CMymfc29EView::OnClockoleLoad()
{
    if(m_clock.CreateInstance(__uuidof(Document)) != S_OK)
    {
        AfxMessageBox("Clock component not found");
        return;
    }
    try
    {
        m_clock->PutFigure(0, COleVariant("XII"));
        m_clock->PutFigure(1, COleVariant("III"));
        m_clock->PutFigure(2, COleVariant("VI"));
        m_clock->PutFigure(3, COleVariant("IX"));
        OnClockoleRefreshtime();
        m_clock->ShowWin();
    }
    catch(_com_error& e)
    {
        AfxMessageBox(e.ErrorMessage());
    }
}

void CMymfc29EView::OnUpdateClockoleLoad(CCmdUI* pCmdUI)
{
    pCmdUI->Enable(m_clock.GetInterfacePtr() == NULL);
}
```

Worst Case Execution Time (WCET)

Action



http://www.fahrschule-luckau-dahme.de

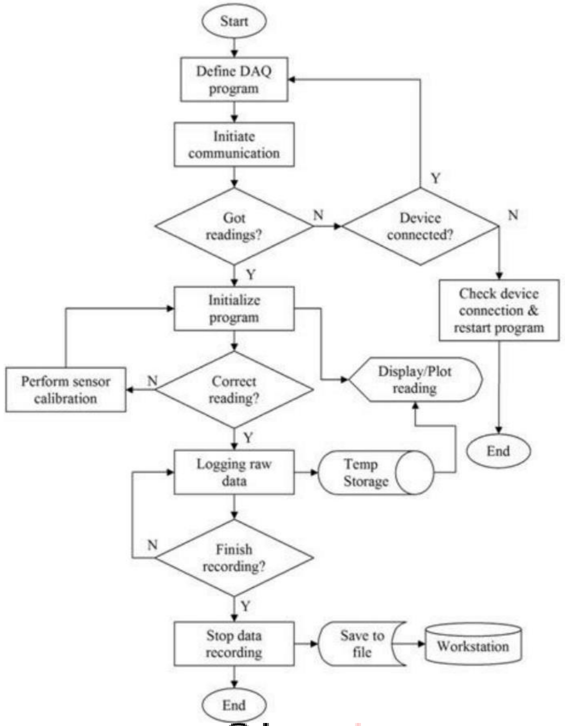
https://www.thoughtco.com

Action

Response Time

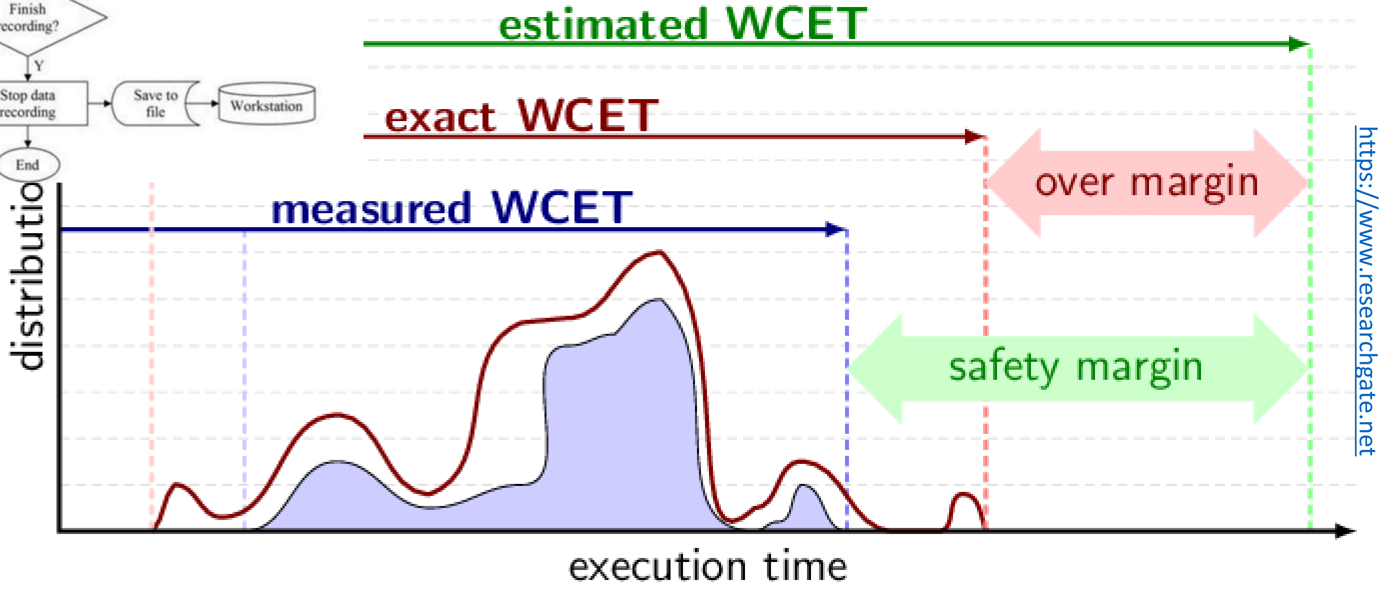
Total elapsed time from an event to the completed, correct action

https://www.amazon.de



Worst Case Execution Time (WCET)

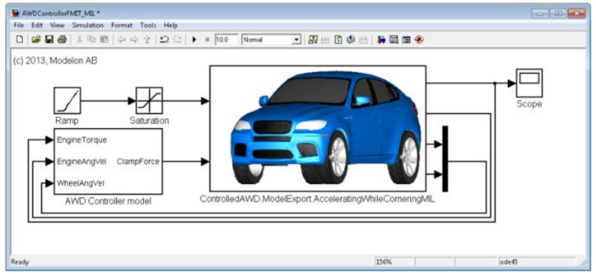
Longest possible time for the execution of the software action



https://www.researchgate.net

⇒ Critical Parameters in Real-Time Cyber-Physical Systems

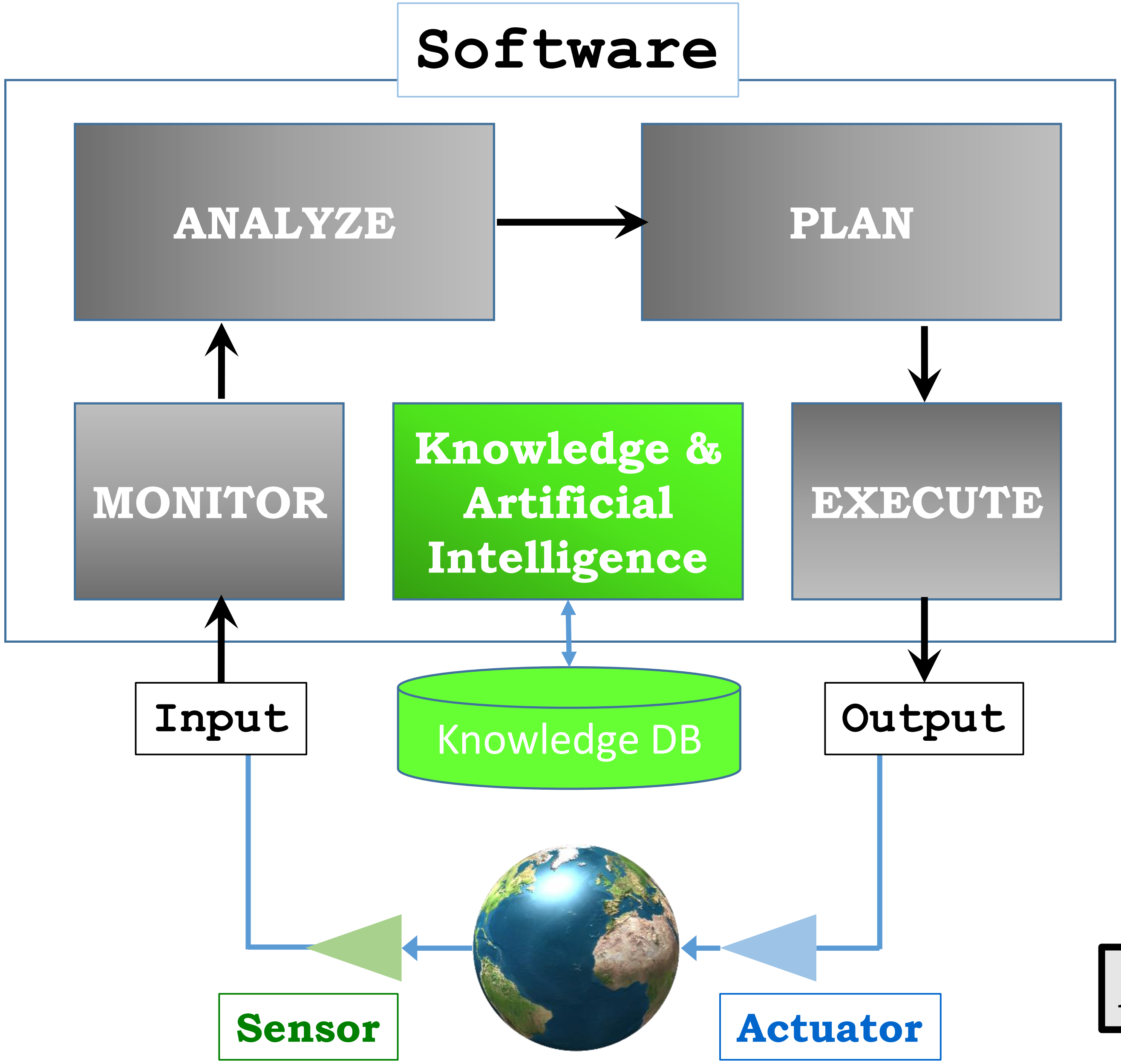
Real-World Model



http://www.modelon.com

Requirements
- Functional
- Quality properties

Specifications
- Functional
- Quality properties



Validation

Certification

Anatomy of a CPS

http://worldartsme.com

Trustworthy CPS and CPSoS:

Cyber-physical system or cyber-physical system-of-systems
with an adequate degree of security and safety
to fulfill the trust expectations of its users

Risk Consideration

Protection from malicious activities

«*The system does what it should - and does not what it should not*»

Protection from failures, faults, errors, malfunctions

https://www.raymancini.academy

Risk Managment = Decisive Part of Systems Engineering !

A trustworthy system is the result of competent and responsible engineering

Future-Proof Software-Systems [Part 3C]

http://vpn4torrents.com

Security Vulnerability

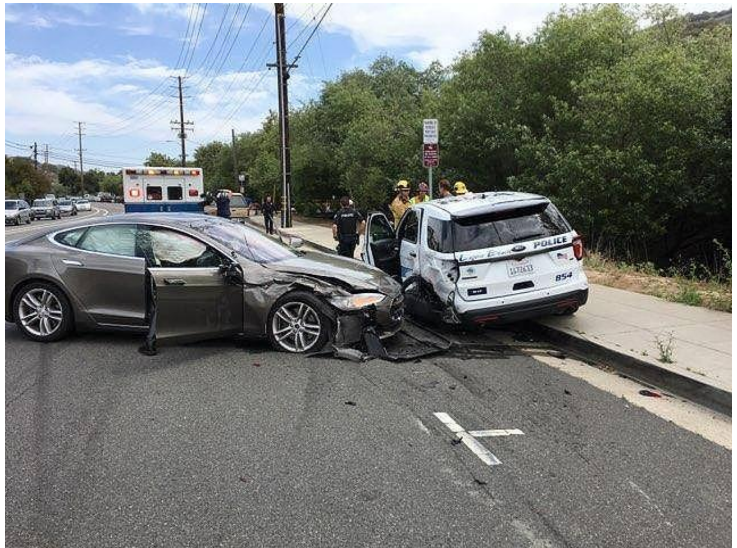
Safety Weakness

https://www.pinterest.com

https://thehackernews.com

https://thehackernews.com

https://www.focus.de



http://letshack2day.blogspot.com

https://www.dw.com

https://www.traveloffpath.com

https://www.aeteurope.com

Security

Information Security

Information Security protects the confidentiality, integrity, and availability (CIA) of computer system data and functionality from unauthorized and malicious **accesses**

Functional Security

Functional security protects the software-system from **malicious, infiltrated code**, both from the outside and from the inside of the organization

- Confidentiality
- Integrity
- Availability
- …

Quality of Service Properties for **SECURITY**

http://www.osalp.com.pk

Safety

Safety

Safety is the state of being protected against faults, errors, failures, or any other event that could be considered non-desirable in order to achieve an acceptable level of risk concerning loss of property, damage to life, health or society, or harm to the environment.

