Summary of Lecture 11.12.2019

http://englishskills.se

... Very condensed summary of the 11.12.2019 lecture

https://de.fotolia.com

Summary 11.12.2019

Horizontal Architecture Layer Principles

- A1: Architecture Layer Isolation
- A2: Partitioning, Encapsulation and Coupling
- A3: Conceptual Integrity
- A4: Redundancy
- A5: Interoperability
- A6: Common Functions
- A7: Reference Architectures, Frameworks and Patterns
- A8: Reuse and Parametrization
- A9: Industry Standards
- A10: Information Architecture
- A11: Formal Modeling
- A12: Complexity and Simplification

Summary 11.12.2019

Horizontal Architecture Layer Principles

- A1: Architecture Layer Isolation
- A2: Partitioning, Encapsulation and Coupling
- A3: Conceptual Integrity
- A4: Redundancy
- A5: Interoperability
- A6: Common Functions
- A7: Reference Architectures, Frameworks and Patterns
- A8: Reuse and Parametrization
- A9: Industry Standards
- A10: Information Architecture
- A11: Formal Modeling
- A12: Complexity and Simplification

Summary 11.12.2019

Why models?

http://www.portlandart.net

Adequate Models provide:

√ **Clarity**

√ **Committment**

√ **Communication**

√ **Control**

C The **4** C's of models

Summary 11.12.2019

C The **4** C's of models

Clarity C_1

The concepts, relationships, and their attributes are unambigously *defined* and *understood* by all stakeholders

Committment C_2

All stakeholders have *accepted* the model, its representation and the consequences (agreement)

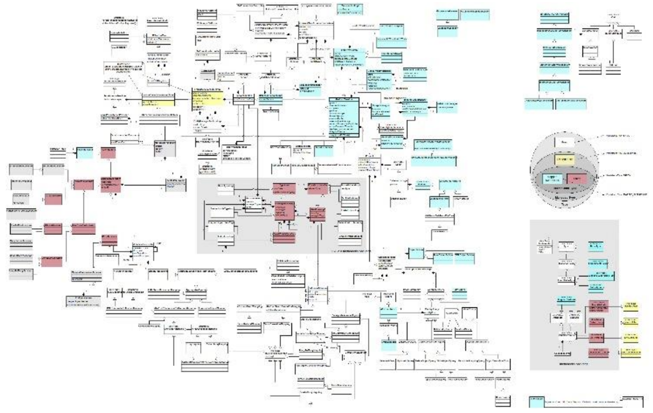
Communication C_3

The model truly and sufficiently represents the key properties of the real world to be mapped into the IT-solution

Control C_4

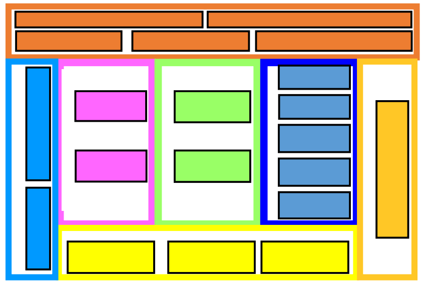
The model is used for the assessment of specifications, design, implementation, reviews and evolution

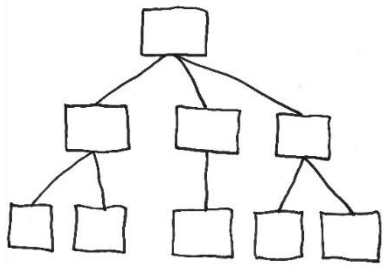
Summary 11.12.2019



How can we handle model size & complexity?

Domains | **Hierarchical refinement** | **Views** | **Tools**





Summary 11.12.2019

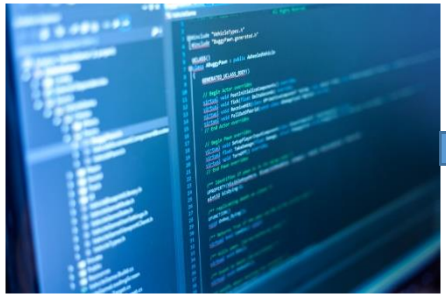
Which are today‘s engineering modeling solutions?

