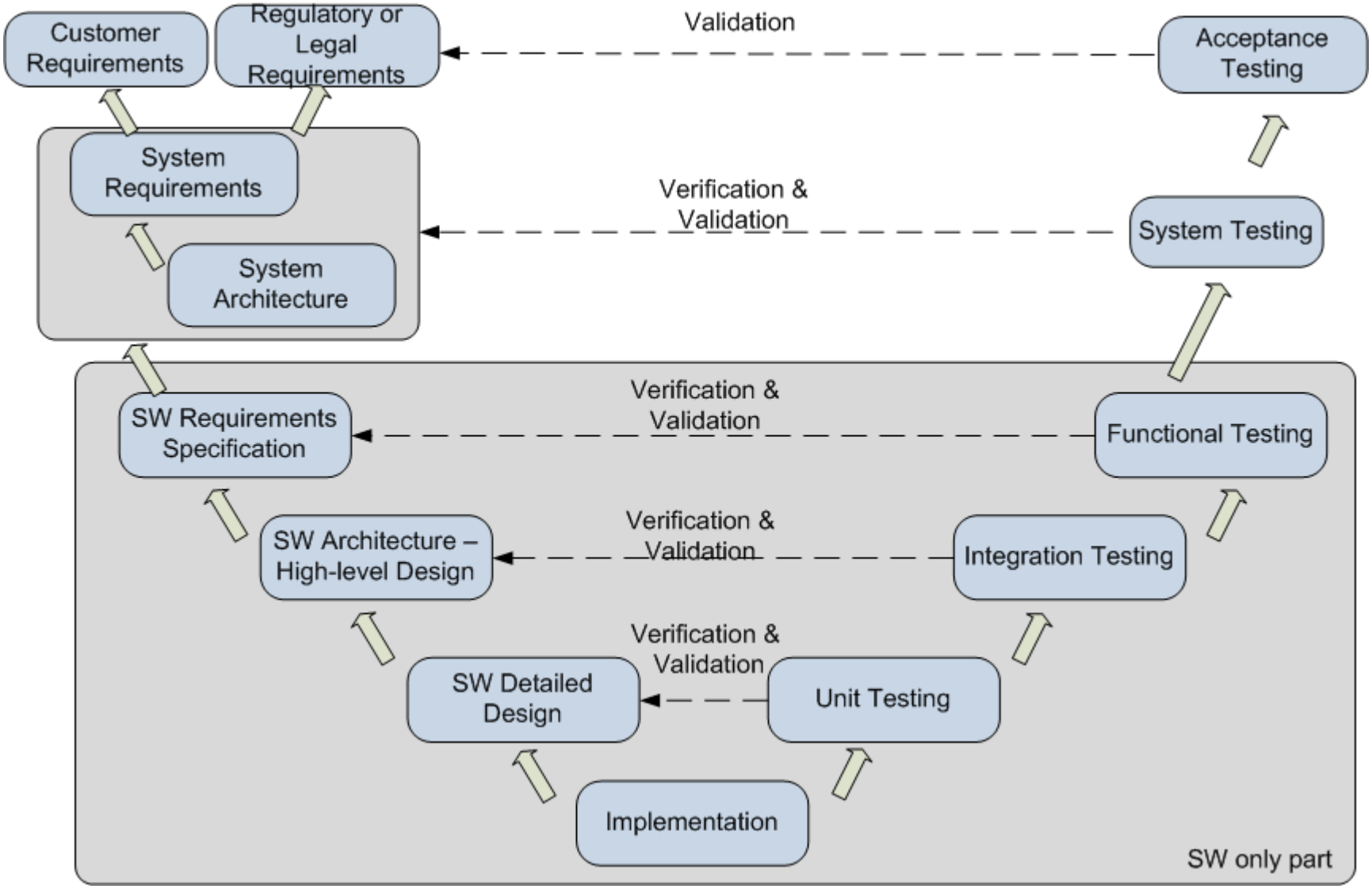
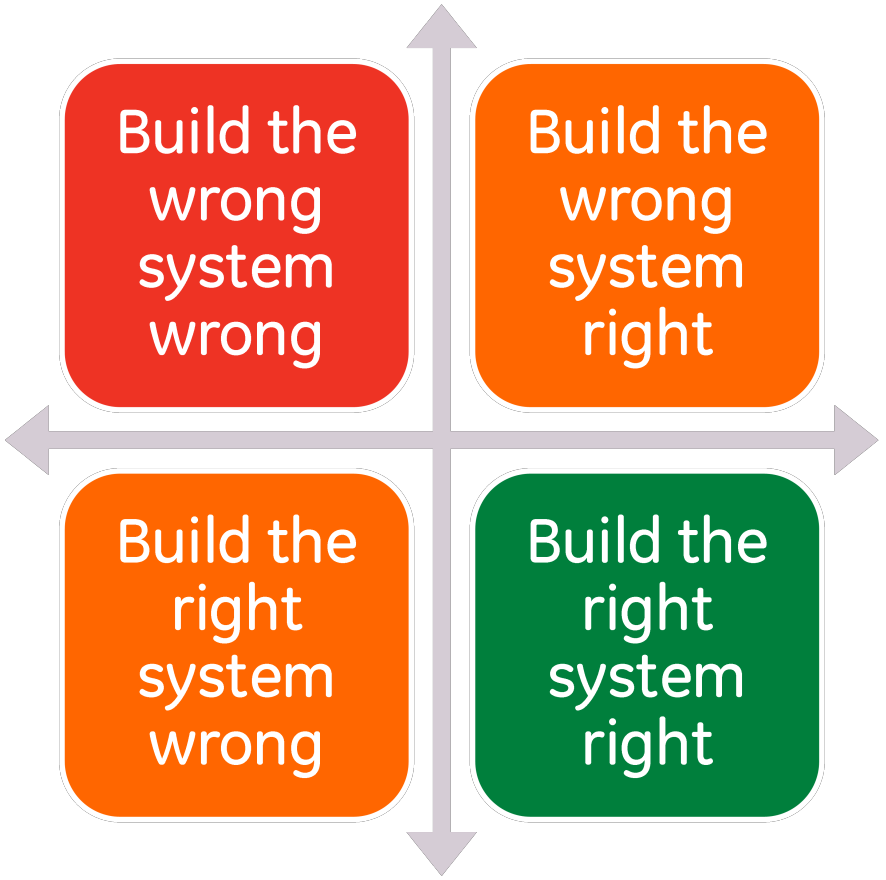
Industrial Embedded Systems
- Design for Harsh Environment -