Safety is
the sum of all accidents
that did <u>not</u> happen

- Fault-tolerance
- Graceful degradation
- Availability
- Fail-safe states
- …

Quality of Service Properties for **SAFETY**

A trustworthy system is the result of competent and responsible engineering

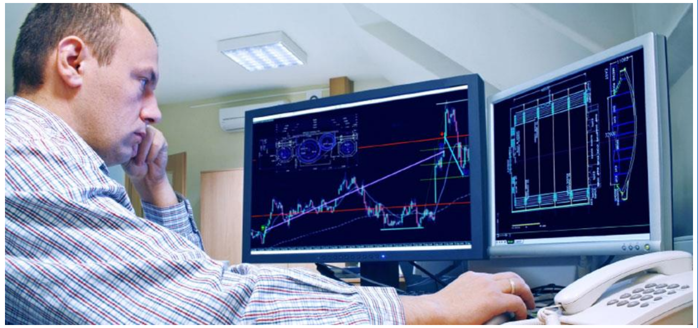
Competence: The ability to do something well
https://dictionary.cambridge.org/dictionary/english/competence

https://kayaconnect.org

A duty or obligation to satisfactorily perform or complete a task that one must fulfill, and which has a consequent penalty for failure
http://www.businessdictionary.com

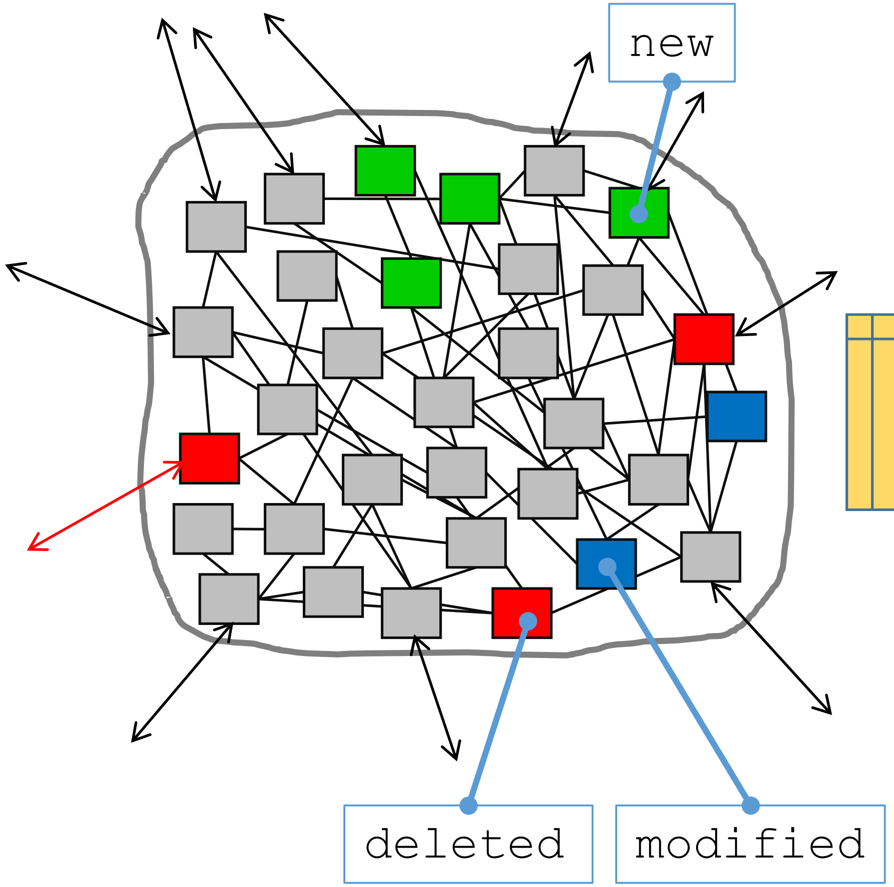
https://w-dog.net

Systems engineering is an interdisciplinary field of engineering and engineering management that focuses on how to design, implement, maintain and manage complex systems over their life cycles
https://en.wikipedia.org/wiki/Systems_engineering

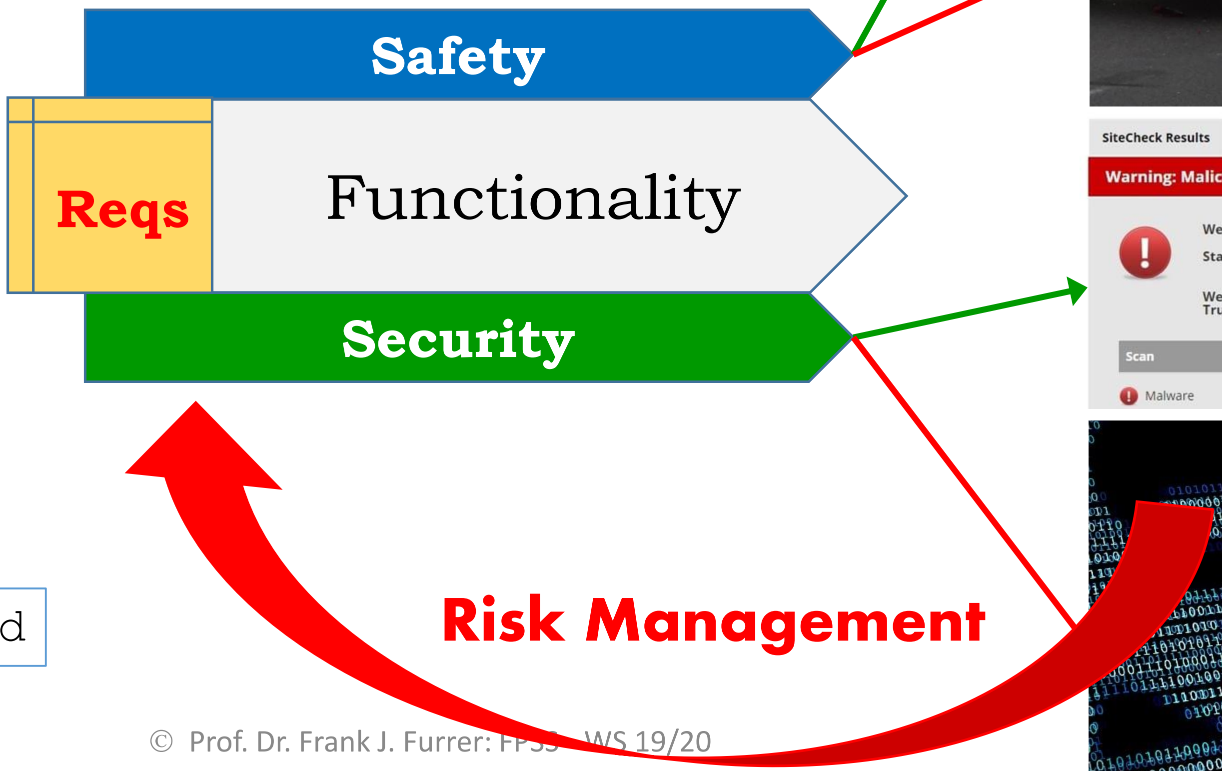


https://www.charteredengineers.sg

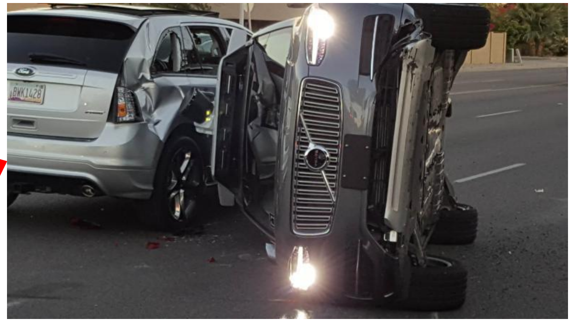
System extension
→ Project



Safety and Security concerns/requirements have higher priority then functionality



https://www.indiatimes.com

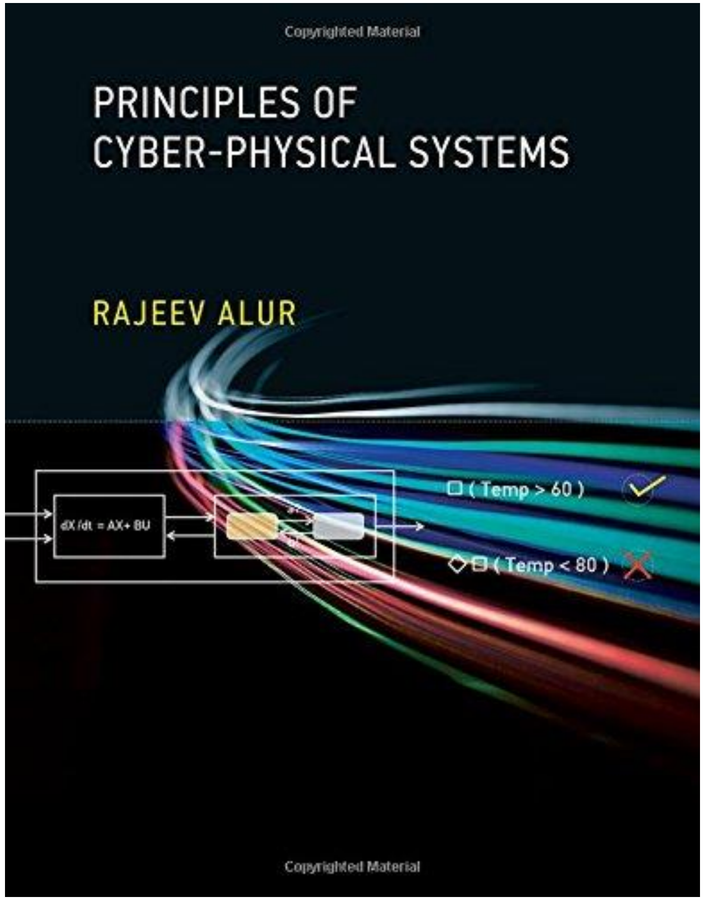


https://bgr.com

https://www.seoblog.com

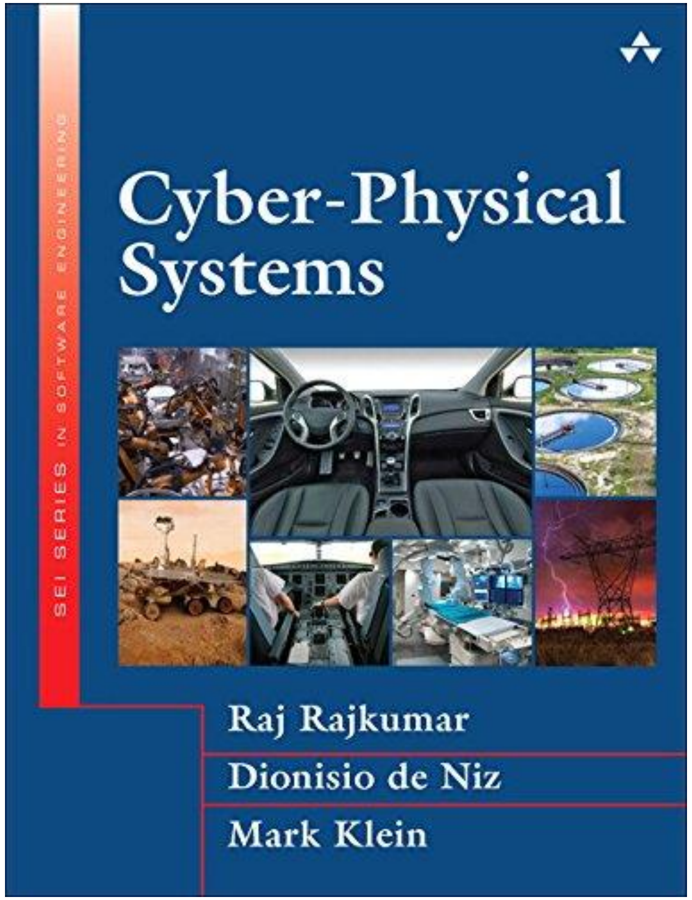
https://news.softpedia.com

Textbook



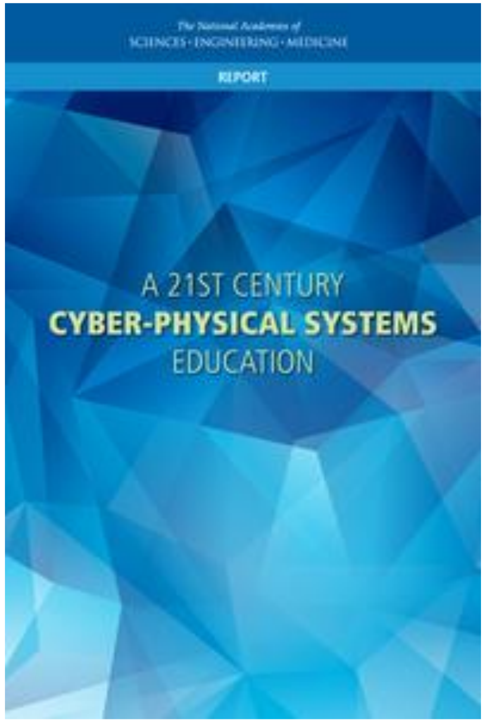
Rajeev Alur:
Principles of Cyber-Physical Systems
MIT Press, Cambridge, USA, 2015. ISBN 978-0-262-02911-7

Textbook



Raj Rajkumar, Dionisio De Niz, Mark Klein:
Cyber-Physical Systems
Addison Wesley Inc., USA, 2016
ISBN 978-0-321-92696-8

Textbook



USA National Academies of Sciences, Engineering, and Medicine:
21st Century Cyber-Physical Systems Education
2016, ISBN 978-0-309-45163-5
Downloadable from :
https://www.nap.edu/download/23686

Recommendations

Cyber-Physical Systems (CPS)

1. *Dependability properties* (safety, security, …) always supersede functionality requirements – in specifications, architecture, design, implementation, and operation
2. Trustworthy CPS are a result of competent and responsible engineering: Build, maintain, and monitor your development processes accordingly
3. Implement *monitoring capabilities* to predict or detect abnormal or dangerous behaviour

Systems-of-Systems (SoS)

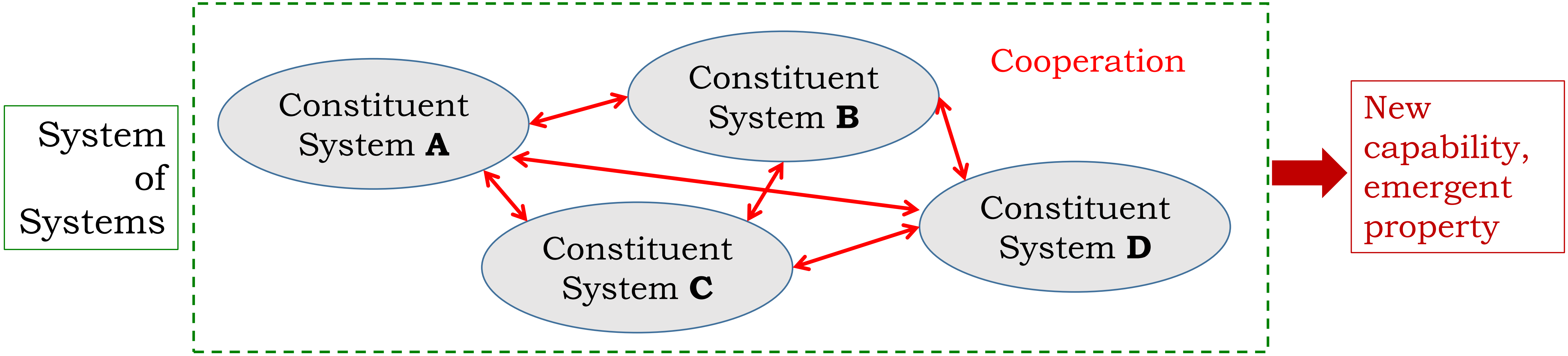
System-of-Systems (SoS)

DEFINITIONS

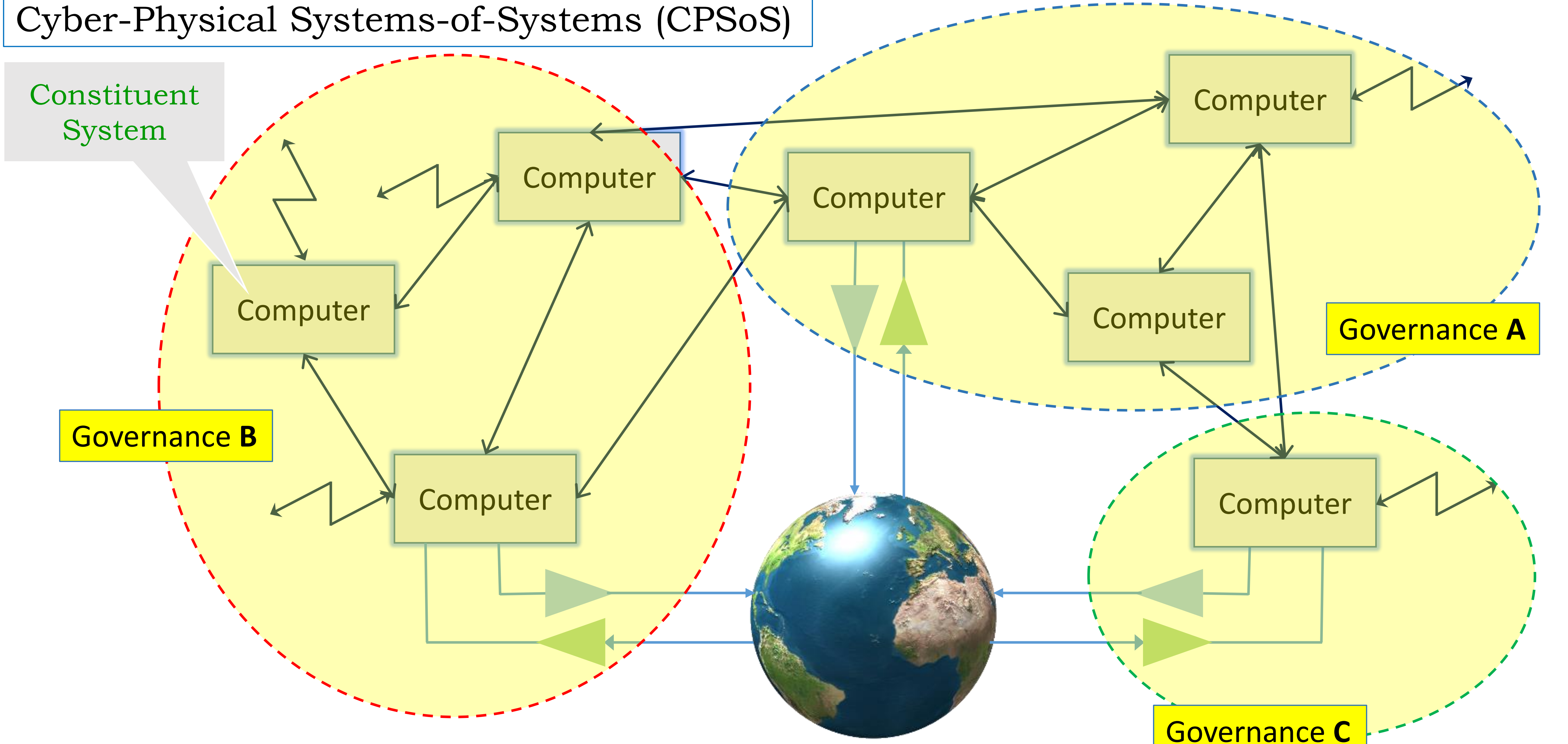
A **system of systems (SoS)** brings together a set of *cooperating systems* for a task that none of the systems can accomplish on its own (= emergent property).

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

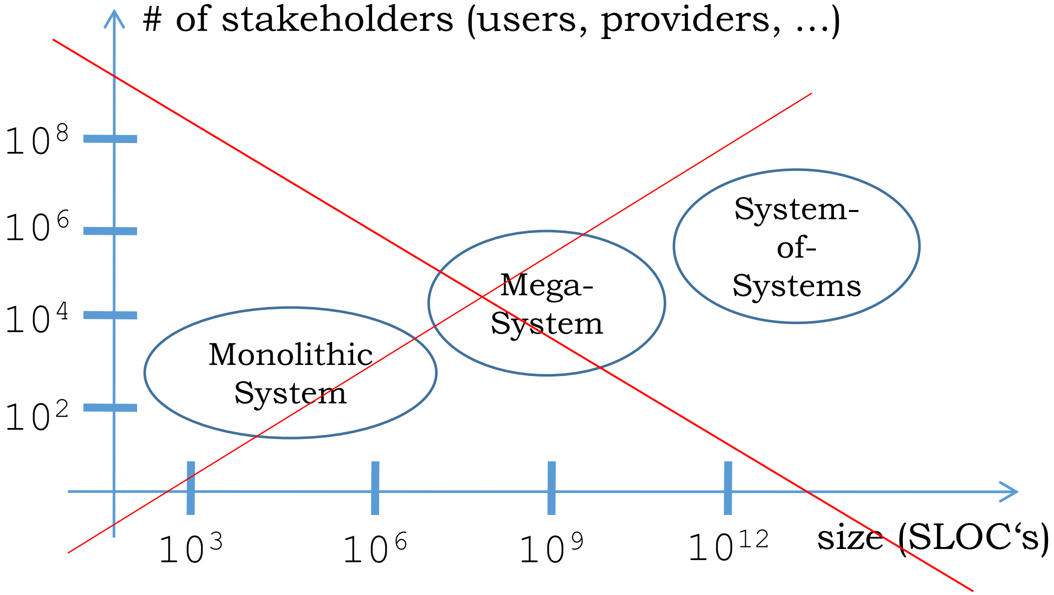
ISO/IEC/IEEE 15288 Annex G



Cyber-Physical Systems-of-Systems (CPSoS)



What is a system-of-systems ?



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 Behavior
5. Geographic Distribution

[5 Maier criteria, 1998]

[5 Maier criteria, 1998]

Systems-of-systems characteristics:

1. Operational Independence of the Elements

Managerial Independence of the Elements

3. Evolutionary Development

Emergent Behaviour

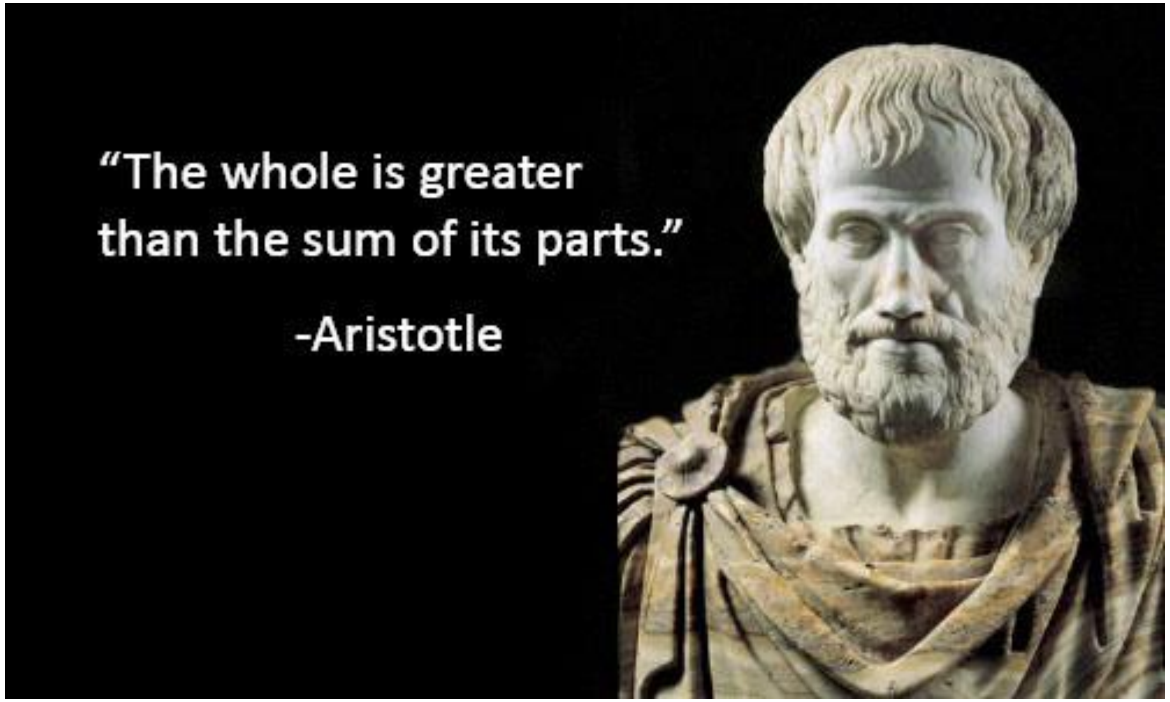
5. Geographic Distribution

Governance

Total = more than the sum of its parts

Collaboration

Collaboration Contract
- functional
- operational
- commercial
- legal

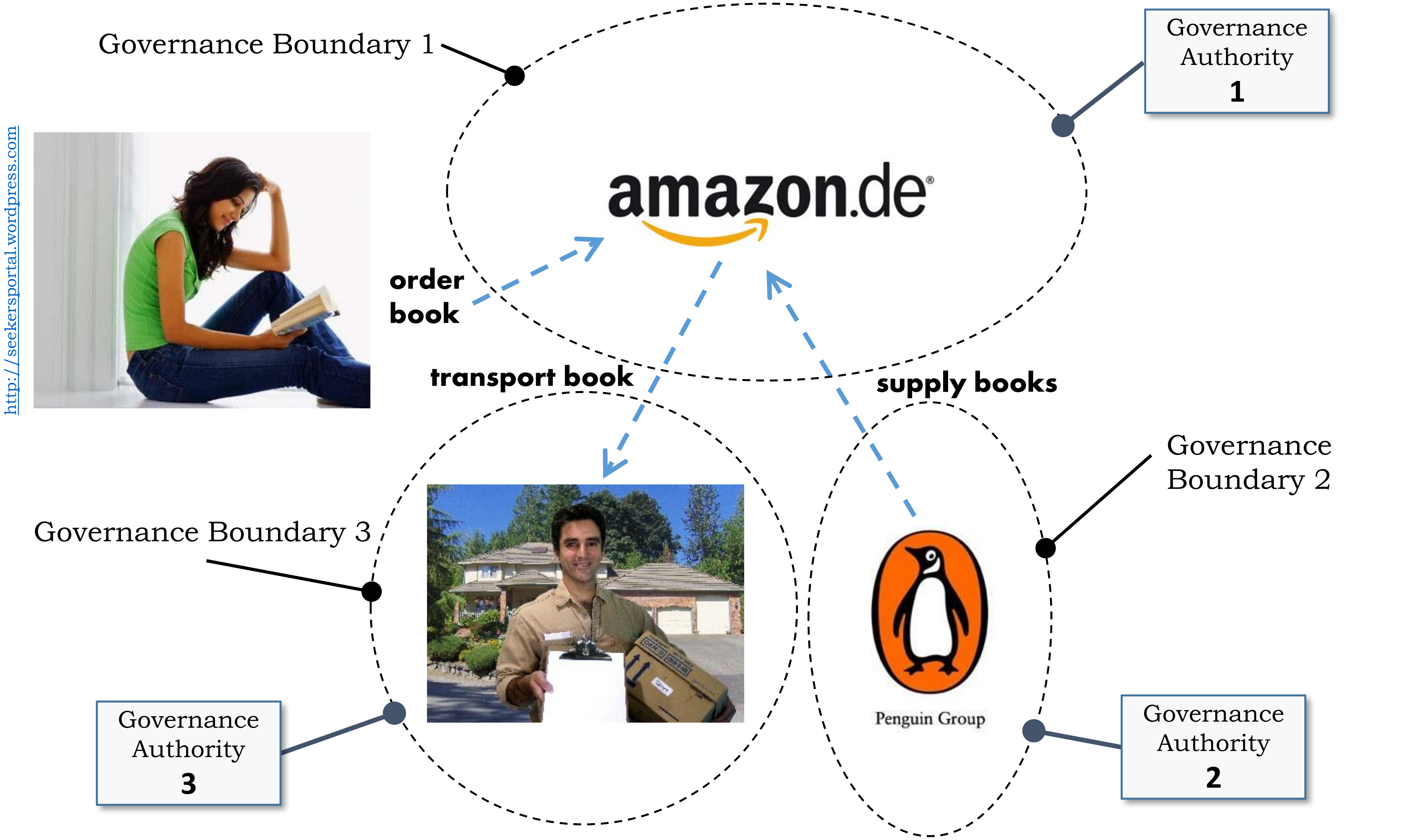


https://i.pinimg.com

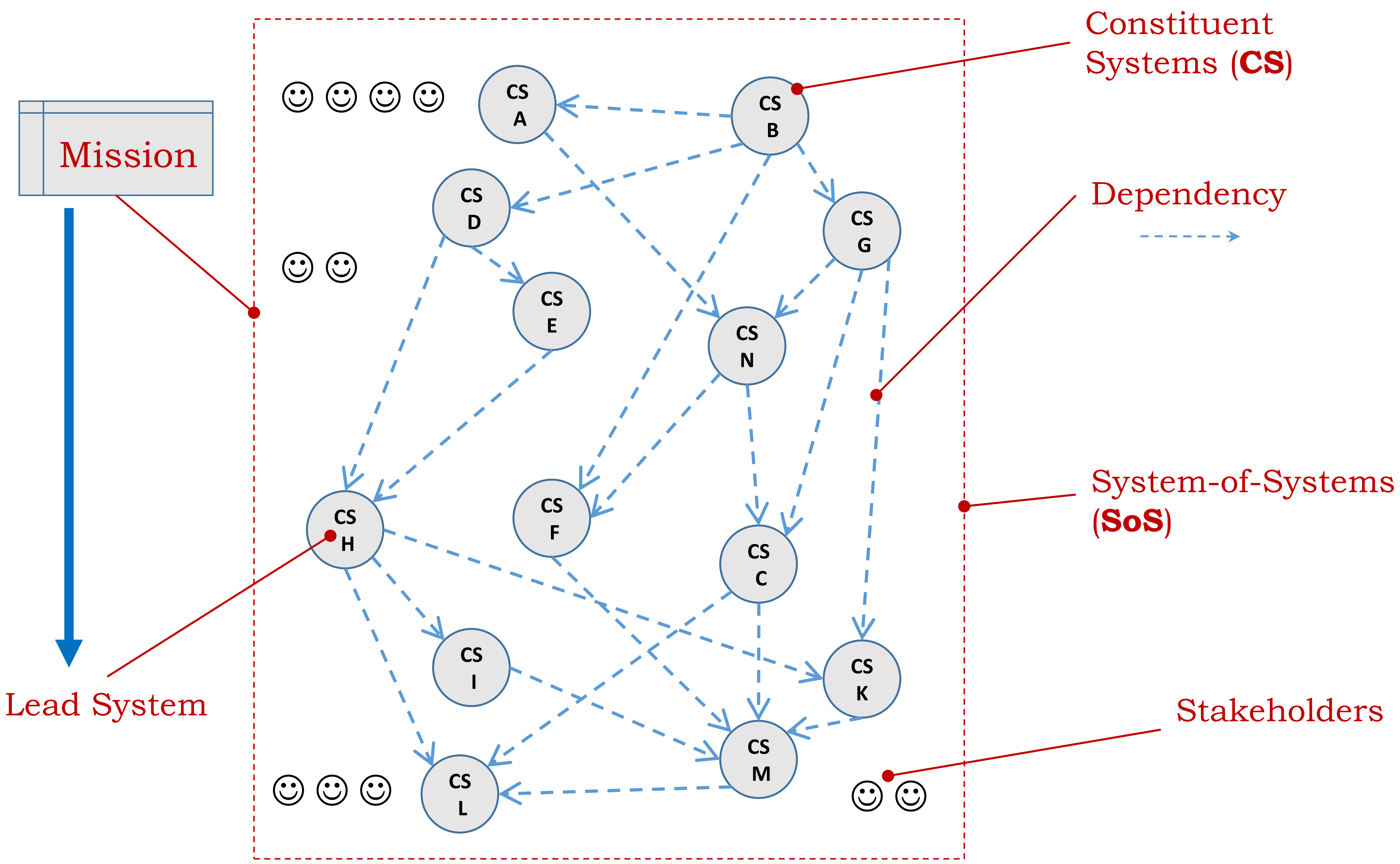
Managerial Independence of the Elements

Governance

SoS Example: AMAZON Business Model

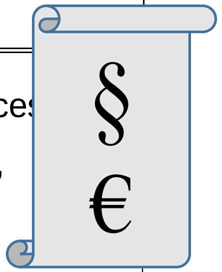


SoS (Systems-of-systems) Terminology

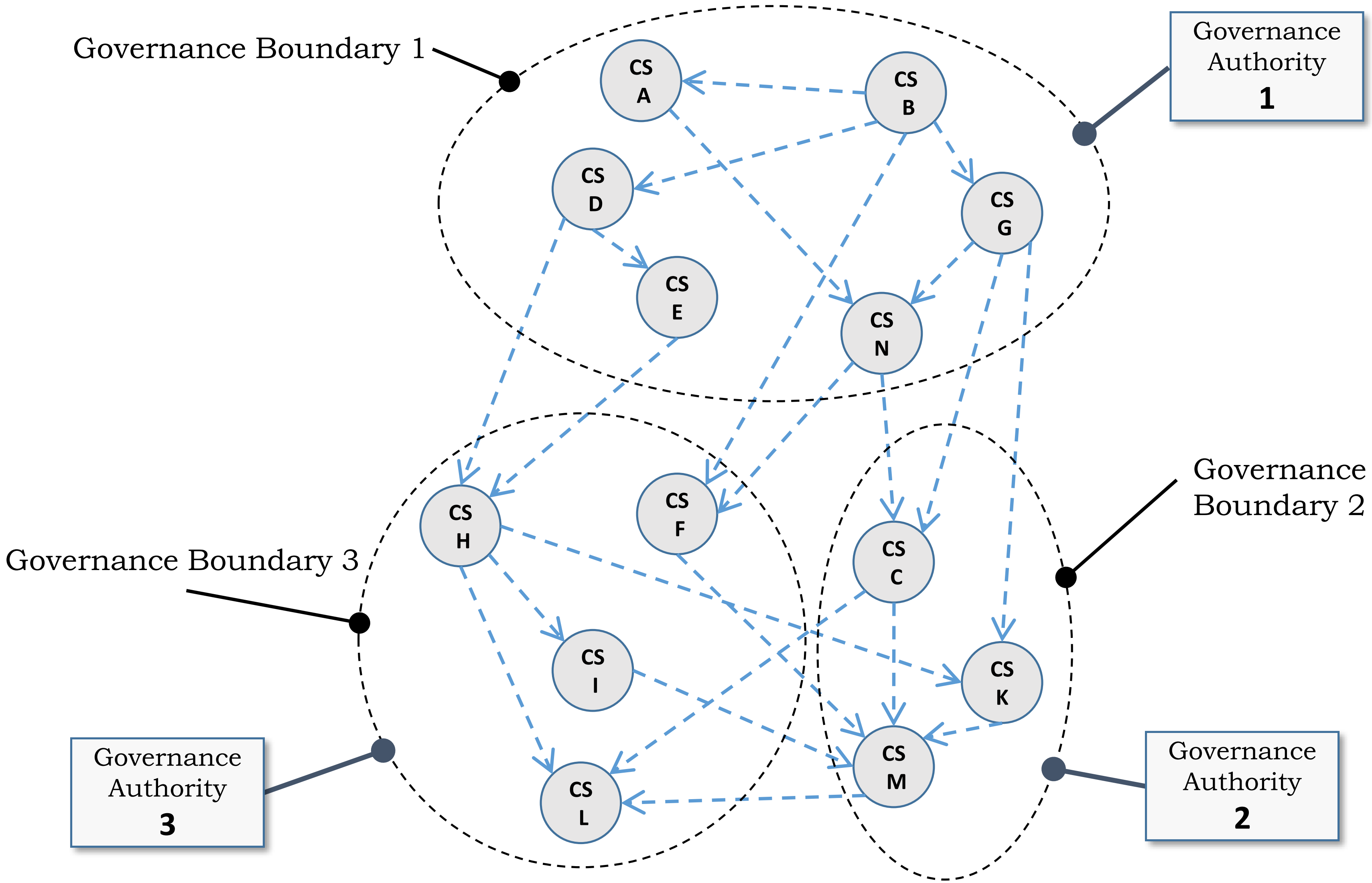


SoS
(Systems-of-systems) Classification

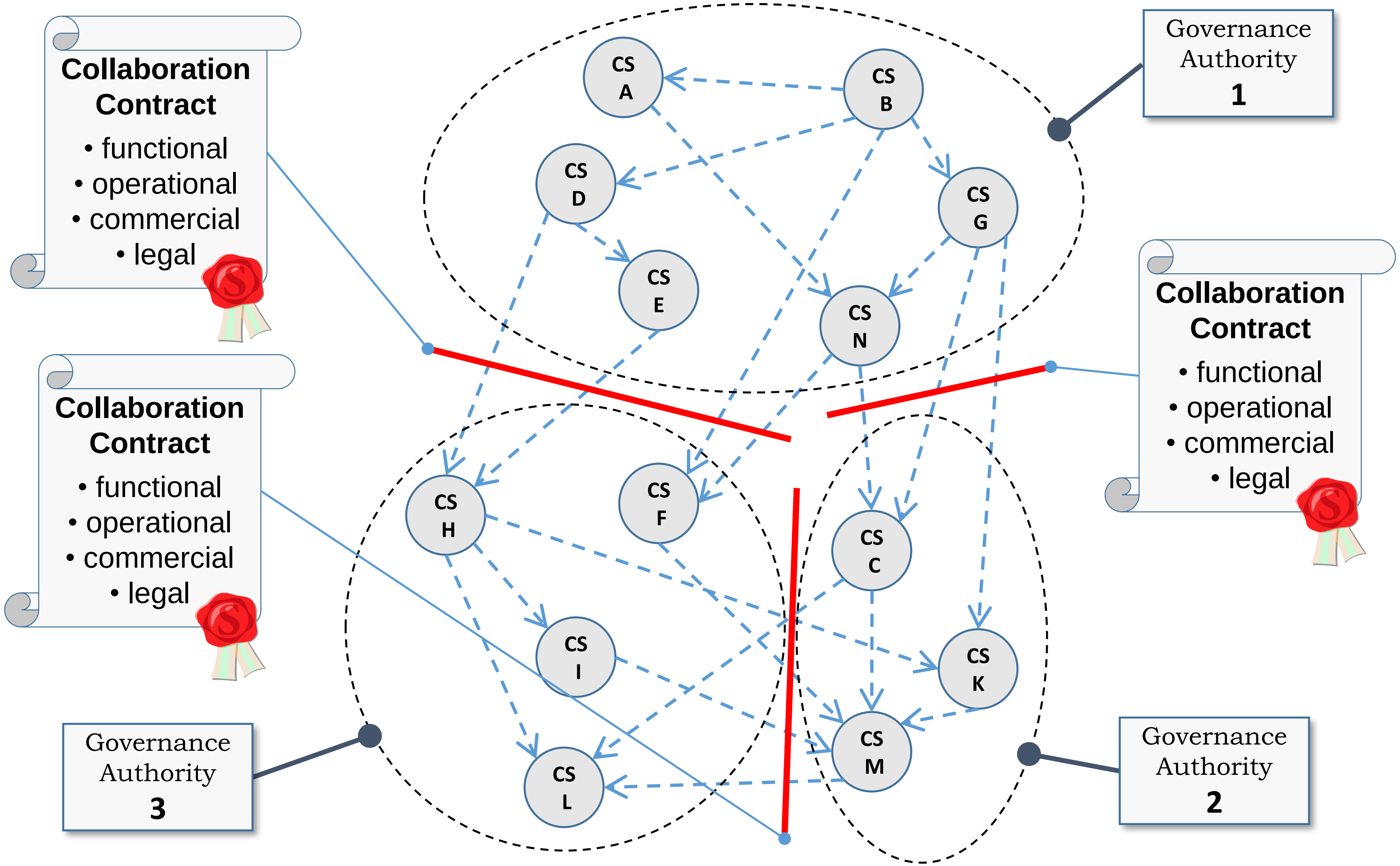
Type of SoS	Description
Directed	Directed SoS are those in which the integrated system-of-systems **is built and managed to fulfill specific purposes**. It is **centrally managed** during long-term operation to continue to fulfill those purposes as well as any new ones the system owners might wish to address. The constituent systems maintain an ability to operate independently, but their normal operational mode is **subordinated to the central managed SoS purpose**
Acknowledged	Acknowledged SoS have recognized objectives, a designated manager, and resource the SoS. However, the **constituent systems retain their independent ownership**, objectives, funding, and development and sustainment approaches. Changes in the systems are **based on collaboration** between the SoS and the systems
Collaborative	In collaborative SoS the **constituent systems interact more or less voluntarily** to fulfill agreed upon central purposes. The central players **collectively decide** how to provide or deny service, thereby providing some means of enforcing and maintaining standards.
Virtual	Virtual SoS **lack a central management authority** and a centrally agreed upon purpose for the system- of-systems. Large-scale behavior emerges – and may be desirable – but this type of SoS must rely upon **relatively invisible mechanisms** to maintain it



Managerial Independence: **Governance**

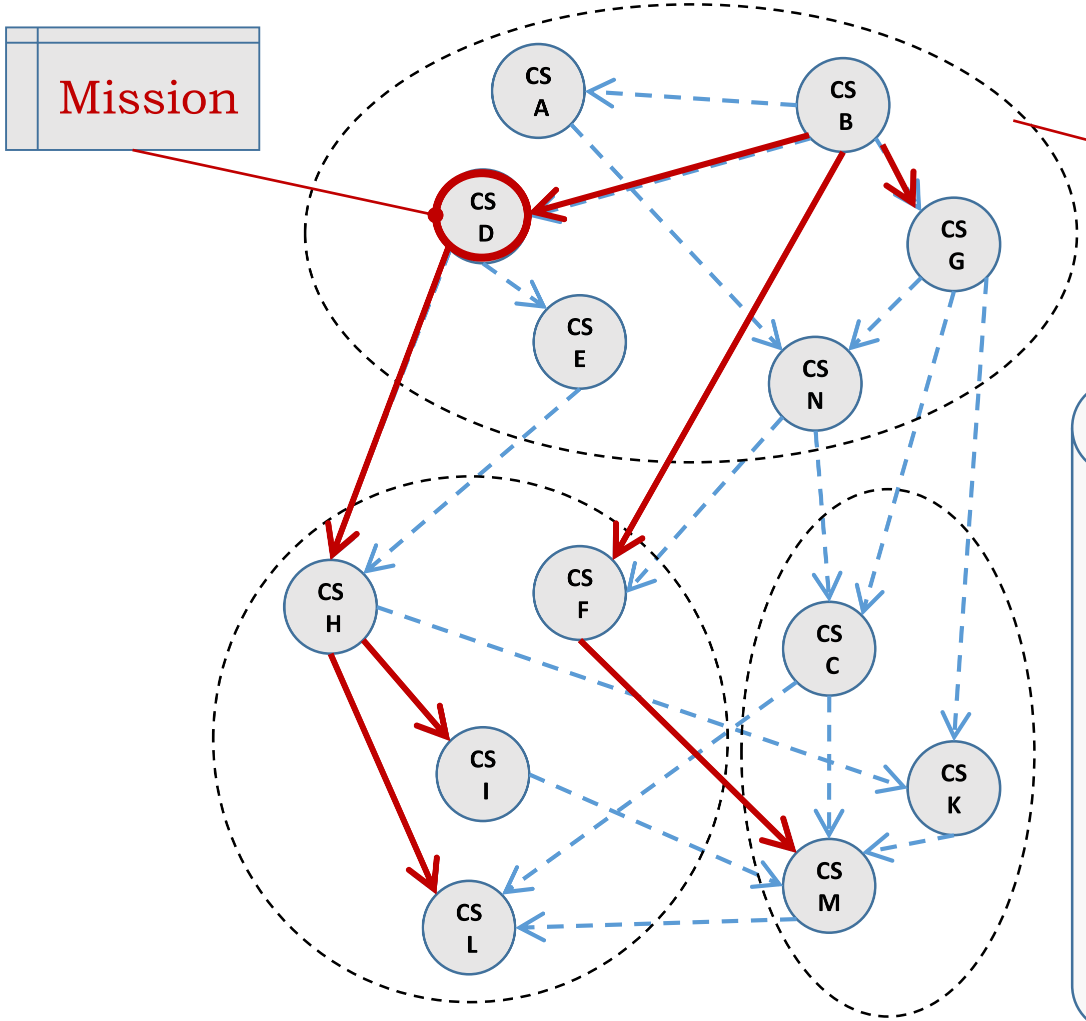


Managerial Independence: **Governance by Contract**



… now we understand SoS's and their challenges

… what does it mean for our future-proof software?



1) Transparent architecture
2) Explicit dependencies
3) Complete contracts
4) Risk mitigation
5) Monitoring and early response

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**

ISO/IEC/IEEE 15288 Annex G

emergent behavior & emergent properties

New, desired behaviour, information, or property

NOT

available from the individual constituent systems

= Reason for **building** the SoS

Example 1 for emerging properties: „flying“

http://www.mskgmbh.com

Constituent systems (CS) of an aircraft:

- engines
- body
- wings
- cockpit
- etc.

… none of the constituent systems is able to fly !

Assemble the essential constituent systems:

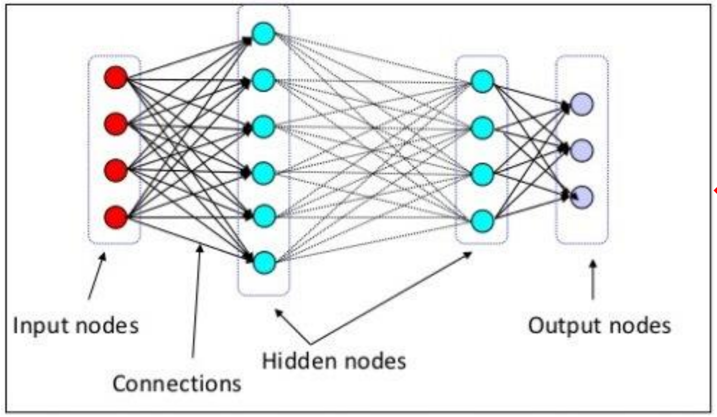
Emerging property: the assembly (= airplane) is able to **fly** !