Mature and in wide use:
- ✓ Domain Models
- ✓ Business Object Models
- ✓ Web-Standards (WSDL, …)
- ✓ OCL
- ✓ Ontologies (OWL-DL)
- ✓ UML, SysML + Profiles
- ✓ State machines
- ✓ Timed automata
- ✓ Simulink Models
- ✓ ERD for Databases

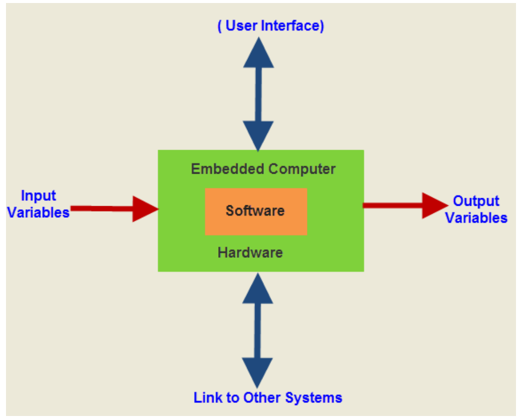
Emerging and in selected use:
- ✓ Domain Specific Languages
- ✓ Contracts (CSLs)
- ✓ (Coloured) Petri Nets
- ✓ Annotated, directed hypergraphs
- ✓ Graph rewriting
- ✓ Role-based modeling (RoSI)

Waiting in the trenches:
- ✓ «Z»-Language
- ✓ «Event-B» Language
- ✓ Certified Code generators
- ✓ Correctness provers

Summary 11.12.2019



Software



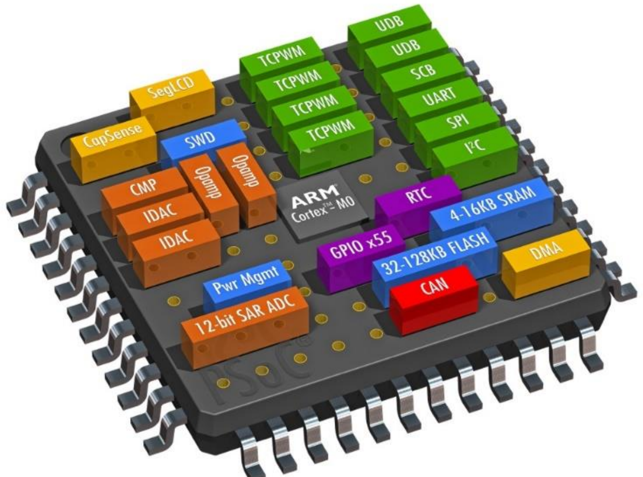
System

SysML:
Systems Modeling Language

DEFINITIONS

The Systems Modeling Language (**SysML**) is a general-purpose modeling language for systems engineering applications. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems.