Dr. Alexander Walsch
alexander.walsch@ge.com
IN 2244
Part VIII
WS 2014/15
Technische Universität München

The V-Model


Customer Requirements
Regulatory or Legal Requirements
Validation
Acceptance Testing
System Requirements
System Architecture
Verification & Validation
System Testing
SW Requirements Specification
Verification & Validation
Functional Testing
SW Architecture – High-level Design
Verification & Validation
Integration Testing
SW Detailed Design
Verification & Validation
Unit Testing
Implementation
SW only part

Verification & Validation



Verification:

Have we done it right?
This is usually an internal activity without customer involvement. Its intent is to show that the specification as compiled by technology has been implemented correctly.

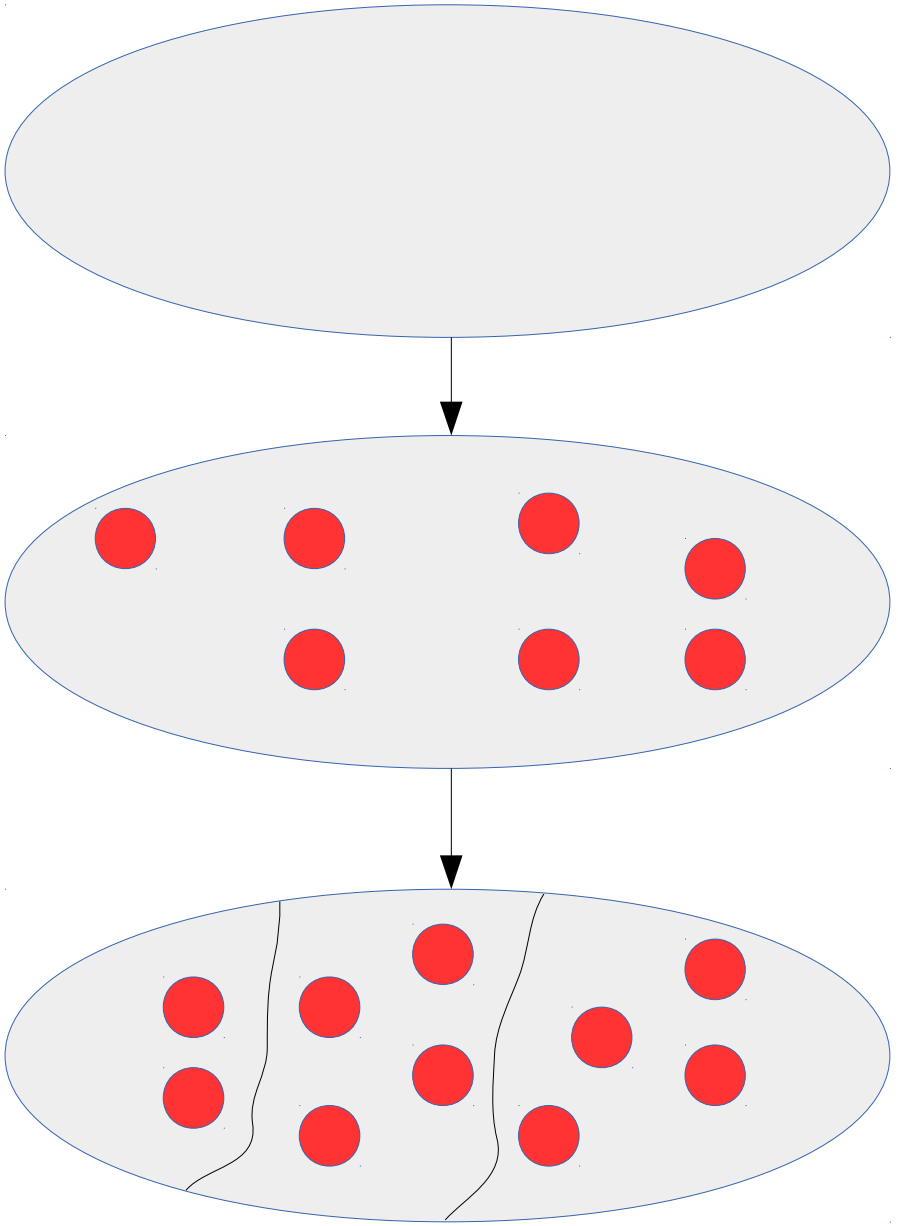
Validation:

Have we done the right thing?
This is usually an external activity with customer involvement (pilot). Its intent is to show that the system meets the customer's expectations.

Verification and Validation Approaches

- Combination of inspections/reviews and test
  - Software QA (life cycle activities/process)
  - Document, coding standards and style guides
  - Planned test campaigns (different phase of V, different objective)
- Additional formal verification methodologies:
  - Model Checking – property checking within a state machine like model
  - Abstract Interpretation – property checking within the an approximation of a computer program
  - Theorem Proving – property is a logical consequence within the scope of a model

Testing Strategy



All possible states and behaviour of a system

Arbitrary tests

Partitioning and testing of each partition

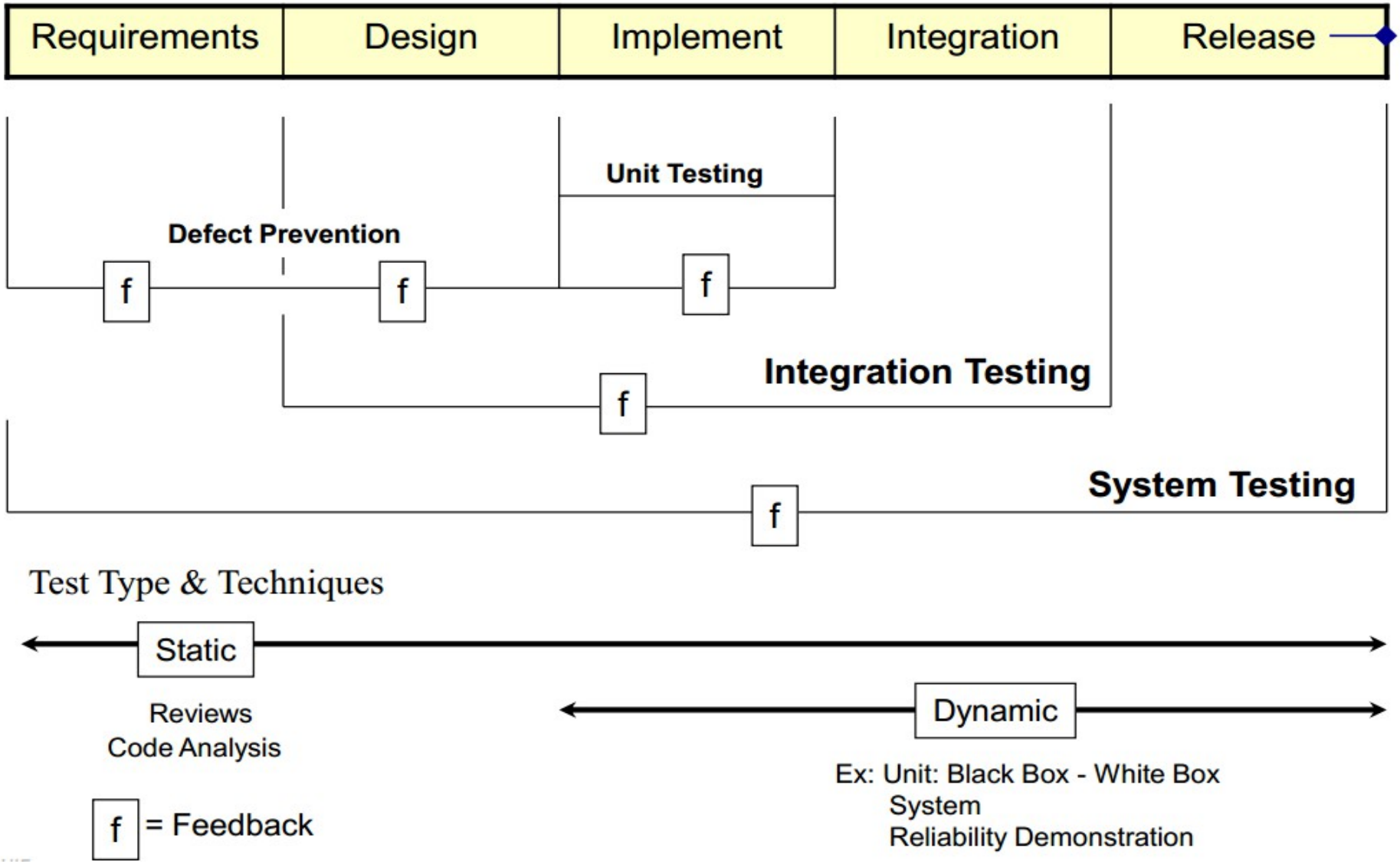
Static and Dynamic Analysis

- Static analysis
  - Code complexity, code compliance (e.g. MISRA C)
  - Run-time faults (e.g. division by zero)
  - Other (e.g. timing)

Dynamic analysis (test case required – executable!):

- Functional analysis (correctness, based on functional requirements)
- Structural analysis (statement → MCDC, did I consider all relevant paths through my code?)
- Integration (IO, timing, based on non-functional requirements)

Example Testing Lifcycle for Software



Source: R. Mueller
OPS ala carte

Software Unit (Module) Testing (based on IEC61508-3)

Technique/Measure	Ref	SIL3	Interpretation for DUT
Static analysis	B.6.4 (IEC61508-7) Table B.8 (IEC61508-3)	HR	- automated coding standard compliance tests - design reviews - program not executed
Functional analysis (dynamic/black-box)	B.5.1 (IEC61508-7) B.5.2 (IEC61508-7) Table B.3 (IEC61508-3)	HR	- test against design document - program is executed - boundary value analysis, equivalence classes and input partitioning **Test Environment:** - evaluation board + test harness (I/O)+ PC software (analysis) - development PC