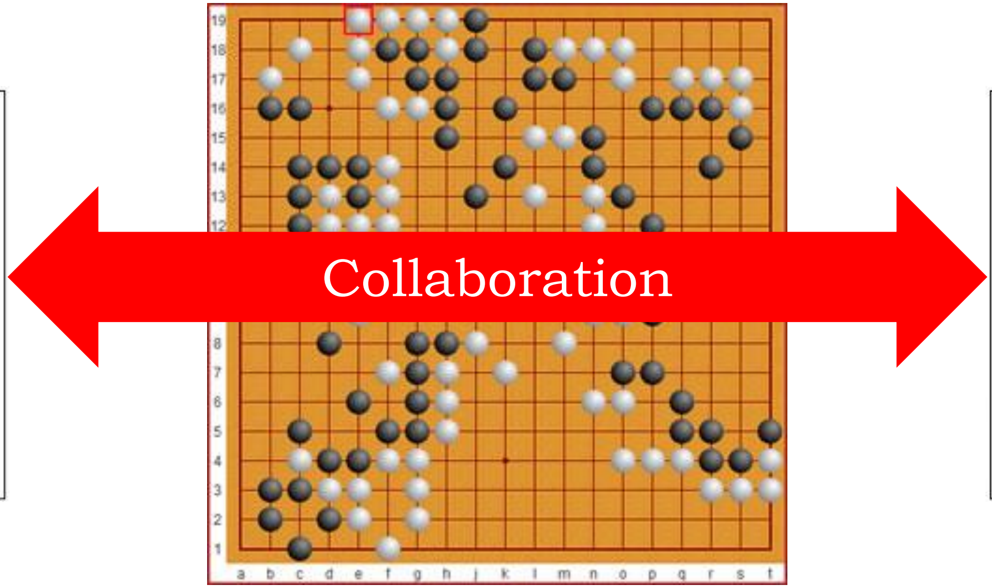
http://en.wikipedia.org/wiki/Airbus_A320_family

Example 2 for emerging properties: „AlphaGo Zero“

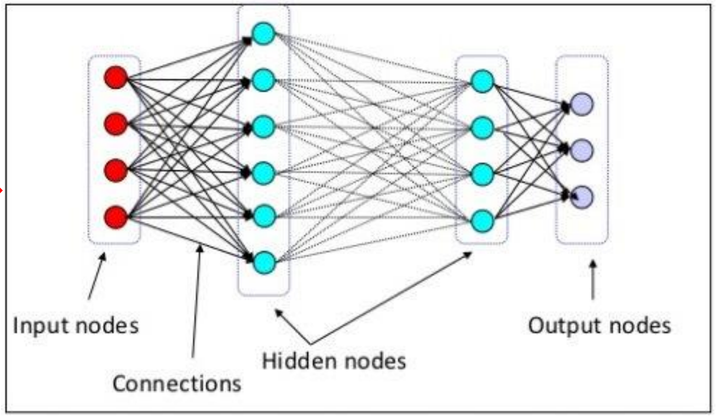
AlphaGo's team published an article in the journal Nature on 19 October 2017, introducing **AlphaGo Zero**, a version created without using data from human games, and stronger than any previous version

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 3 for emerging properties: „Landing Crash“

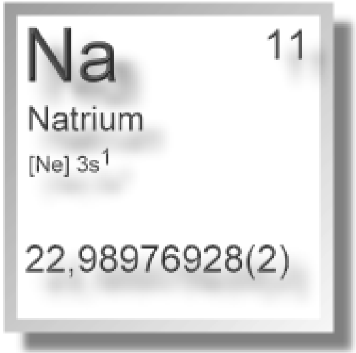
http://avioners.net/2011/03/airbus-a319-approach-to-landing.html

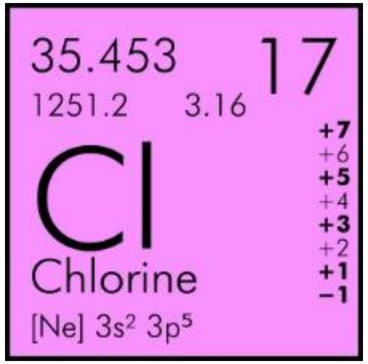
Constituent systems:
- Airplane (DC-8)
- Airport (Runway)

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

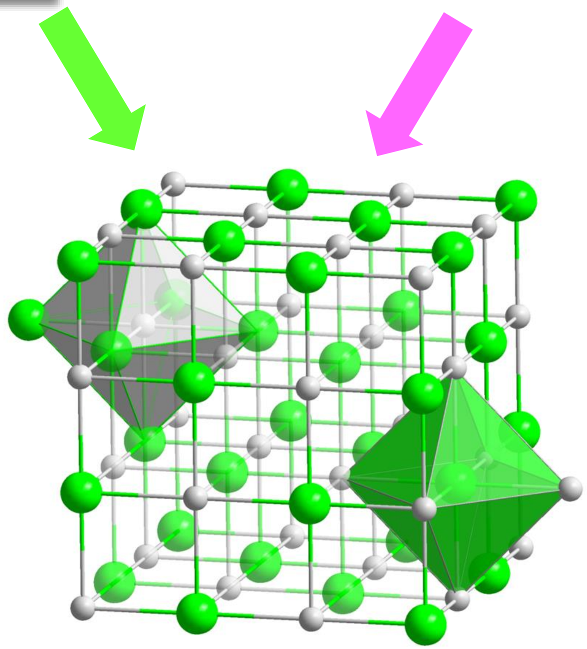
Example 4: Emergent Properties (NaCl)





Constituent systems:
Na
Cl

https://upload.wikimedia.org



Na+ Cl-

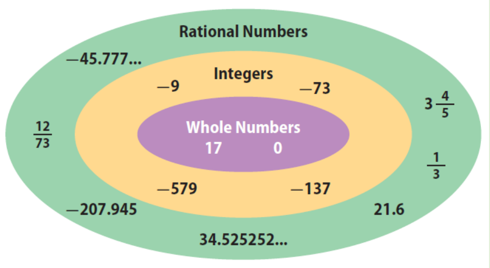
Table Salt
(= Natriumchloride NaCl)

https://de.wikipedia.org

Example 5

Emergent Data: Rational Numbers

Emergent Property: Rational Numbers



Constituent System #1

Elements: Even numbers

2	12	22	32	42	52	62	72	82	92
4	14	24	34	44	54	64	74	84	94
6	16	26	36	46	56	66	76	86	96
8	18	28	38	48	58	68	78	88	98
10	20	30	40	50	60	70	80	90	100

Operation: Multiplication

Data

Functionality

New Functionality: **Division**

Constituent System #2

Elements: Odd numbers

1	11	21	31	41	51	61	71	81	91
3	13	23	33	43	53	63	73	83	93
5	15	25	35	45	55	65	75	85	95
7	17	27	37	47	57	67	77	87	97
9	19	29	39	49	59	69	79	89	99

Operation: Multiplication

SoS Emergent Behaviour **Classification**

Emergence	Desirable/positive	**Undesirable/negative**
Expected emergent behavior	Reason for building the SoS (**SoS objective**) 	**Mitigate** by appropriate design measures, such as threat/risk analysis and countermeasures 
Unexpected emergent behavior	Sometimes (however, quite rarely) an SoS shows unexpected, **beneficial behaviour** 	<u>Un</u>expected & <u>un</u>desirable negative emergent behavior is one of the **critical risks** of most SoS 

Emergent behaviour
Emergent Information
Emergent properties

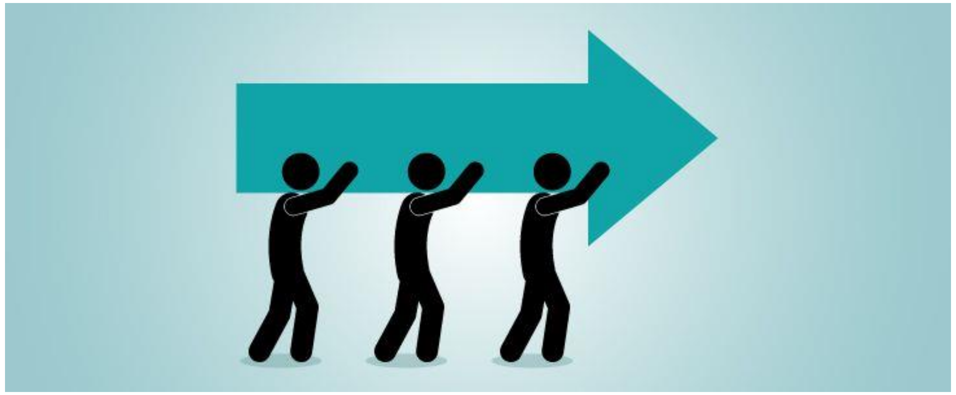
System A
- Properties «A»
- Behaviour «A»

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

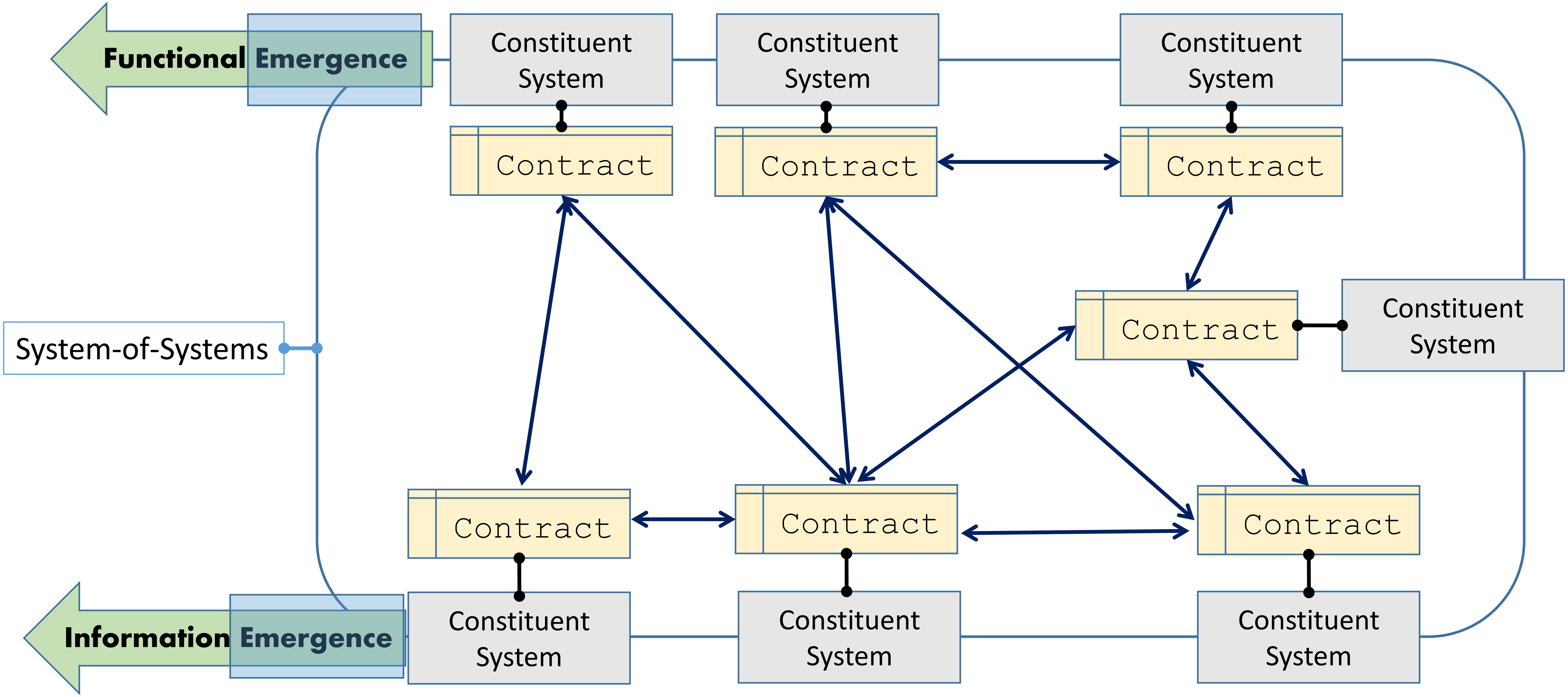
…

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

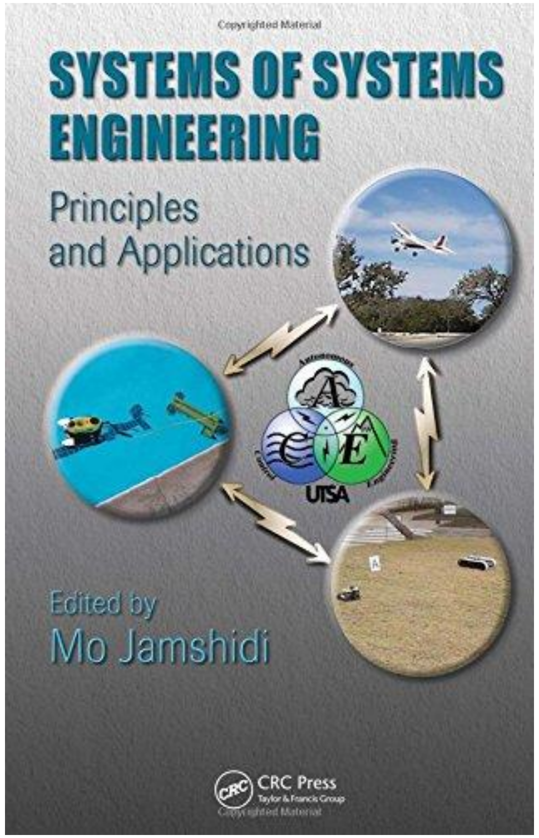
Agreed Collaboration: **SoS Goal**



https://www.pinterest.com

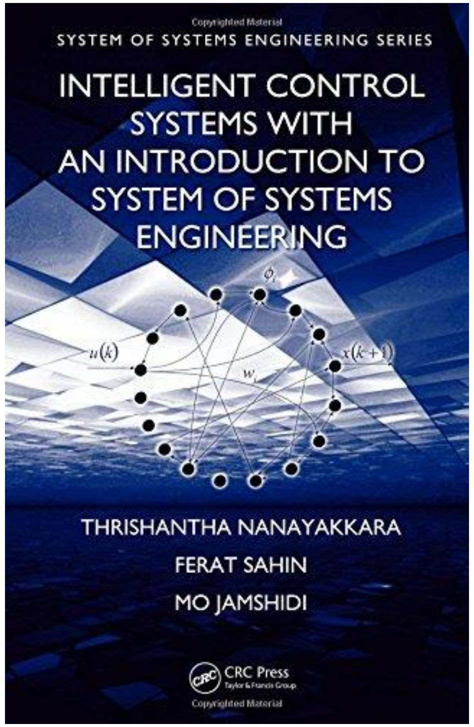

Functional Emergence
Constituent System
Constituent System
Constituent System
Contract
Contract
Contract
System-of-Systems
Contract
Constituent System
Contract
Contract
Contract
Information Emergence
Constituent System
Constituent System
Constituent System

Textbook



Mo Jamshidi:
Systems of Systems Engineering – Principles and Applications
CRC Press Inc., USA, 2008. ISBN 978-1-420-06588-6

Textbook



Thrishantha Nanayakkara, Ferat Sahin, Mo Jamshidi:
Intelligent Control Systems with an Introduction to Systems of Systems Engineering
CRC Press Inc., USA, 2009. ISBN 978-1-420-07924-1

Recommendations

Systems-of-Systems (SoS)

1. Develop and maintain a transparent, complete, up-to-date, well documented architecture for the SoS
2. Fully understand and (formally) specify all dependencies
3. Fully understand and (legally) specify the governance of the SoS
4. Define all dependencies by formal contracts
5. Use effective risk analysis and mitigation for the early detection of operational faults, errors or unwanted emergent behaviour
6. Implement monitoring capabilities to detect unwanted emergent behaviour

Justification: Due to the fragmented governance/ownership in a system-of-systems, the management, evolution and operation of a SoS are more demanding. Therefore new procedures, engineering processes and operational measures must be used.

Cyber-Physical Systems-of-Systems (CPSoS)

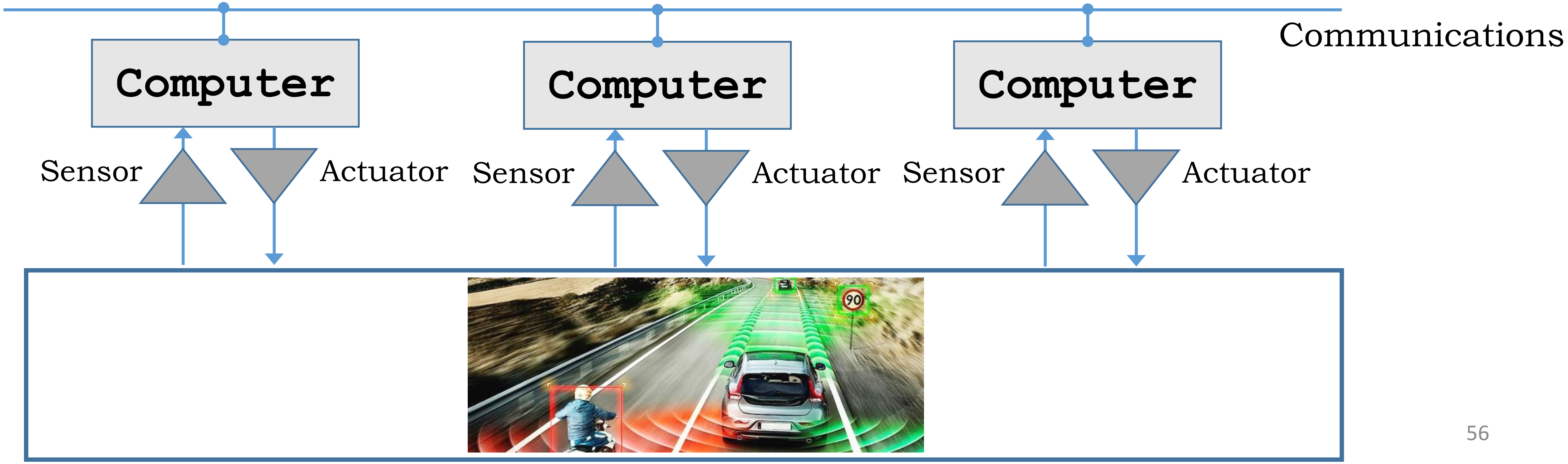
Cyber-Physical System-of-Systems (SoS)

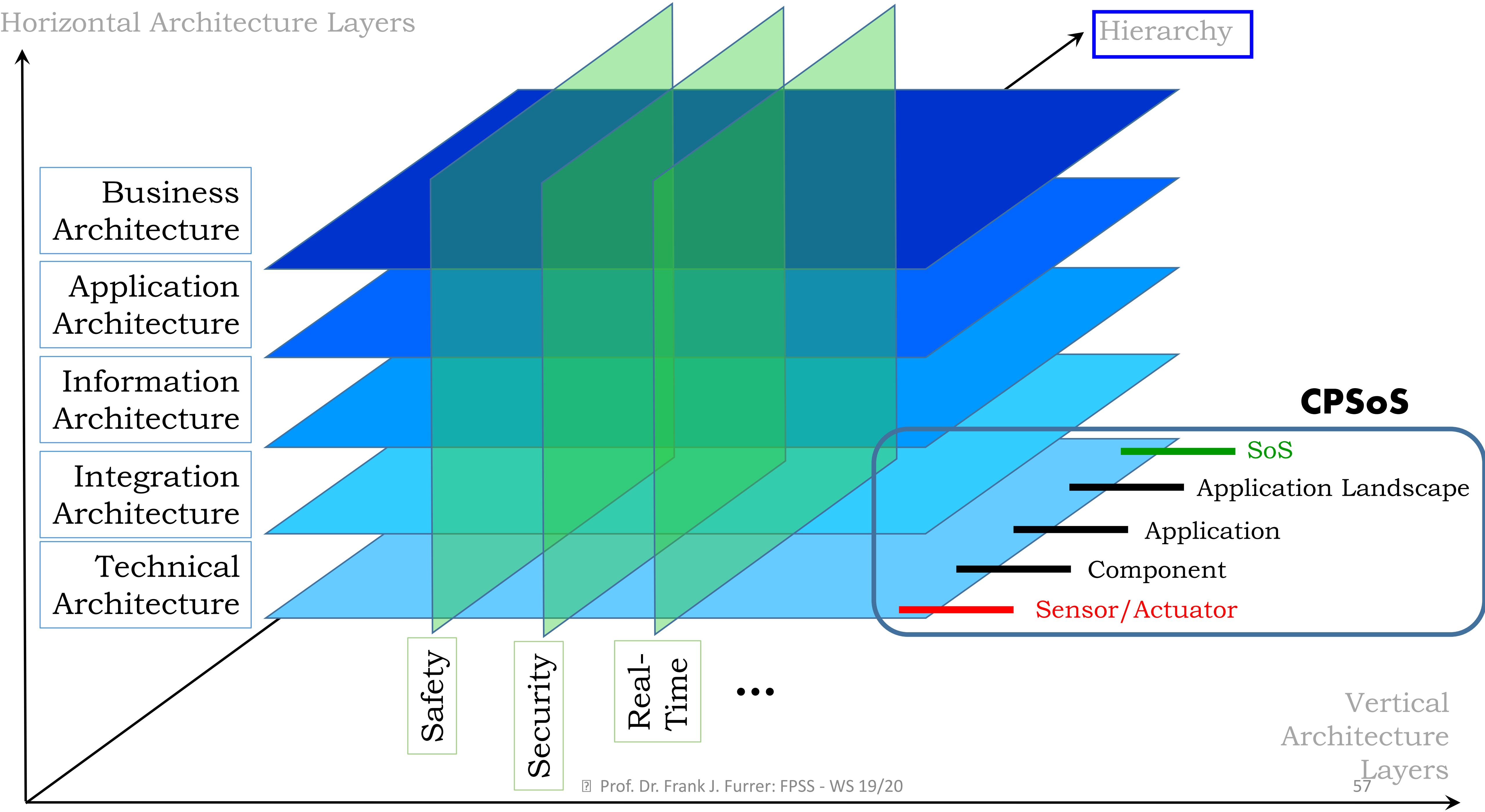
DEFINITIONS

A **cyber-physical system of systems (CPSoS)** brings together a set of *cyber-physical ooperating systems* for a task that none of the systems can accomplish on its own (= emergent property).

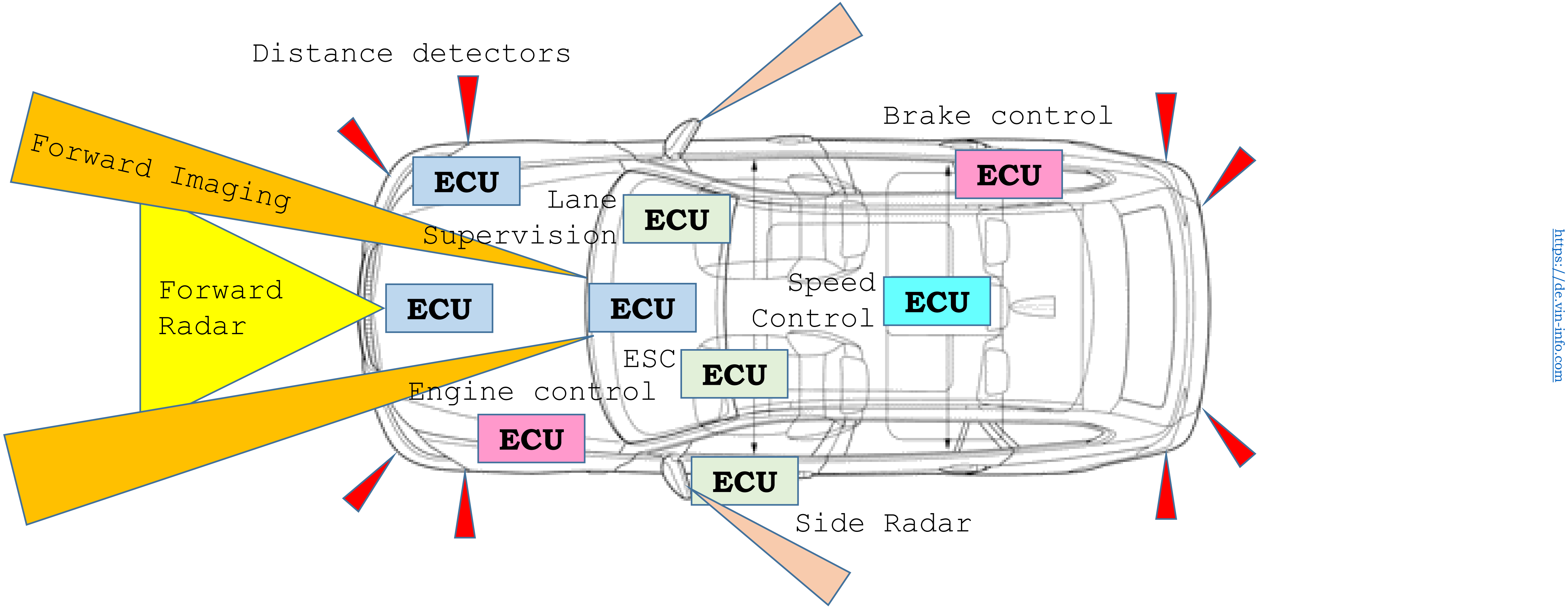
Each constituent cyber-physical system keeps its own management, goals, and resources while coordinating within the CPSoS and adapting to meet CPSoS goals.