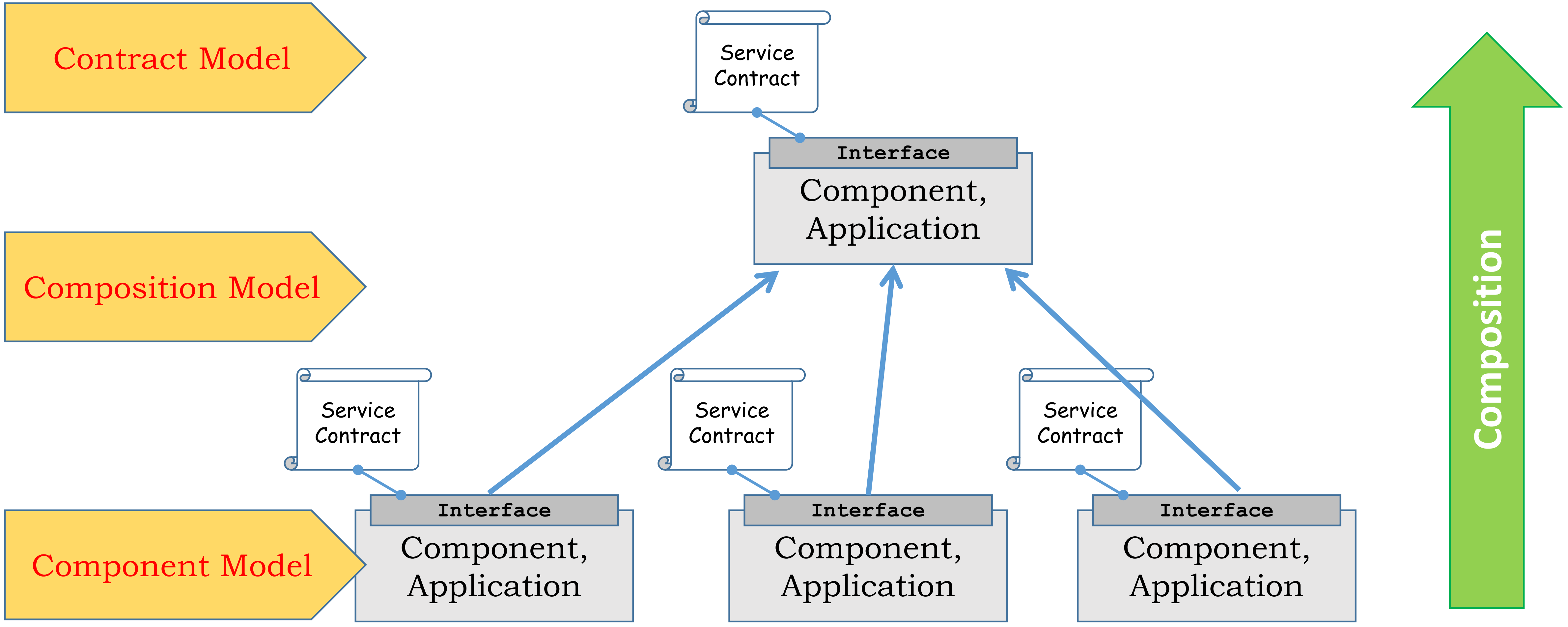
SysML expresses **systems engineering** semantics (interpretations of notations) better than than UML. SysML is smaller and easier to learn than UML. Since SysML removes many software-centric constructs, the overall language is smaller as measured in diagram types (9 vs. 13) and total constructs.



https://www.thegeniusworks.com

Summary 11.12.2019

The Future: Contract-Based Systems Engineering



Summary 11.12.2019

Horizontal Architecture Layer Principles

- A1: Architecture Layer Isolation
- A2: Partitioning, Encapsulation and Coupling
- A3: Conceptual Integrity
- A4: Redundancy
- A5: Interoperability
- A6: Common Functions
- A7: Reference Architectures, Frameworks and Patterns
- A8: Reuse and Parametrization
- A9: Industry Standards
- A10: Information Architecture
- A11: Formal Modeling
- A12: Complexity and Simplification

Summary 11.12.2019

DEFINITIONS

“Complexity is that property of an IT-system which makes it difficult to formulate its overall behaviour, even when given complete information about its parts and their relationships“

Complexity = (IT-) Risk

Summary 11.12.2019

Essential complexity

… is the *inherent* complexity of the system to be built.

Essential complexity for a given problem *cannot* be reduced.

It can only be lessened by *simplifying* the requirements for the system extension.

⇒ However, essential complexity can be *managed* and its negative effects can be *minimized* by good architecture

Accidental Complexity

… is introduced in addition to the essential complexity by our development activities or by constraints from our environment.

This is unnecessary and threatening complexity!

⇒ *Avoiding* and *eliminating* accidental complexity is a continuous task in the development process – from requirements to deployment!

