Dagstuhl Seminar “Software Dependability Engineering”

Position Statement

Ronny Kolb and Dirk Muthig
Fraunhofer IESE

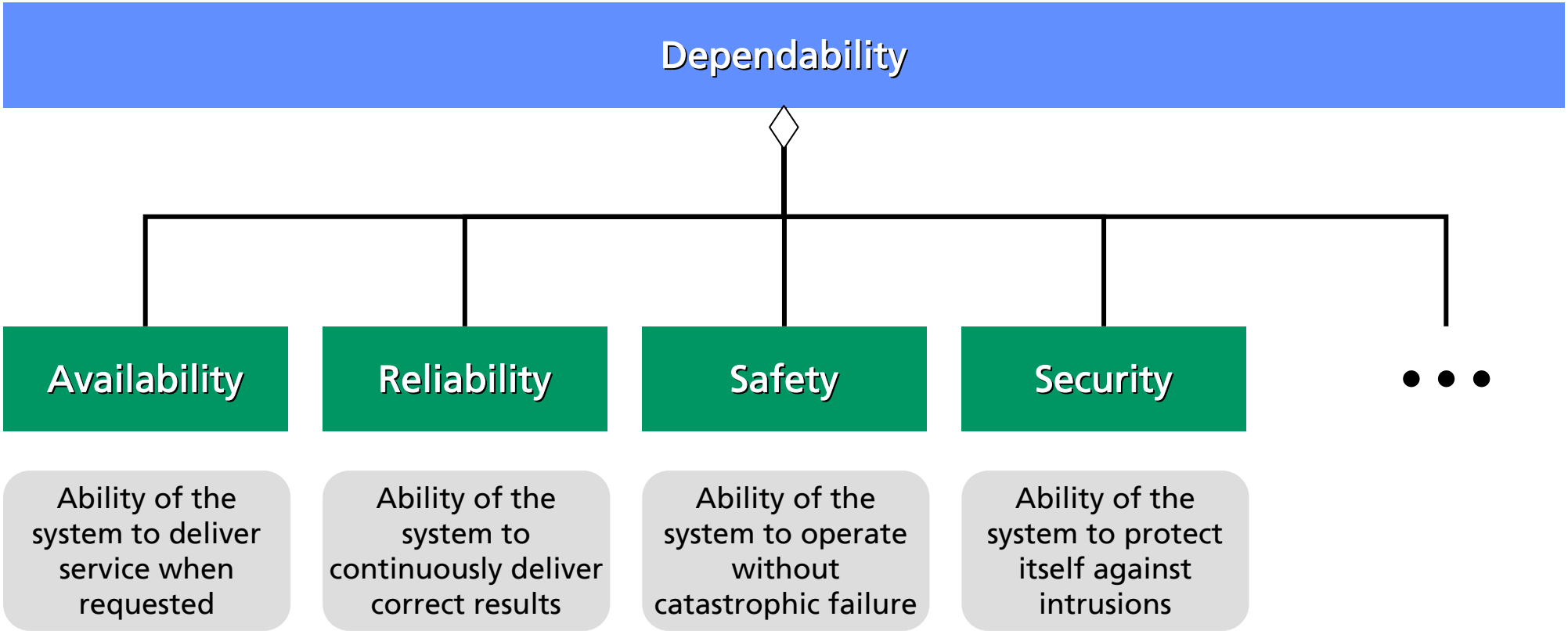
{kolb, muthig}@iese.fraunhofer.de

Fraunhofer Institut
Experimentelles
Software Engineering

Introduction – Dependability (1/2)

- "Dependability is the trustworthiness of a computing system that allows reliance to be justifiably placed on the services it delivers." [IFIP WG-10.4]
- Reflects the users' needs and trust in a system
- Different users/stakeholders might (and usually do) have different perspectives on what dependability is
- Term "Dependability" commonly used with a variety of distinct meanings
- Dependability subsumes different qualities within a single conceptual framework

Introduction – Dependability (2/2)



Introduction – Attributes of Dependability

- Availability
- Reliability
- Safety
- Security
  - Confidentiality: non-occurrence of unauthorized disclosure of information
  - Integrity: non-occurrence of improper alterations of information
- Performance
- Survivability
- Maintainability
- …

Characteristics of Dependability (1/4)