Adapted from ISO/IEC/IEEE 15288 Annex G




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

Example: A modern car as CPSoS



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

CPSoS Example: Roboracе [Unmanned Automobile Racing]

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

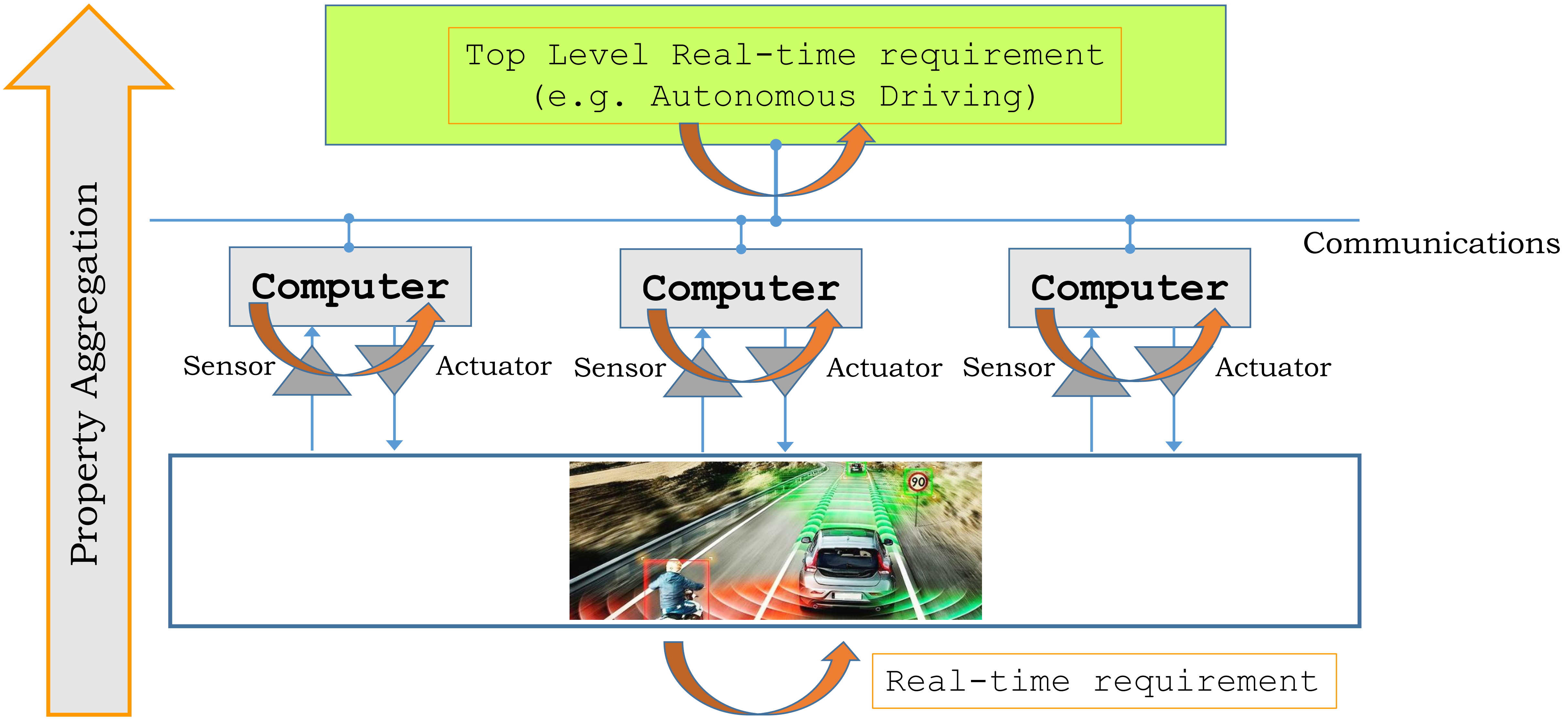
https://www.crowdact.com

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

https://wi-images.condecdn.net

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

NO drivers

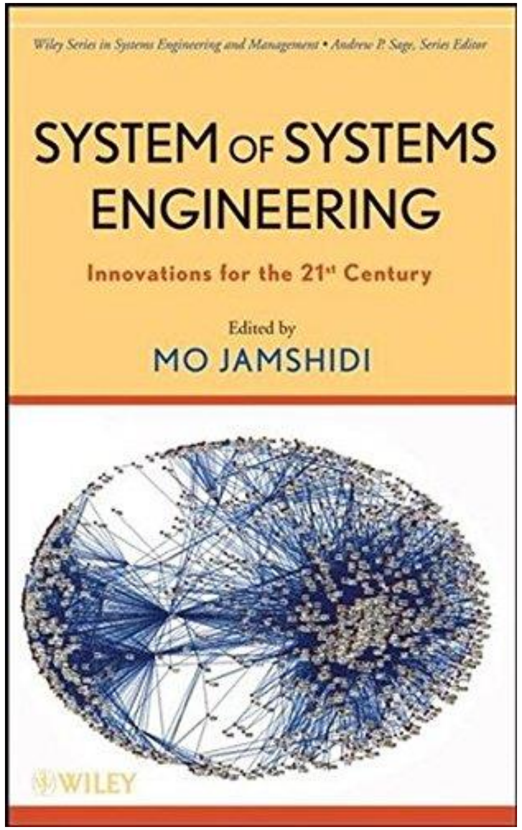

Top Level Real-time requirement
(e.g. Autonomous Driving)
Communications
Computer
Computer
Computer
Sensor
Actuator
Sensor
Actuator
Sensor
Actuator
90
Real-time requirement
Property Aggregation

CPSoS Example: Autonomous earthquake search & rescue robots

https://spectrum.ieee.org

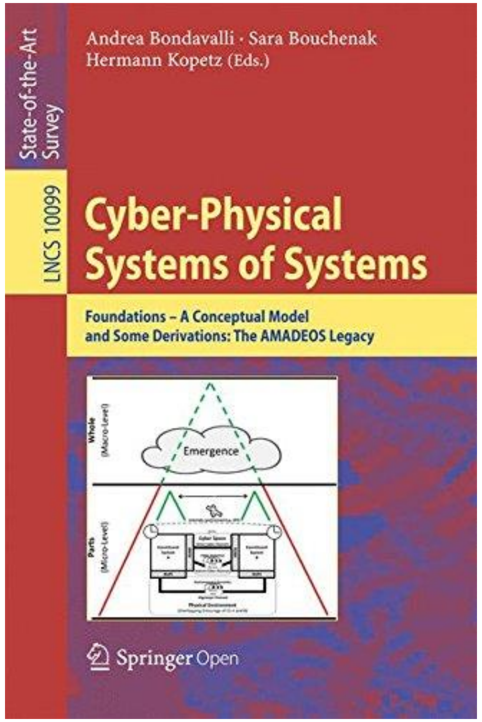
http://www.allonrobots.com

Textbook



Mo Jamshidi (Editor):
Systems of Systems Engineering – *Innovations for the 21st Century*
John Wiley & Sons Inc., Hoboken, New Jersey, USA, 2009. ISBN 978-0-470-19590-1

Textbook



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

Recommendations

Cyber-Physical Systems-of-Systems (CPSoS)

1. Dependability properties (safety, security, ...) always superse functionality requirements – in specifications, architecture, design, implementation, and operation
2. Implement monitoring capabilities – especially on the *interfaces* – to predict or detect abnormal or dangerous behaviour

Cloud Computing

Cloud computing …

- is a ***model*** for enabling ubiquitous, convenient, on-demand access:
- to a shared pool of configurable ***computing resources*** (e.g., networks, servers, storage, applications, and services)
- that can be rapidly provisioned and released
- with minimal management effort or service provider interaction

US National Institute of Standards and Technology
http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

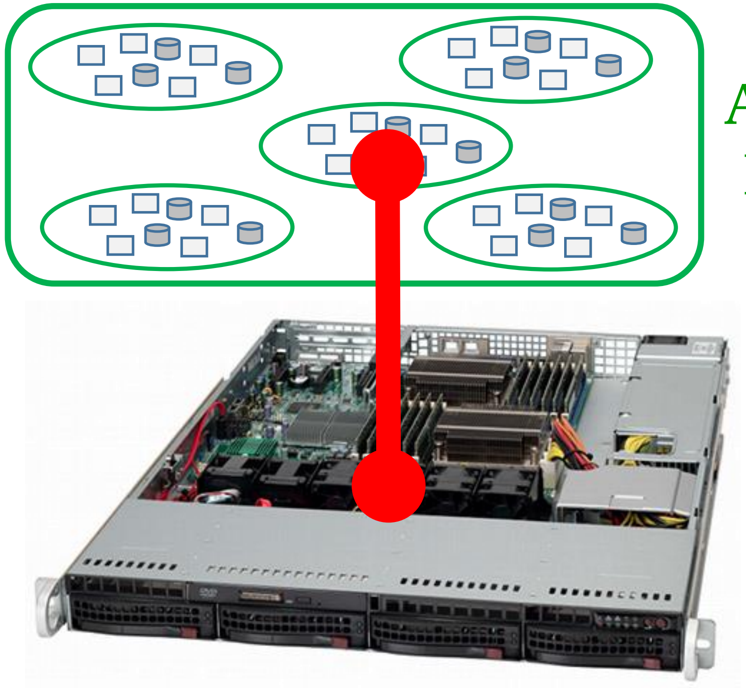
http://www.telegraph21.com

Iron Age

http://www.freiburg.de

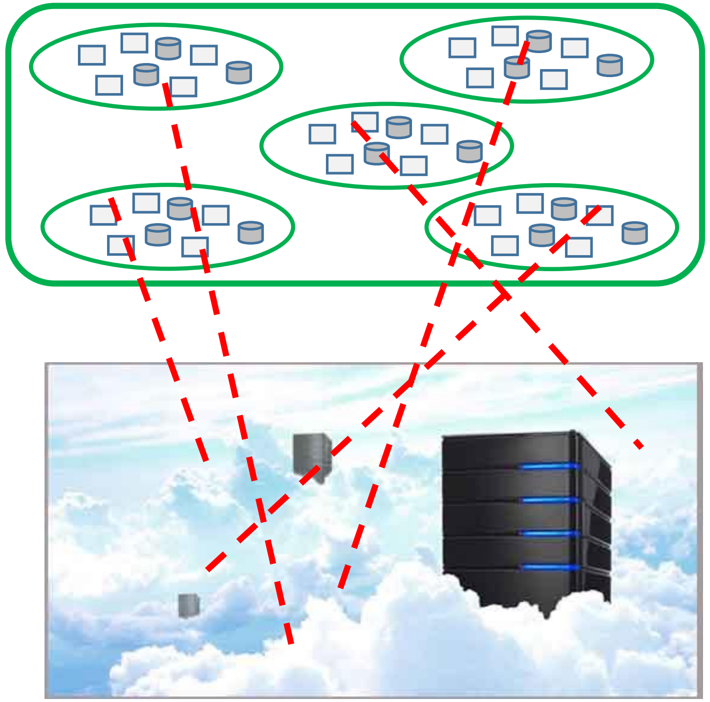
Cloud Age

http://blog.schneider-electric.com



Application Landscape

https://www.snelserver.com



http://www.telegraph21.com

Cloud-based computing

Enterprise Cloud Computing

http://www.telegraph.co.uk

Embedded Cloud Computing

Enterprise Cloud Computing

Enterprise Cloud Computing is a ***business model*** with promising commercial and technical advantages. It relies on using 3rd party IT-services delivered via the Internet instead of in-house IT-installations.

⇒ Enterprise IT **deployment-architecture** for the next decades

- On-demand use of IT-capabilities
- Massive reduction of IT capital investment
- Significant reduction of (low level) IT staff
- Access to modern technology and services

Advantages of Cloud Computing

http://diginetinfosystems.com

Risks of Cloud Computing

http://www.ultimatetech.org

Security

http://ex-c.co.uk

Loss of Control

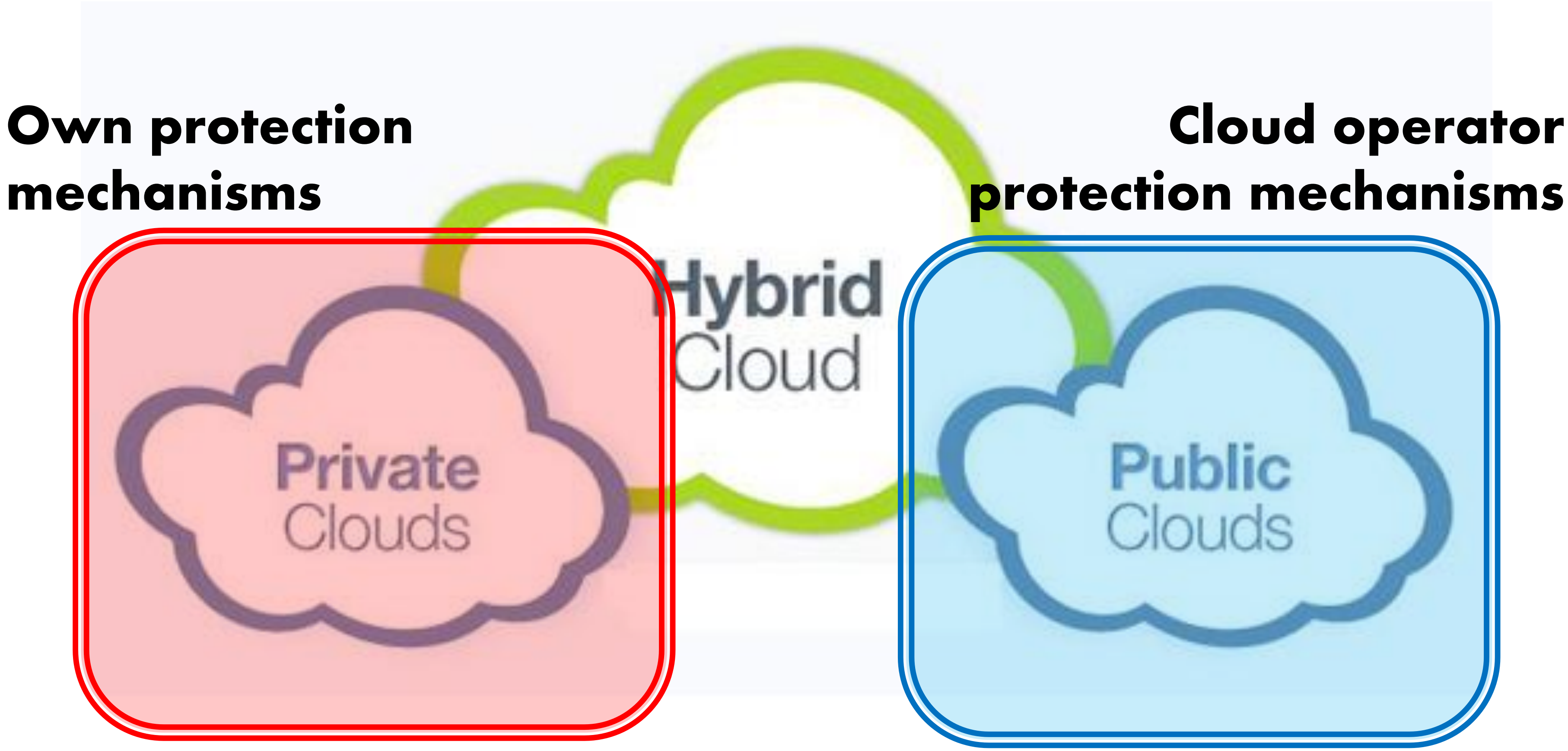
Loss/Theft of Company Data

http://www.iperiusbackup.net

http://lionfield.com

Disaster Recovery

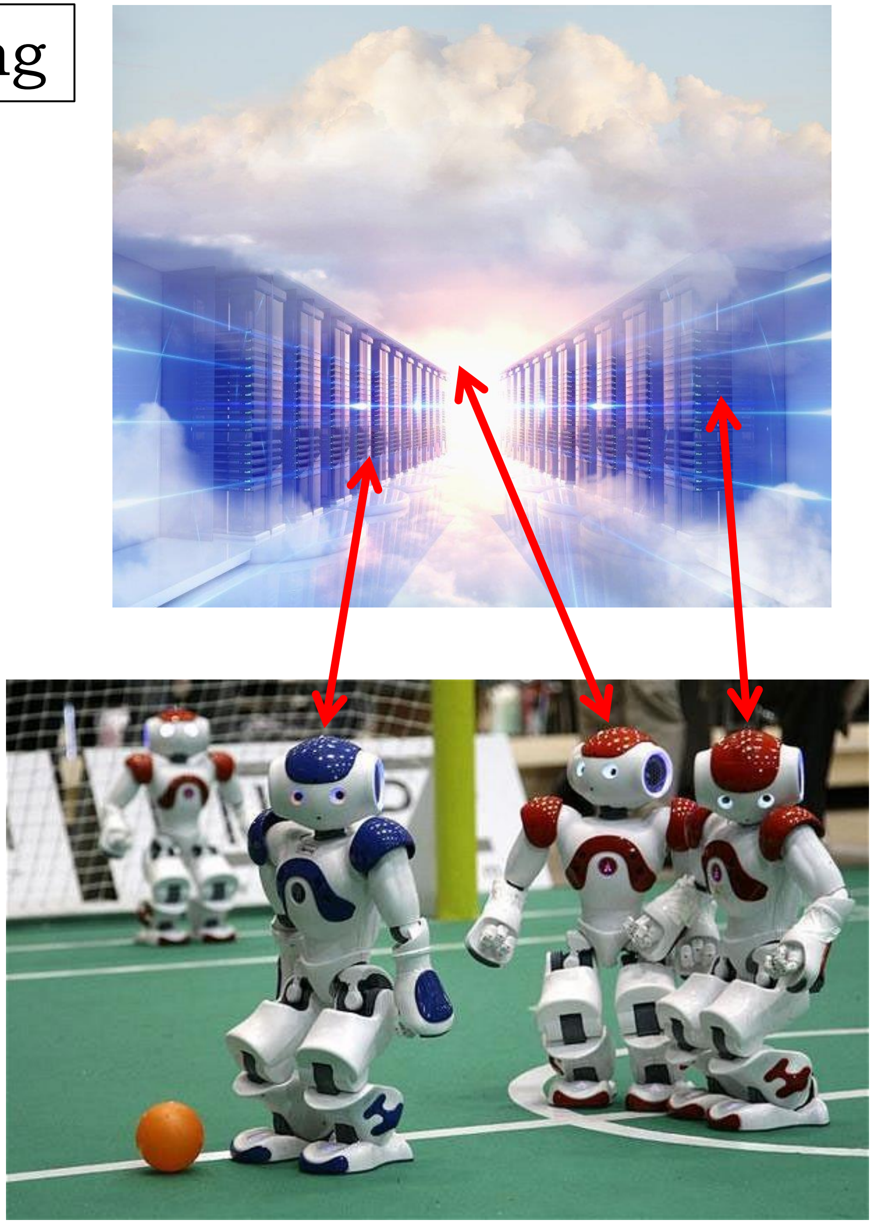
Risks of Cloud Computing



http://www.itbriefcase.net

Embedded Cloud Computing

http://www.manager-magazin.de



Distributed systems *communicate* and *synchronize* in real-time via the cloud

Distributed systems use the power of *cloud resources* http://www.telegraph.co.uk

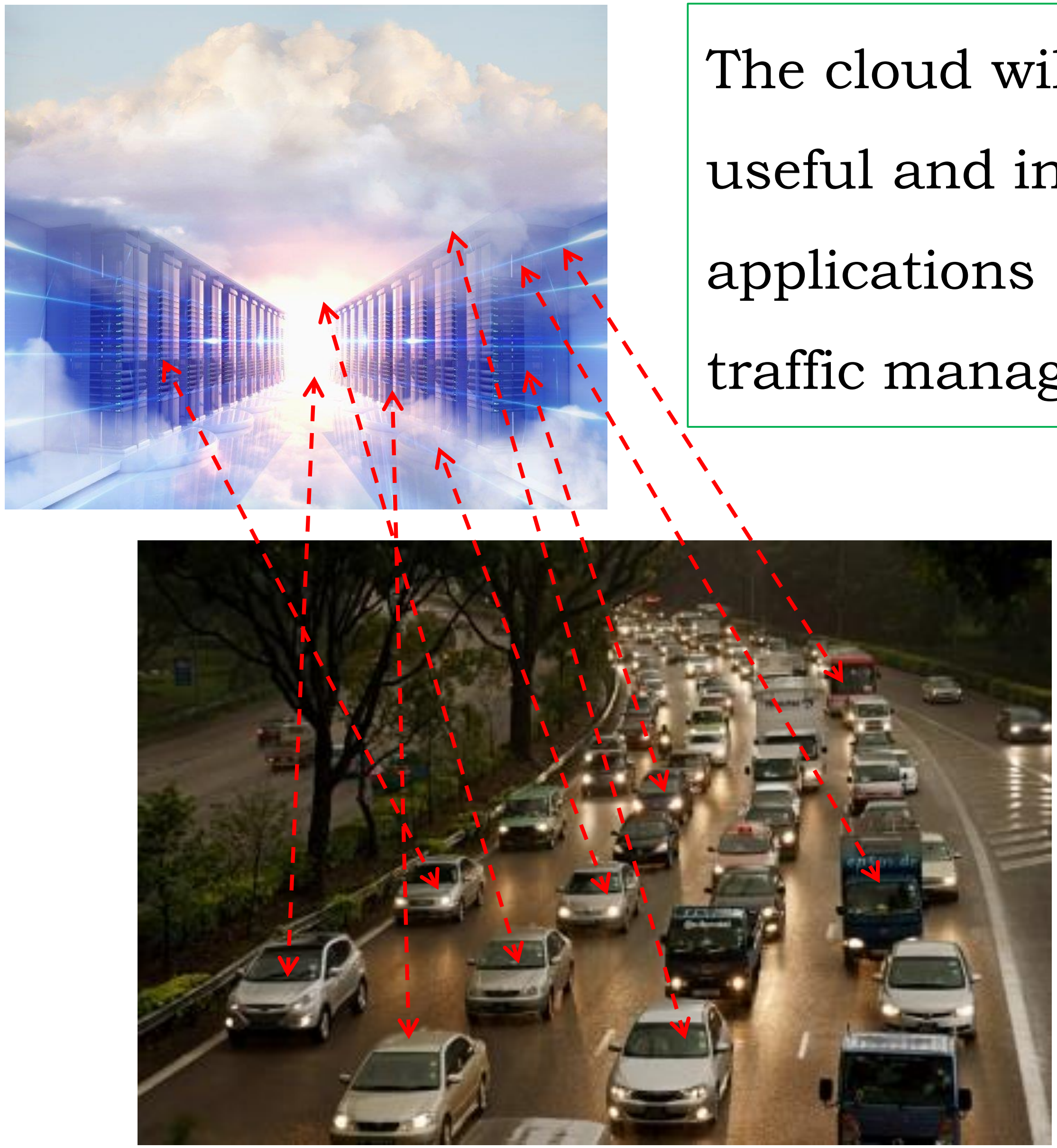
Embedded Cloud Computing

Cloud Robotics is the application of the cloud computing concept to **robots**. This means using the Internet to augment the robots capabilities by off-loading computation and providing services on demand

⇒ Interesting and active research area:

- Cloud Robotics
- Collaborative Manufacturing
- Intelligent Traffic Management
- etc.