Structural analysis (dynamic/white box)	B.6.5 (IEC61508-7) Table B.2 IEC61508-3)	HR	- Test against design document and code - Boundary value analysis, performance testing equivalence classes and input partitioning (100% branch coverage). **Test Environment:** - Same as functional analysis
Data recording and analysis	C.5.2 (IEC61508-7)	HR	- All testing needs to be documented. Pass/fail criteria need to be in place.

HW/SW Integration (based on IEC61508-3)

Technique/Measure	Ref	SIL3	Interpretation for DUT
Functional analysis (task level + framework)	B.5.1 (IEC61508-7) B.5.2 (IEC61508-7) Table B.3 (IEC61508-3)	HR	**Tests against architecture and design:** - boundary value analysis, equivalence classes and input partitioning (at least one per equivalence class) - Test cases need to cover input, output boundaries and extreme values. Test cases which drive the output to exceed the specification need to be considered **Test Environment:** - target hardware + test harness (I/O)+ PC software (analysis)
Data recording and analysis	C.5.2 (IEC61508-7)	HR	- All testing needs to be documented. Pass/fail criteria need to be in place.
Performance testing	C.5.20 (IEC61508-7) Table B.6 (IEC61508-3)	HR	- Avalanche/stress testing - e.g. high network load, highest sampling rate - Response timings and memory constraints – analysis of the resource usage and elspsed time for every DTU functionality. - Fault insertion testing

Functional Software Testing (System Level)

Technique/Measure	Ref	SIL3	Interpretation for DUT
Functional and black-box testing (target system)	B.5.1 (IEC61508-7) B.5.2 (IEC61508-7) Table B.3 (IEC61508-3)	HR	Test against requirements: boundary value analysis, equivalence classes and input partitioning Test Environment: - target system + sensors and actuators + PC software for analysis