Summary 11.12.2019

Managing Complexity

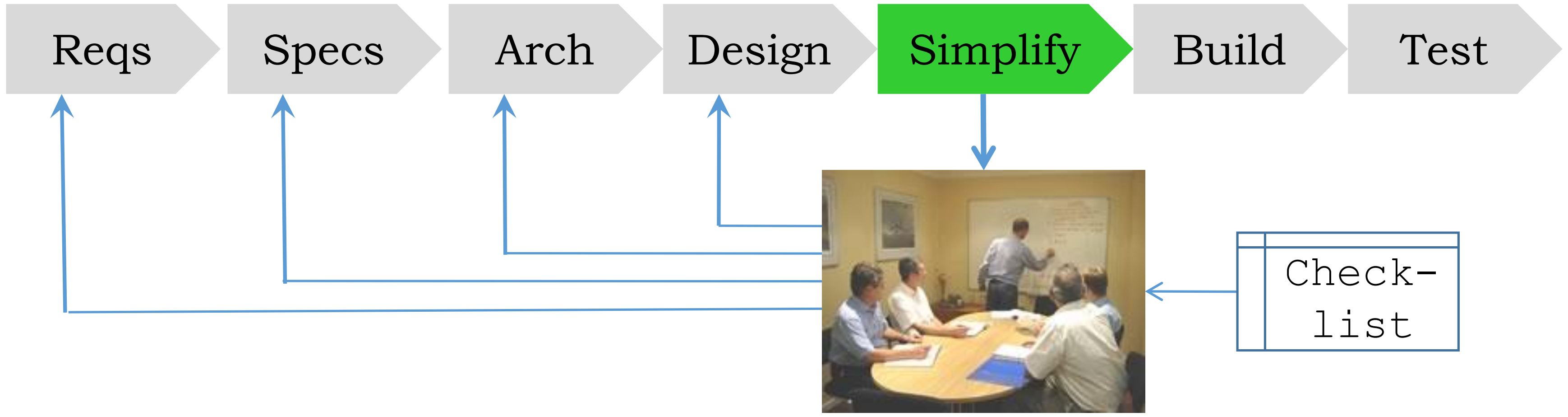
Complexity	*Known* (identified) Complexity	*Unknown* (hidden) Complexity
Necessary (desired) Complexity [*Essential* Complexity]	**manage it**	**use it carefully**
Unnecessary (undesired) Complexity [*Accidental* Complexity]	**avoid it** **eliminate it**	**attack it**

- OS
- DBMS
- TCP/IP Stack
- etc.

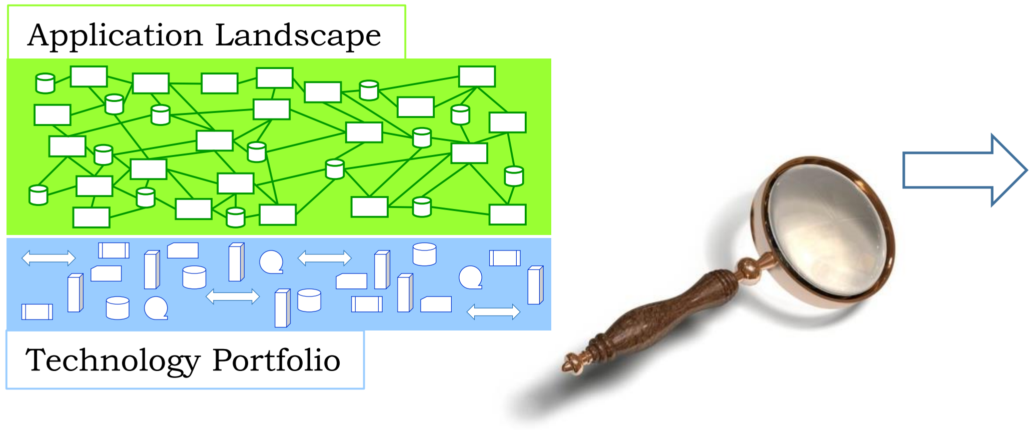
- Technical debt
- Architecture erosion

Summary 11.12.2019

Implement a process step „*simplification*“ in your development process



Periodically carry out re-architecture programs „*complexity reduction*“



Re-Architecture Program 2014

- ➢ Eliminate …
- ➢ Refactor …
- ➢ Replace …
- ➢ Redesign …
- ➢ etc.

Effort: 1‘100 MM

Cost: 27 M€

http://blogs.proquest.com

Summary 11.12.2019

… Continue with Part 4A

Dependability