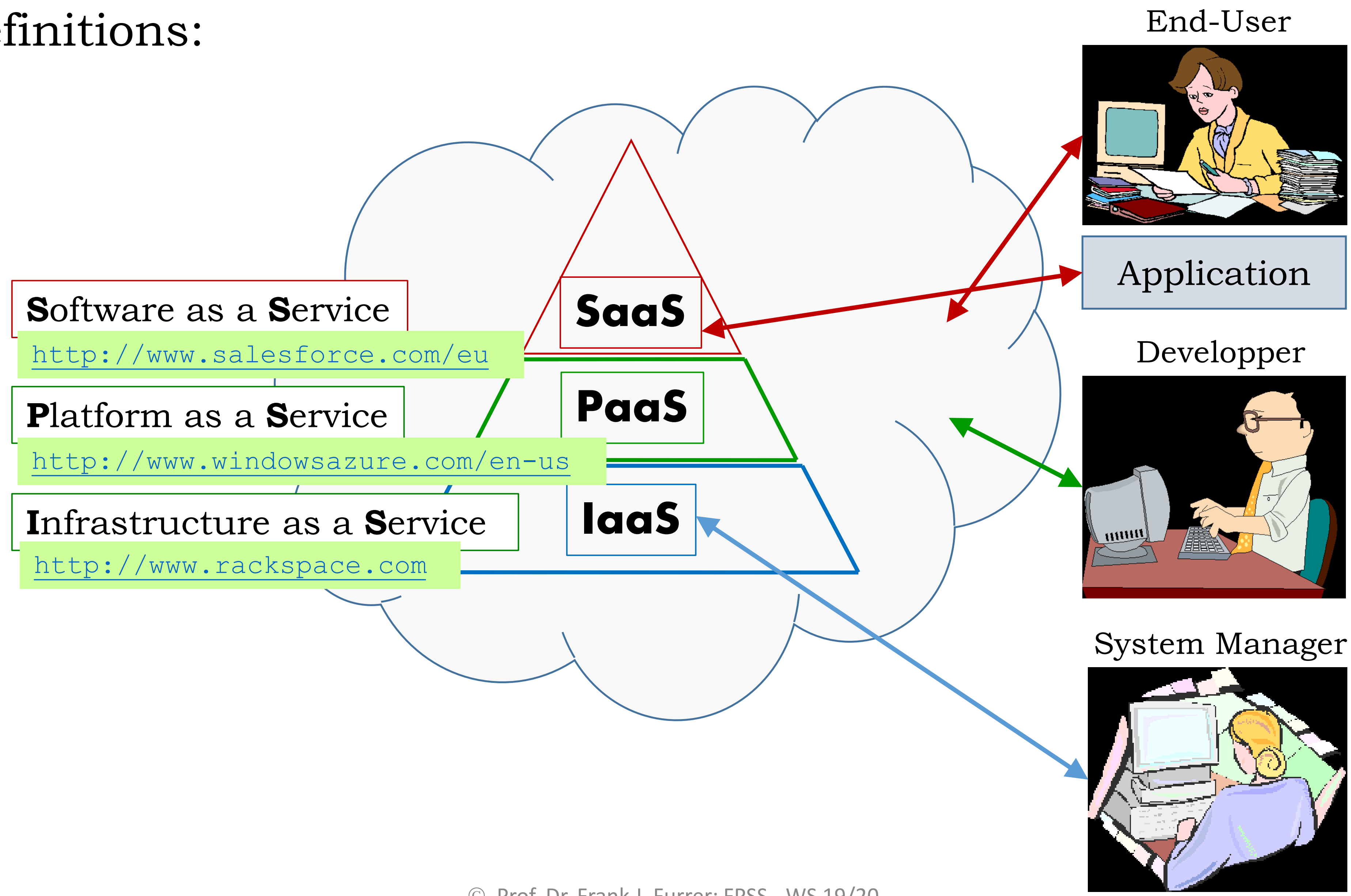
http://www.manager-magazin.de



The cloud will enable many useful and interesting applications (e.g. optimized traffic management)

https://gigaom.com

Cloud Definitions:



Cloud Definitions:

Cloud Service **Provider**

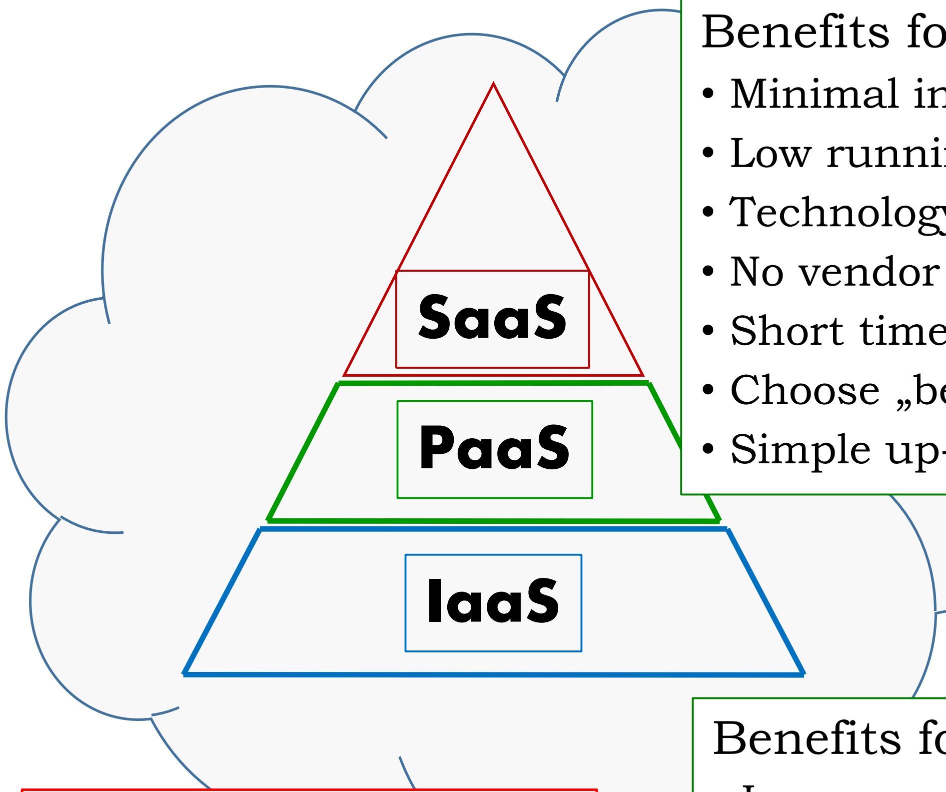
Contract

Cloud Service **Consumer**

http://1.bp.blogspot.com

http://www.vision-training.com

Cloud **Benefits**:



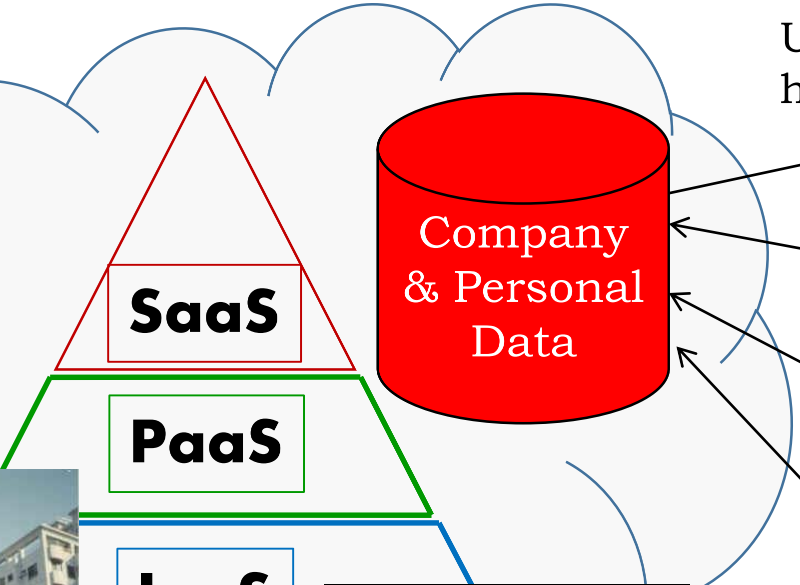
Benefits for the **Cloud Service Consumers**:
- Minimal investment
- Low running/operational cost
- Technology-independent
- No vendor lock-in
- Short time-to-market („assemble“ instead of „build“)
- Choose „best of breed“
- Simple up-/downscaling

Cloud = Execution and Delivery Structure

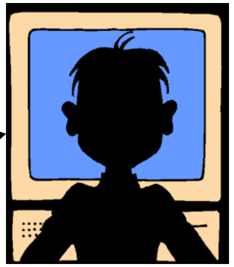
Cloud Usage = Commercial decision (Business model)

Benefits for the **Cloud Service Providers**:
- Large market
- No copying, no IPR disclosure
- Economy of scale
- Interesting new business models
- Specialization

Cloud **Risks**:



Unauthorized access, hacking



Privacy Data Loss

Availability

Industrial Espionage

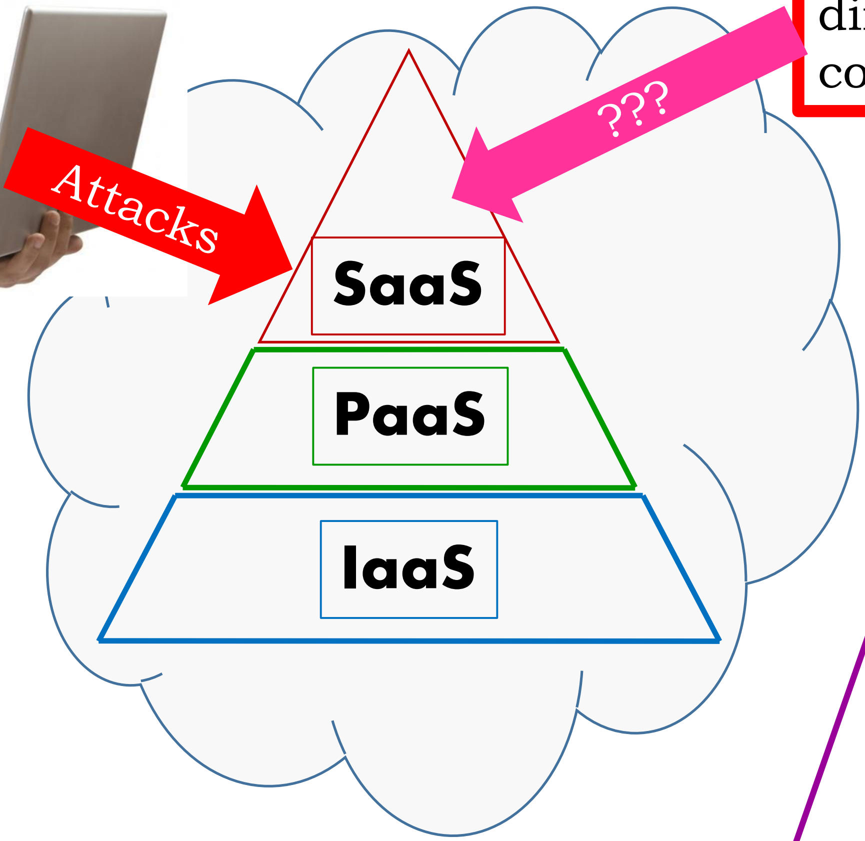
Large Scale Disaster

Local Disaster

Cloud **Risks**:


http://pssiusa.wordpress.com

Security,
Dependability,
Resilience,
…

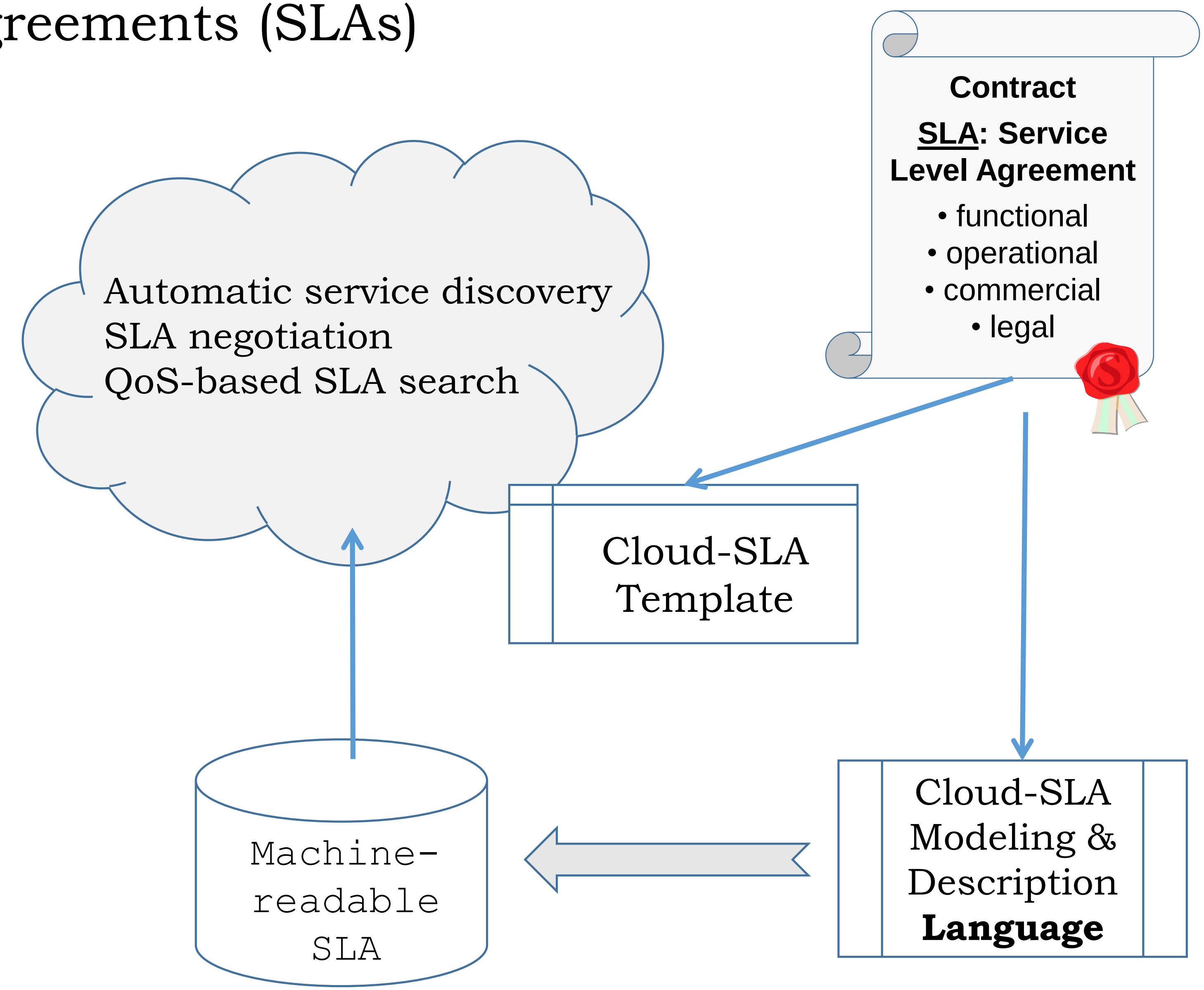


Loss of direct control

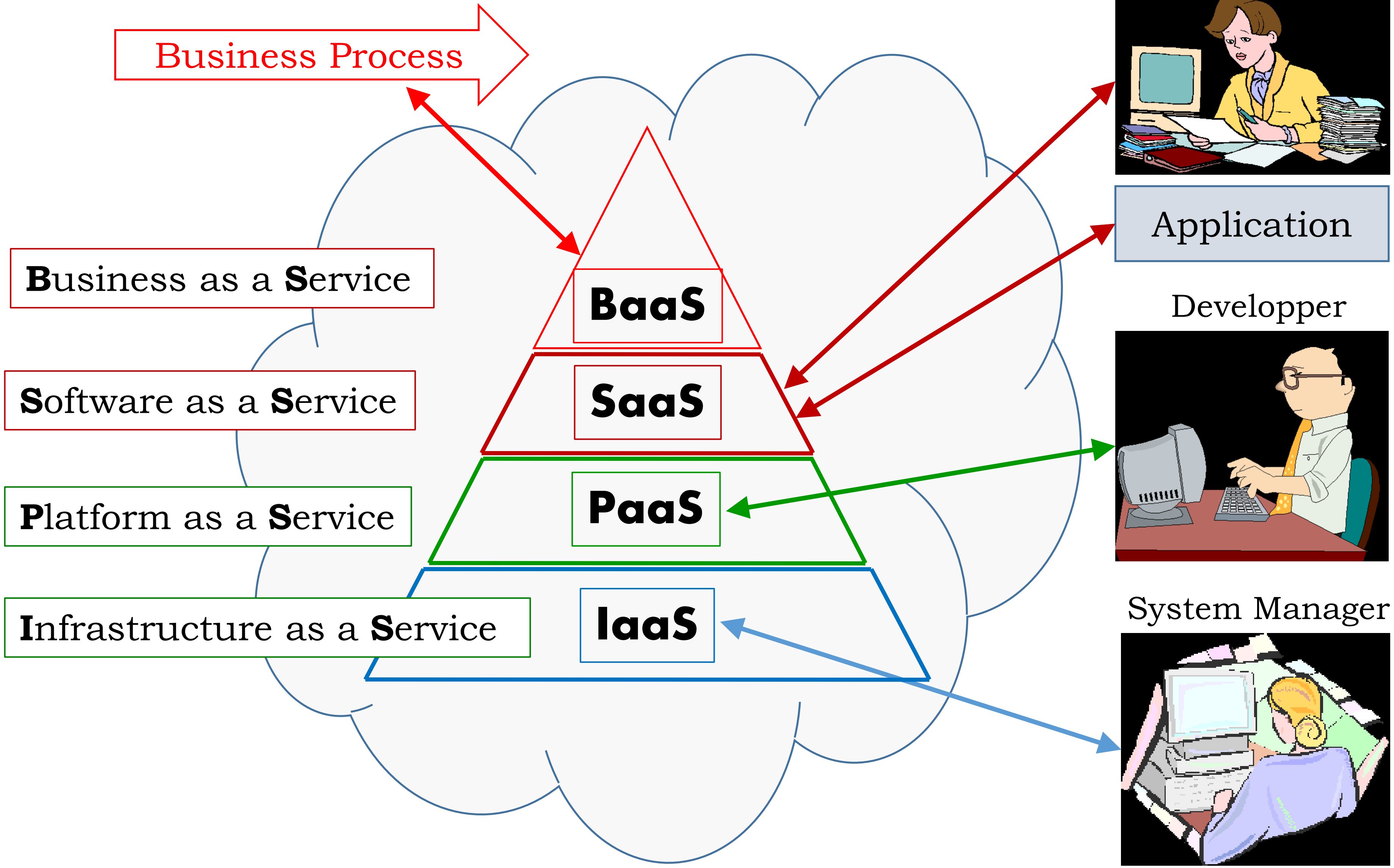
compensate by:

Cloud Service Level Agreements (Cloud-SLA)

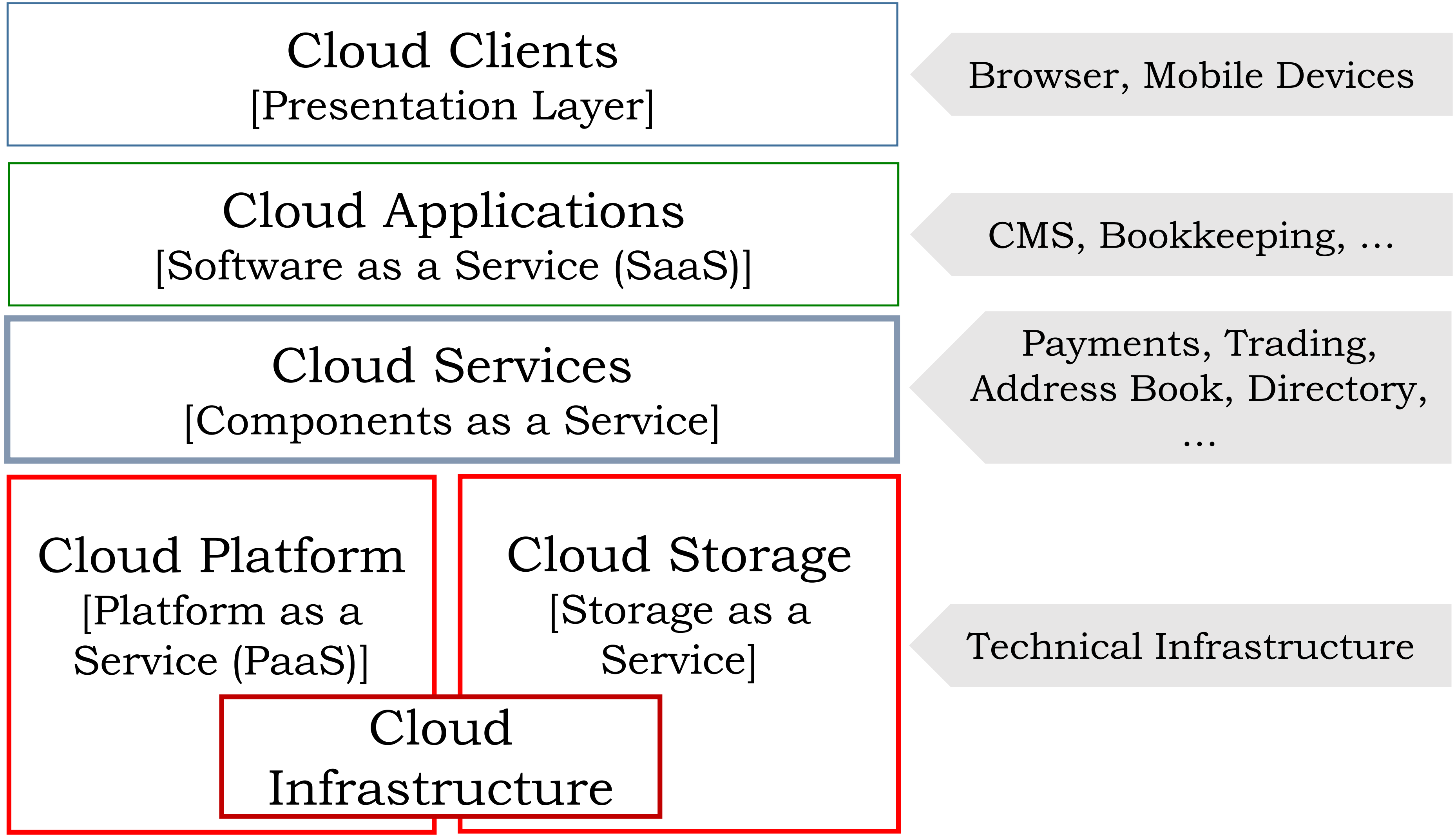
Cloud Service Level Agreements (SLAs)



Cloud Definitions:



Cloud Computing Stack:



Mahmood, 2011

Cloud Responsibilities

http://kriszha.com

Provider

- Service **architecture**
- Industry standards
- Effective & efficient
- Agile
- Security
- Disaster recovery
- Cost models

Responsible Cloud Infrastructure

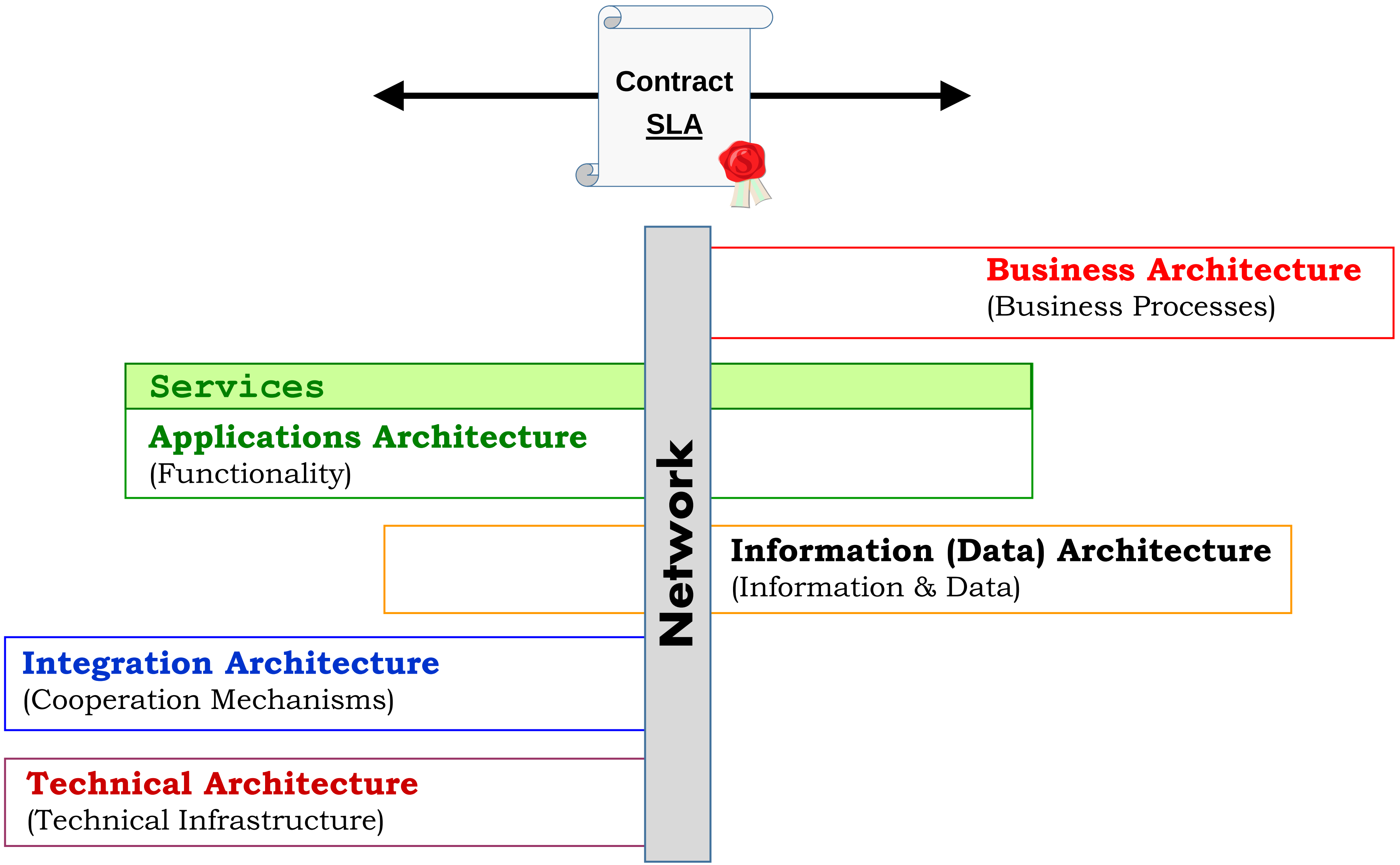
http://www.gograph.com

Consumer

- Business model
- Enterprise cloud **architecture**
- Risk analysis
- Contracts
- Privacy ?