Software that is acceptable in one situation may be deficient in another

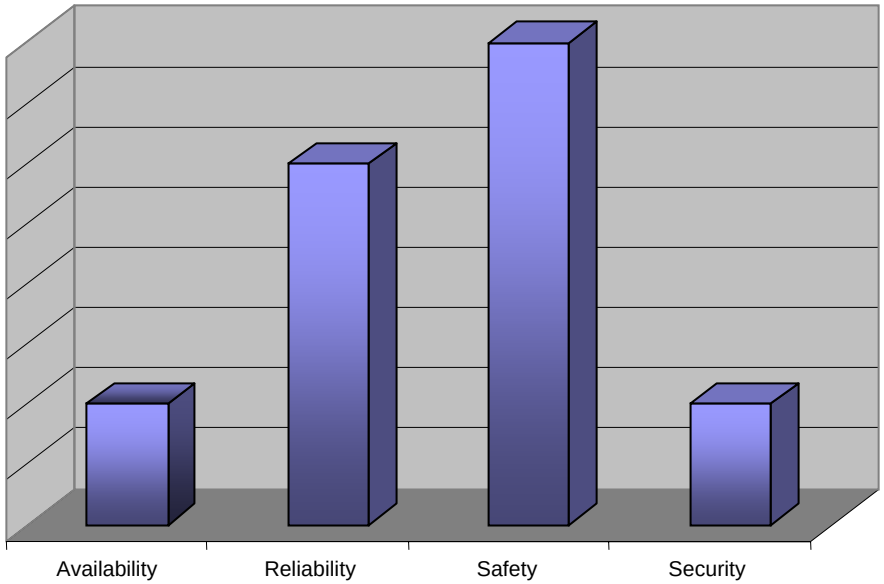
For example, some applications depend on low latency but can tolerate low precision; in other applications precision is critical but latency is not

- "… trustworthiness of a computing system that allows **reliance** to be justifiably placed on the services …"

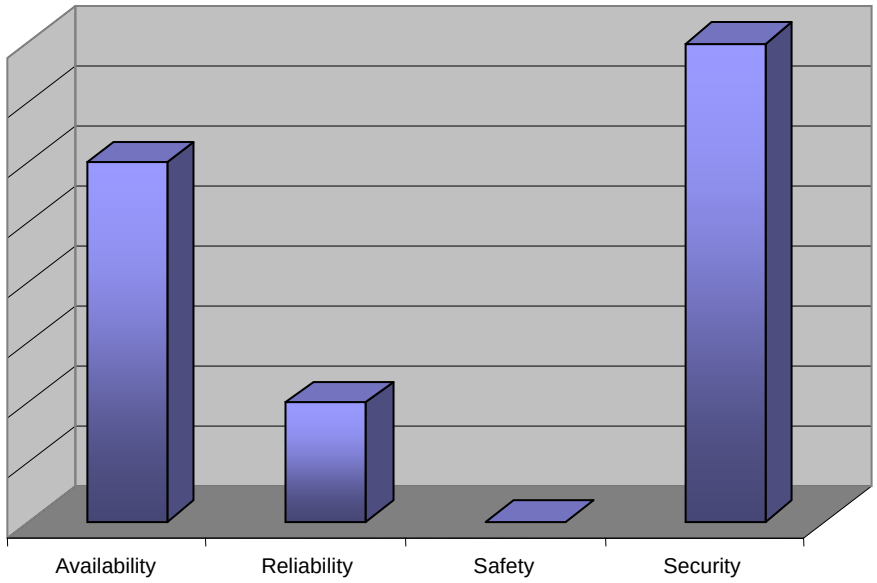
- Contextually **subjective**
  - Different quality attributes of relevance for end-product ➔ May cover *different set of attributes*
  - Same attribute may *mean different things* to different people (different interpretations, overlaps, etc.)
  - Quality attributes not equally important
  - Customer expects that qualities are fulfilled to a certain degree ➔ *Different levels of adherence* to those attributes

Characteristics of Dependability (2/4)

- Quality attributes may be of **varying importance** depending on application domain or type of system: telecommunication systems, automobiles, transportation systems, aircraft, medical devices, financial services, etc.



Embedded Car Control System (e.g. ABS, ESP)



Web-based Financial Information System

Characteristics of Dependability (3/4)

Dependability needs arise from user expectations

➔ **Exact meaning of dependability depends on context**

- type of system,
- application domain,
- the particular stakeholders and their needs,
- etc.

➔ Dependability hast to be defined

- with respect to particular application domains, systems, contexts, and organizations

Characteristics of Dependability (4/4)

- Subsumes **operational properties**, i.e., quality attributes that are observable by users during run-time of a system

➔ Important to understand **interrelationships and tradeoffs among dependability attributes (e.g. performance and security)** as well as dependability attributes and developer-related qualities (e.g. testability and reliability)

What is Software Dependability?