- Business Continuity Planning
- Monitoring

Enterprise Cloud Service Architecture

Cloud: Influence on Architecture



The cloud paradigm offers very interesting commercial and technical ***benefits***

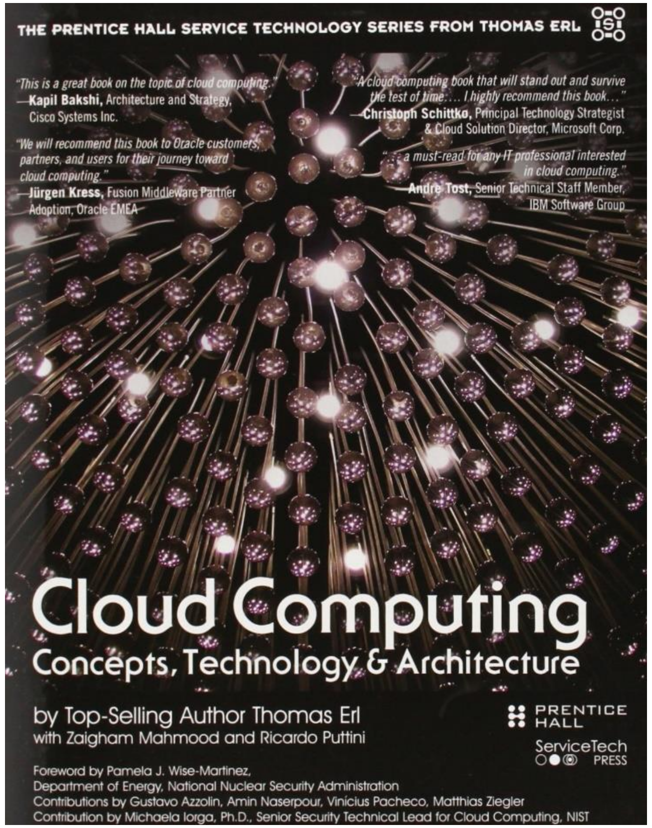
When using IaaS, PaaS, SaaS or BaaS („cloudification“) a ***loss*** of direct control results

Many of the quality properties (Security, availability, privacy etc.) are mainly under the control of the cloud service provider

The loss of direct control must be compensated by clear, explicit, and legally binding ***contracts***

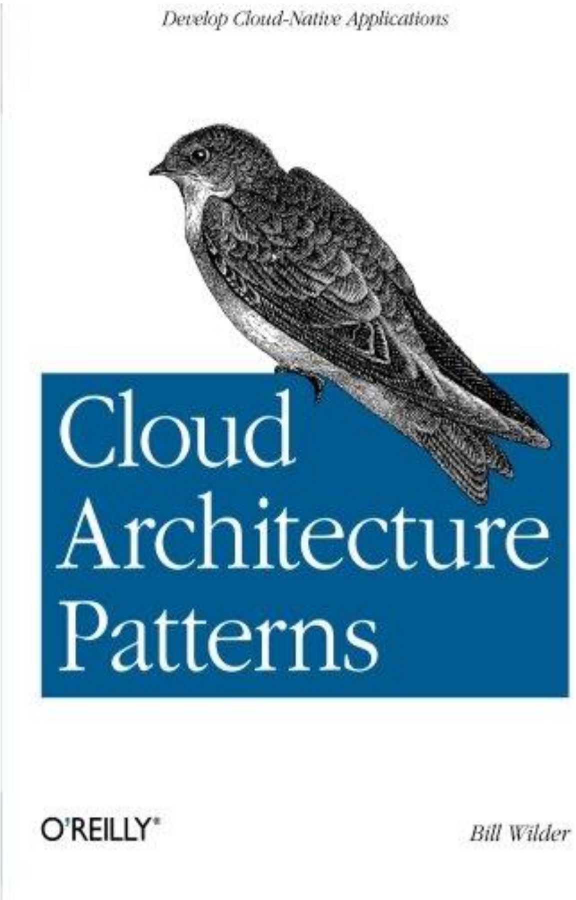
Focus shifts from „building“ to „assembling“ – and needs new, different engineering + architecture processes

Textbook



Thomas Erl, Ricardo Puttini, Zaigham Mahmood:
Cloud Computing – *Concepts, Technology & Architecture*
Prentice Hall Inc., USA, 2013. ISBN 978-0-133-38752-0

Textbook



Bill Wilder:
Cloud Architecture Patterns
O'Reilly and Associates, USA, 2012. ISBN 978-1-449-31977-9

Recommendations

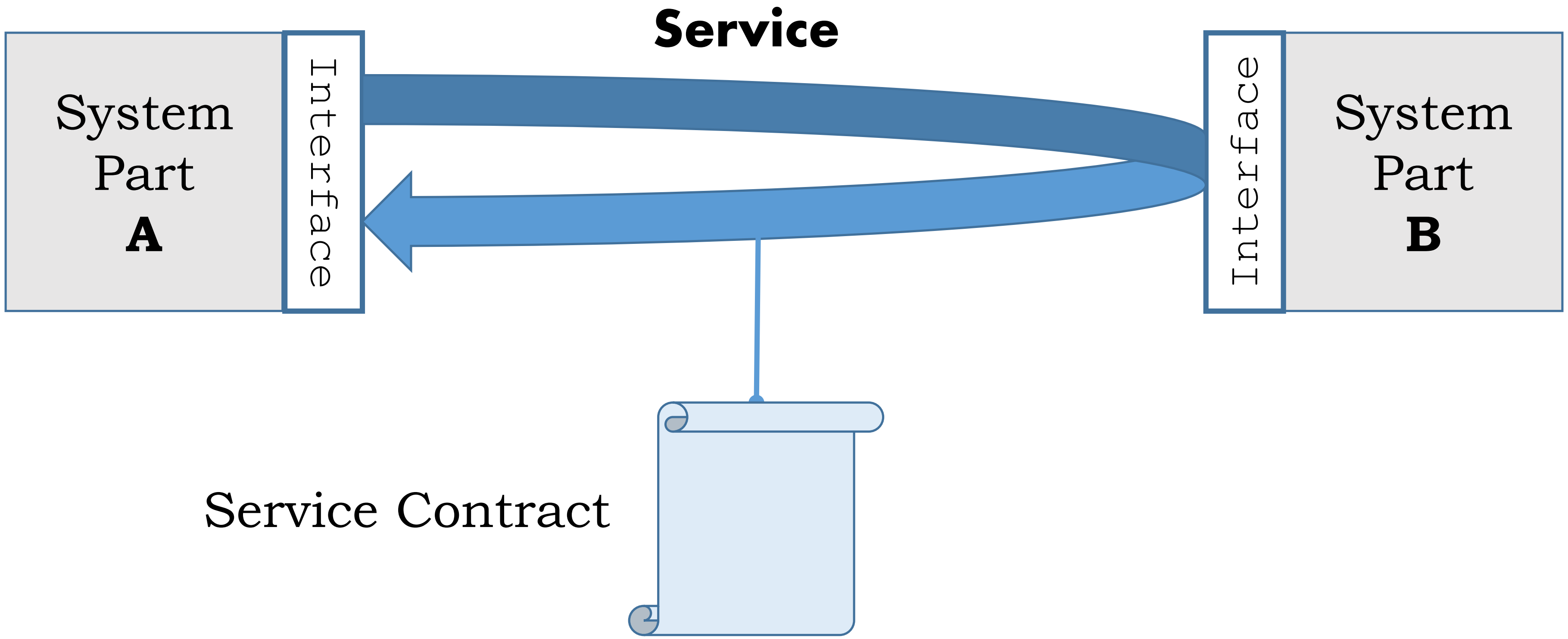
Architecture Recommendation for Cloud Service *Providers*:

1. Implement the architecture principles as presented in this lecture
2. Deliver the cloud-services via established, accepted industry-standards
3. Provide transparency on your architecture, implementation and evolution
4. Give factual & contractual assurance for the quality properties (dependability, availability, privacy, disaster-recovery, performance etc.)

Architecture Recommendation for Cloud Service *Consumers*:

1. Compensate the loss of transparence by requiring sufficient information about cloud service provider architecture, quality properties etc.
2. Compensate the lack of control by clear, explicit, legally binding Cloud-SLAs (Cloud service level agreements)
3. Insist on established, accepted industry-standards for the delivery of all cloud-services

Microservices



- Functionality
- Operational parameters
- Commercial Conditions

Services implement:

- Good partitioning
- Excellent encapsulation
- Loose coupling

Coarse-grained service

Granularity

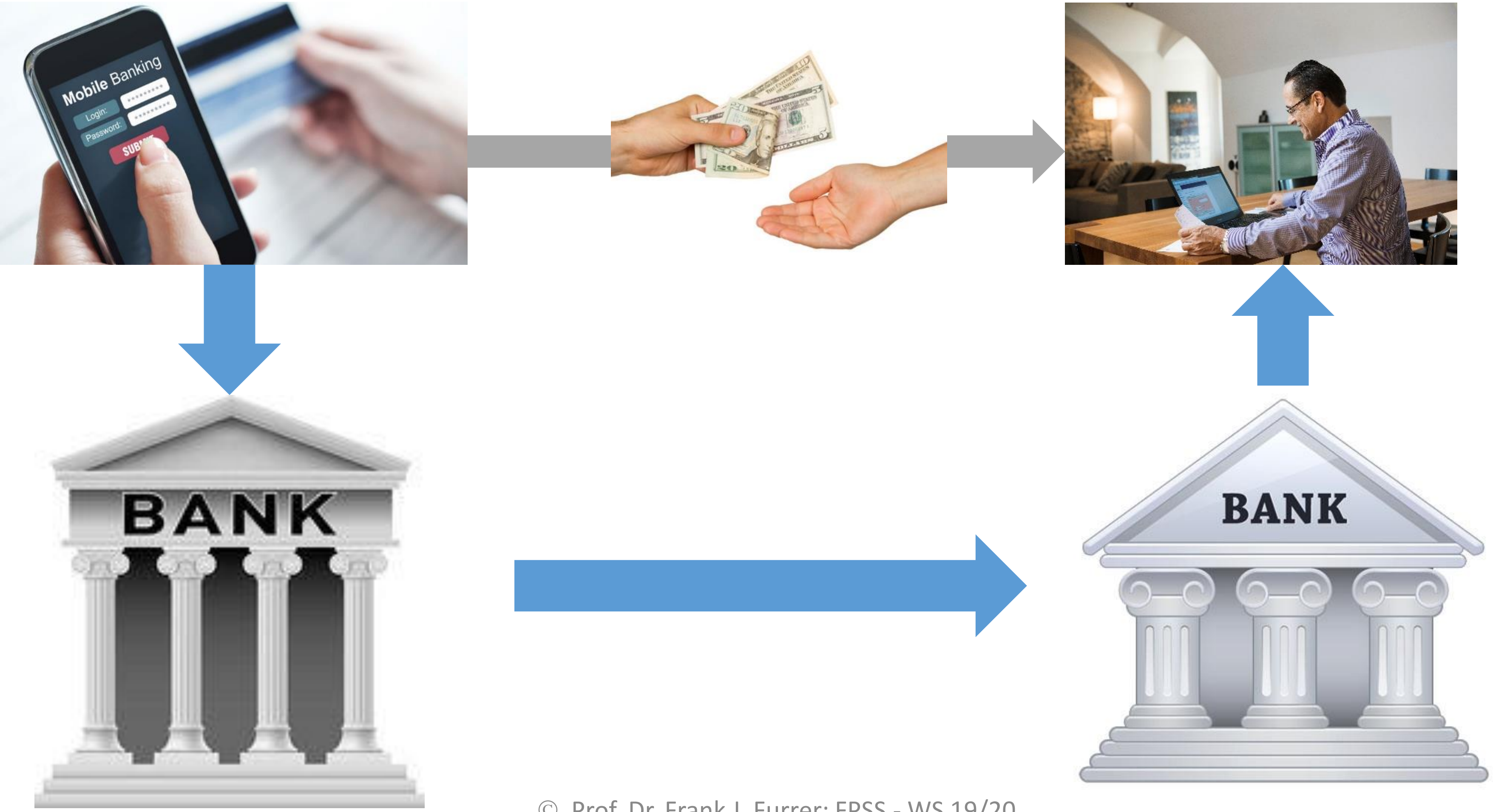
Fine-grained service

http://lordsandknights.enjoyed.today/common/fansite_kit/units/OxCart.png

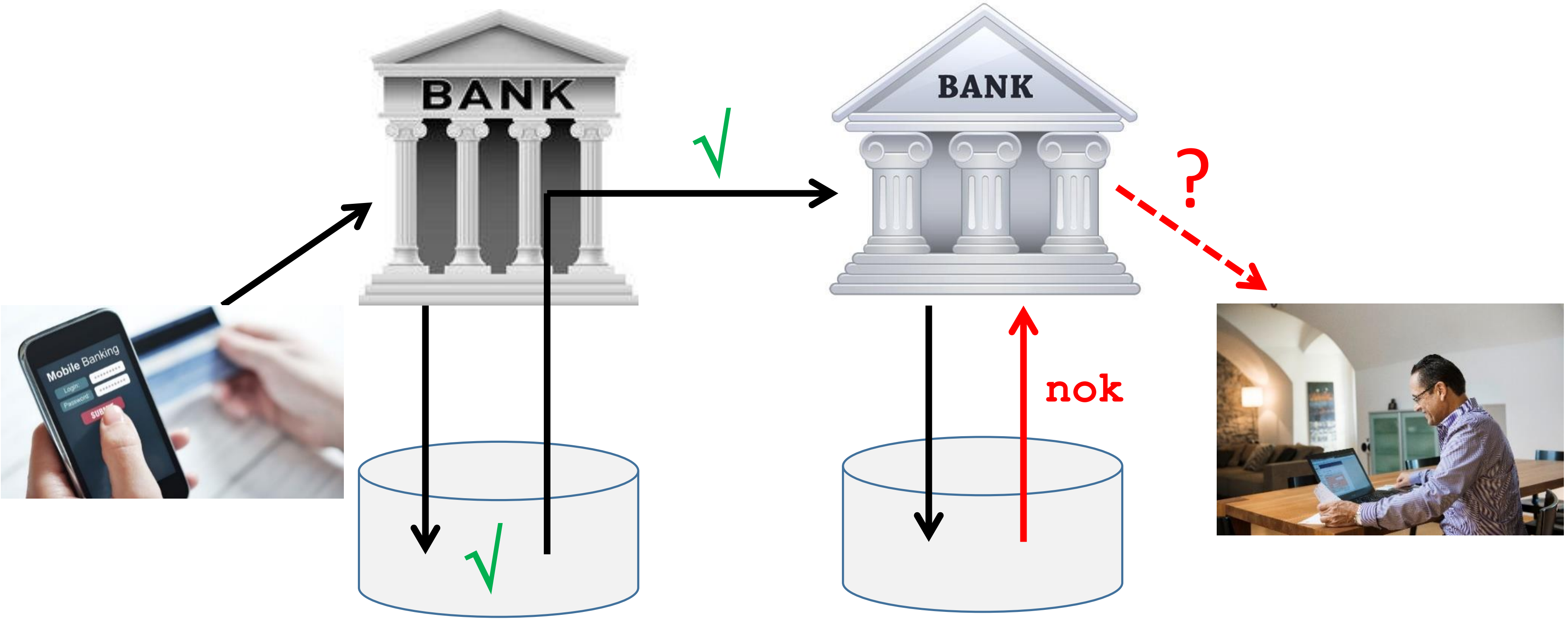
http://www.prodogroomingsupplies.com

Example: Coarse-grained service (1/4)

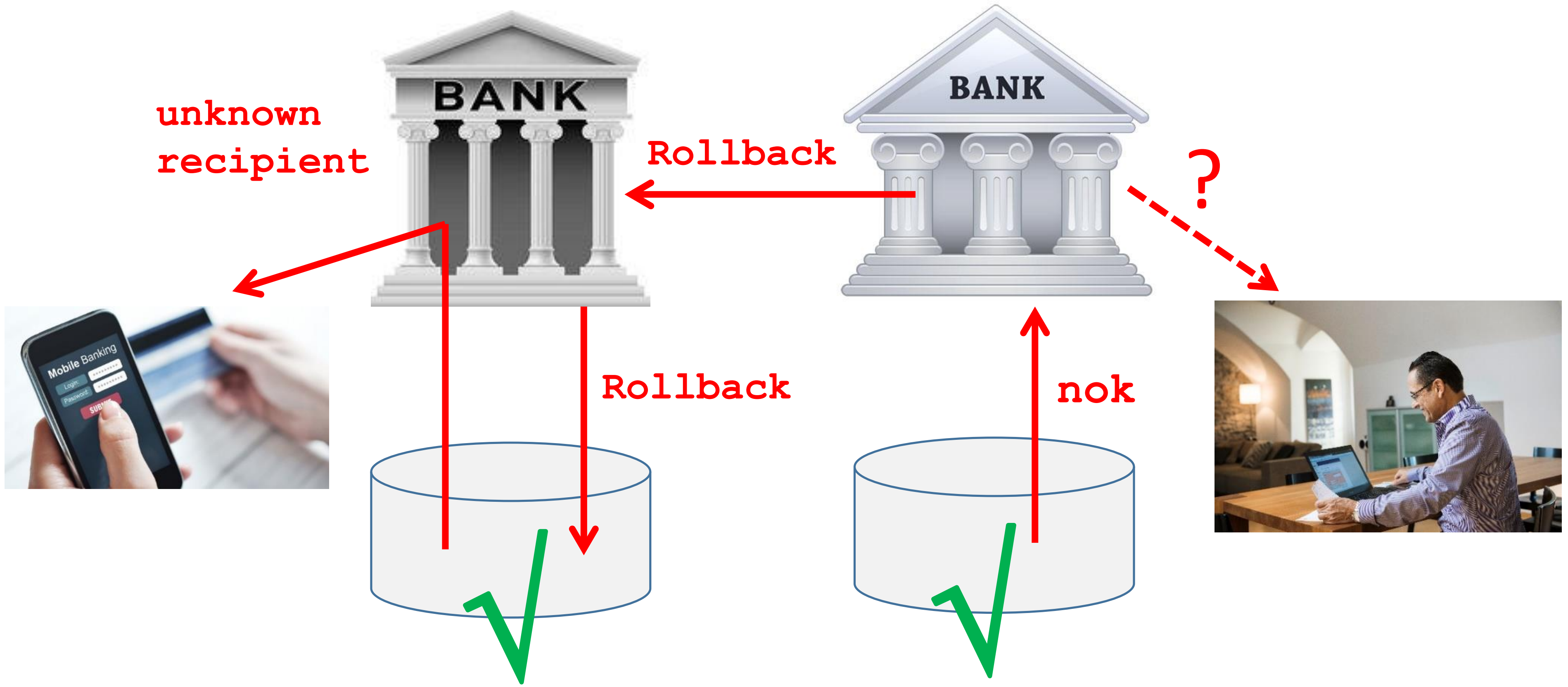
Interbank Electronic Money-Transfer:



Example: Coarse-grained service (2/4)

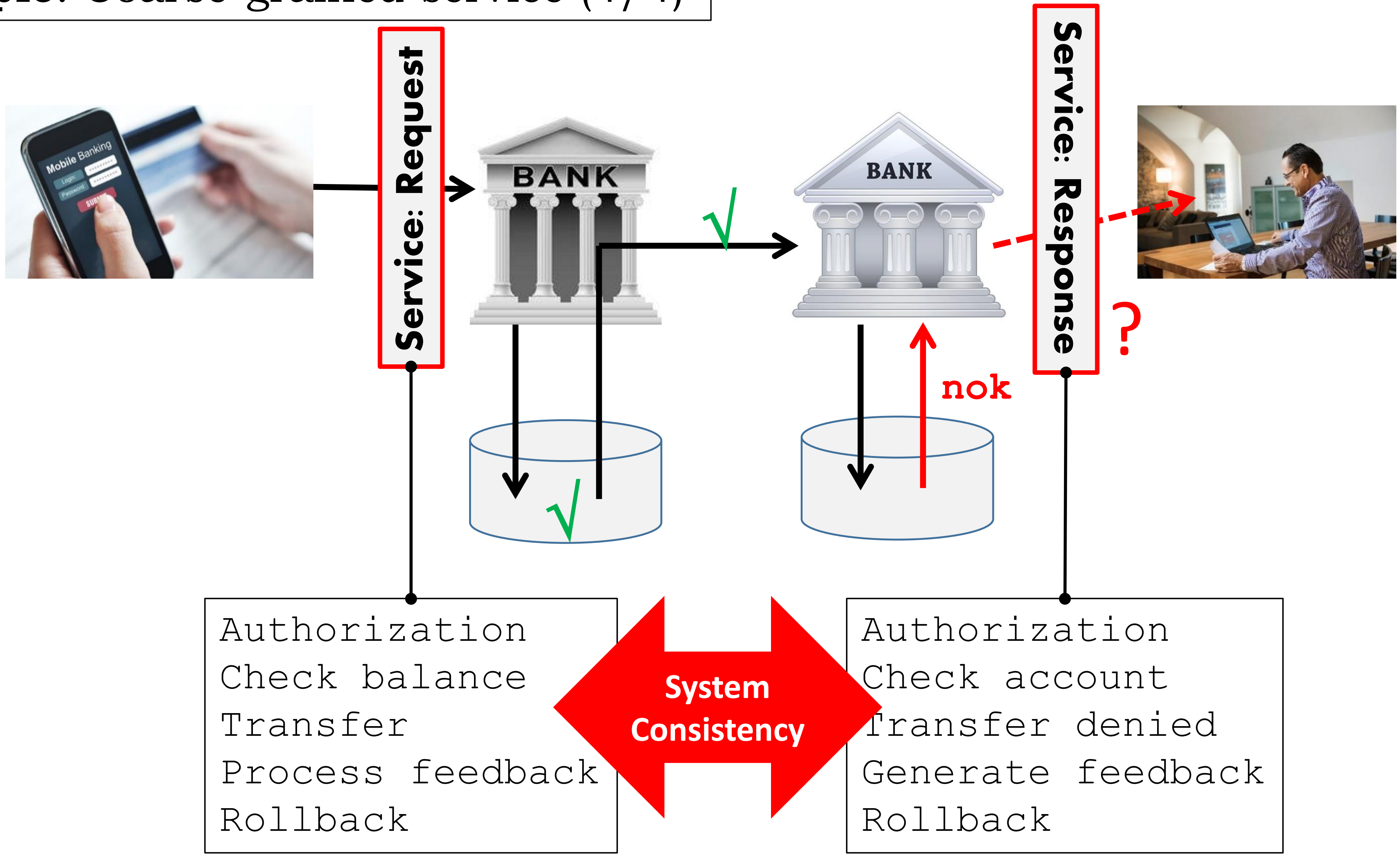


Example: Coarse-grained service (3/4)



All databases are correct and consistent

Example: Coarse-grained service (4/4)



Example: Fine-grained service (1/x)

Client-ID
X3298a289

Client-Data
Max Muster
Beispielstr. 5
Neustadt

Service

DB search
DB access
- **Nothing more**

Coarse-grained service

Granularity

Fine-grained service

Service Usage

Service

Functionality
- Computation
- Data access
- Data transfer
- Control transfer

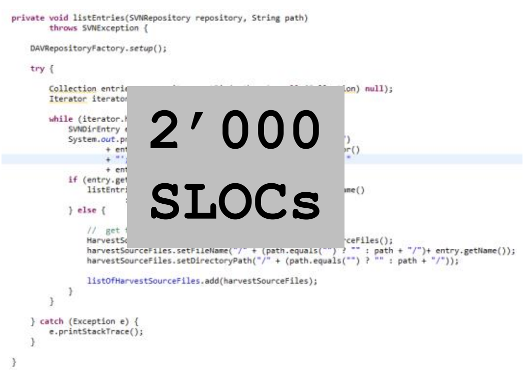
Non-functional properties
- Error handling
- Fault processing
- Security
- Safety, …
- Parallelism
- Scalability
- …

System-Properties
- Synchronization
- Consistency
- Monitoring
- Auditability
- Accountability
- Versioning
- Recovery
- …

Service

Functionality

- Computation
- Data access
- Data transfer
- Control transfer

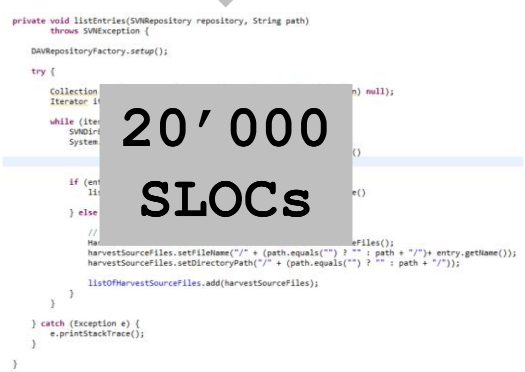


Non-functional properties

- Error handling
- Fault processing
- Security
- Safety, …
- Parallelism
- Scalability
- …

System-Properties

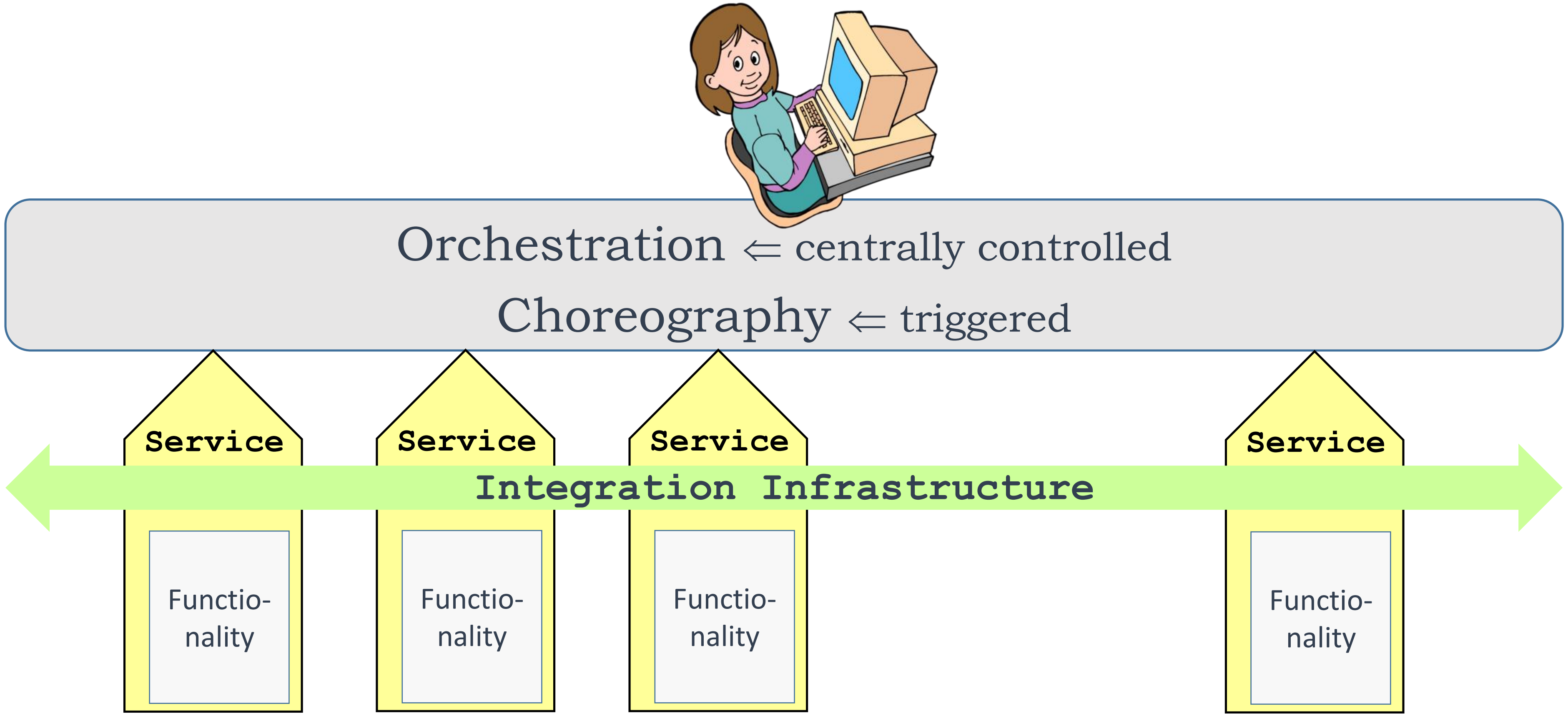
- Synchronization
- Consistency
- Monitoring
- Auditability
- Accountability
- Versioning
- Recovery
- …



DEFINITIONS

Microservice Principles

Microservices are ***small***, ***autonomous***, ***independently deployable*** services that work together using an ***integration infrastructure***



Specifi-
cation

Development
Evolution

Operation (24/7)

Micro-
service

Micro-
service

Micro-
service

Micro-
service

Teams

Management

http://publishinginsider.net

Microservice Principles

- A microservice is «***small***» (various, fuzzy definitions)
- A microservice does ***one*** thing – and does it well
- A microservice works with ***business objects***
- A microservice respects all ***architecture principles***

  (especially partitioning, encapsulation, loose coupling and no reduncancy)
- A microservice is consistently ***embedded*** into the enterprise architecture ⇐ *Importance of overall architecture!*

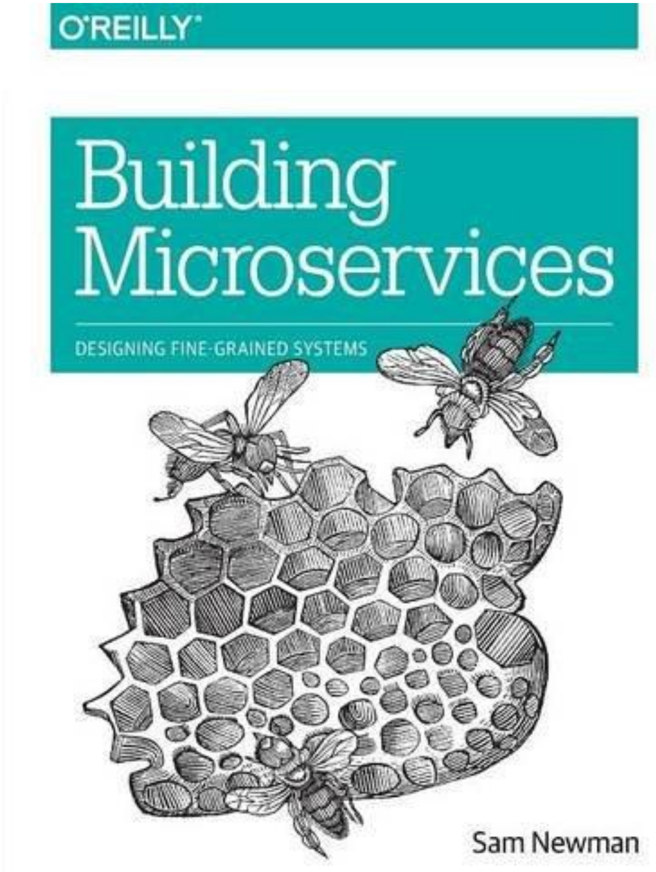
  (Structure, technology, security, safety, …)
- A microservice is independently ***evolvable*** and ***deployable***
- A microservice can be ***monitored*** as part of the whole
- A microservice has a defined ***owner***, embedded in the company governance structure

CAUTION: Adopting microservices requires an adequate company organization

- **Maximize team autonomy**: Create an environment where teams can get more done without having to coordinate with other teams.
- **Optimize for development speed**: Hardware is cheap, people are not. Empower teams to build powerful services easily and quickly.
- **Focus on automation**: People make mistakes. More systems to operate also means more things that can go wrong. Automate everything.
- **Provide flexibility without compromising consistency**: Give teams the freedom to do what's right for their services, but have a set of standardized building blocks to keep things sane in the long run.
- **Built for resilience**: Systems can fail for a number of reasons. A distributed system introduces a whole set of new failure scenarios. Ensure measures are in place to minimize impact.
- **Simplified maintenance**: Instead of one codebase, you'll have many. Have guidelines and tools in place to ensure consistency.

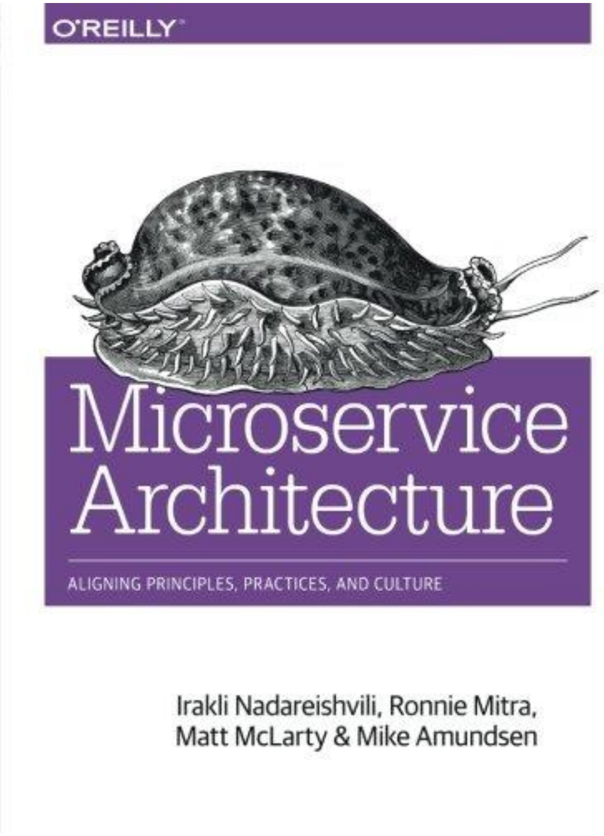
http://www.vinaysahni.com/best-practices-for-building-a-microservice-architecture

Textbook



Sam Newman:
Building Microservices
O'Reilly and Associates, USA, 2015. ISBN 978-1-491-95035-7

Textbook



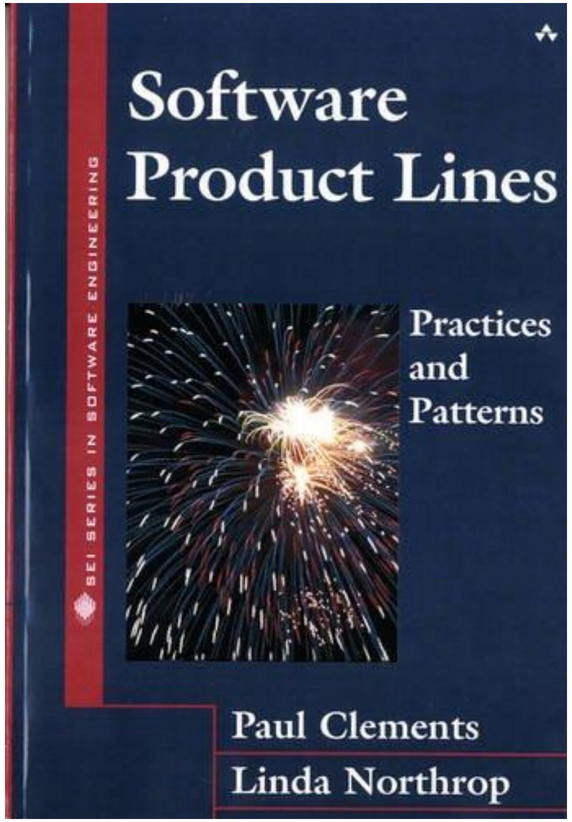
Mike Amundsen, Matt Mclarty:
Microservice Architecture – *Aligning Principles, Practices, and Culture*
O'Reilly Media, Inc, USA, 2016. ISBN 978-1-491-95625-0

Recommendations

Architecture Recommendations for Microservices

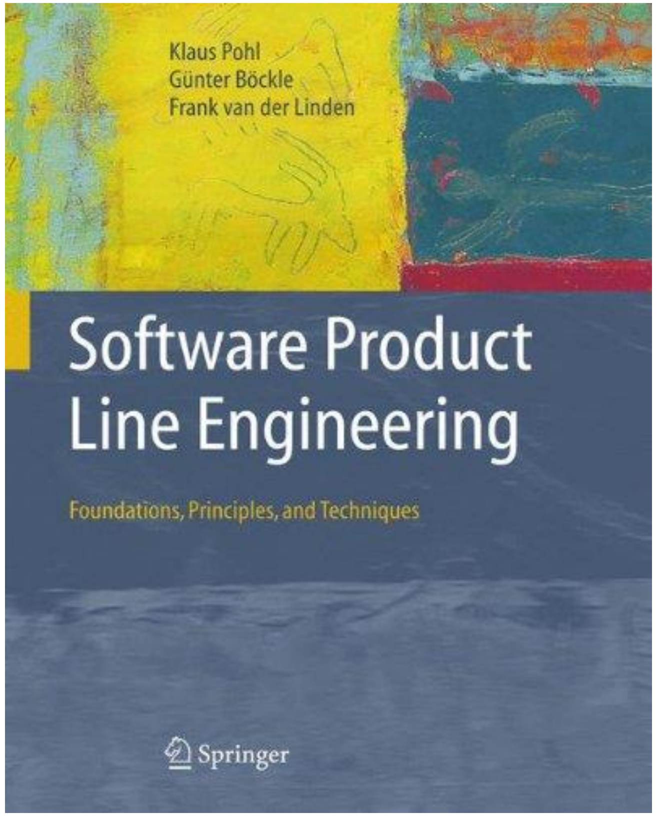
1. Use microservices only if they fit into the *overall architecture* and strategy (enterprise strategy)
2. Respect all *architecture principles* while building and evolving microservices
3. Establish a working governance structure for microservices
4. Base microservices on the *domain model* (domain driven engineering)
5. *Automate* the management/deployment of microservices and the integration infrastructure

Textbook



Paul Clements, Linda Northrop:
Software Product Lines – *Practices and Patterns*
Addison Wesley Inc., USA, 2015. ISBN 978-0-134-42408-8

Textbook



Klaus Pohl, Gunter Bockle, Frank J. Linden:
Software Product Line Engineering – *Foundations, Principles and Techniques*
Springer-Verlag, Berlin, 2010. ISBN 978-3-642-06364-0

Part 3 C