- Dependability is
  - a **collective term** that
  - explicitly defines a **context-specific set** of external, **user-observable system characteristics** or qualities
  - **weighted** according to stakeholders' needs that
  - if **fulfilled to a minimum pre-defined level**
  - collectively **determine the reliance** users of a software system can justifiably place on the service it delivers

- **Dependability = Quality profile** subsuming all quality attributes increasing users' trust in a particular system

What is Software Dependability Engineering?

Costs matter ➔
Few projects can afford highest dependability at any cost

- Software Dependability Engineering
  - subsumes all activities in an organization that together support the **most efficient achievement of dependability requirements** and desired quality levels **under the consideration of time, cost, available resources, and organizational constraints** for a software system
  - is the application of a collection of systematic, disciplined, quantifiable methods and techniques to the design, development, and maintenance of software systems that allow reliance to be justifiably placed on the service they deliver

Software Dependability Engineering – Problems (1/3)

- **Numerous methods and techniques** that enable to assure that a certain level of quality can be achieved exist
  - **Constructive** Quality Engineering Techniques
  - **Analytic** Quality Engineering Techniques (quality assurance techniques)

- Available methods and techniques
  - typically addressing one single quality attribute
  - Not all equally effective and efficient for different qualities

Software Dependability Engineering – Problems (2/3)

- Challenge: **Select techniques** to ensure high-quality products with **restricted resources** (time/cost)
  - Satisfy customers' dependability needs
  - Support organizational goals

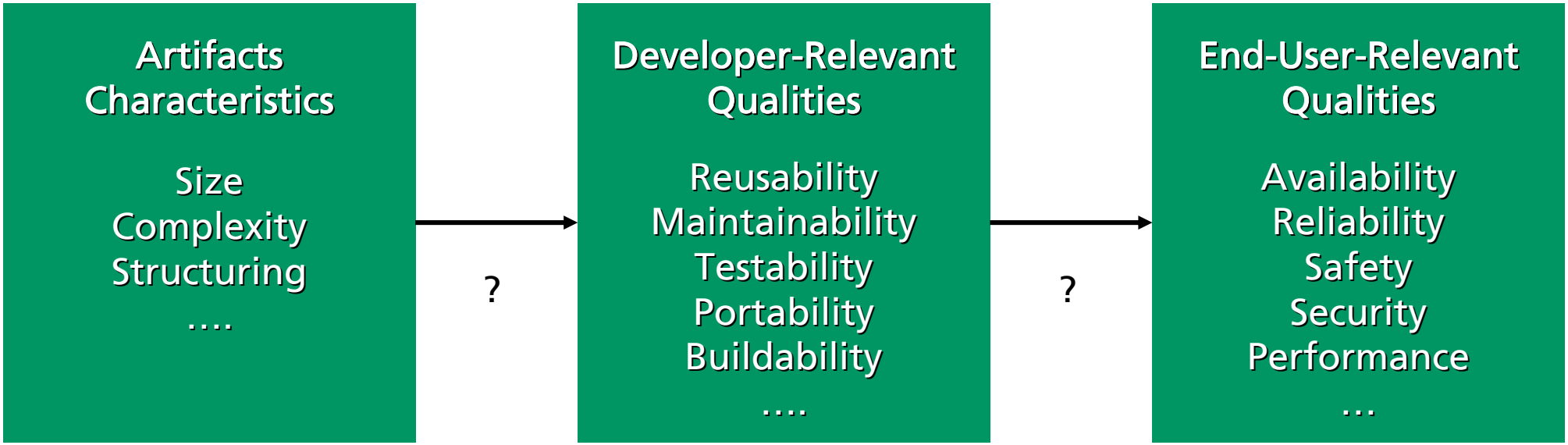
➔ Available resources need to be spent in a most efficient and systematic way

➔ Essential to address different qualities and defects with **most suitable** quality engineering techniques

- **Define customized (quality) engineering strategies matching dependability needs of particular projects**

Software Dependability Engineering – Problems (3/3)

- Identify dependencies between quality aspects in the different life-cycle phases
- Relating artifact qualities to system/customer qualities



How to Achieve Dependability? (1/5)

- Determine dependability requirements (i.e., quality profile) for particular system
- Consider individual dependability attributes during all life cycle phases and address them individually according to their priority/weight using the most effective and efficient techniques and methods
- Focus on the architecture design phase

- Not necessarily new quality engineering techniques and methods required, but clever combination of existing ones in the different life cycle phases ➔ **Quality Engineering Strategies**

How to Achieve Dependability? (2/5)

Quality Engineering Strategy

- Defines applied quality engineering methods and techniques
- Balances between various quality engineering activities

Strategies describe

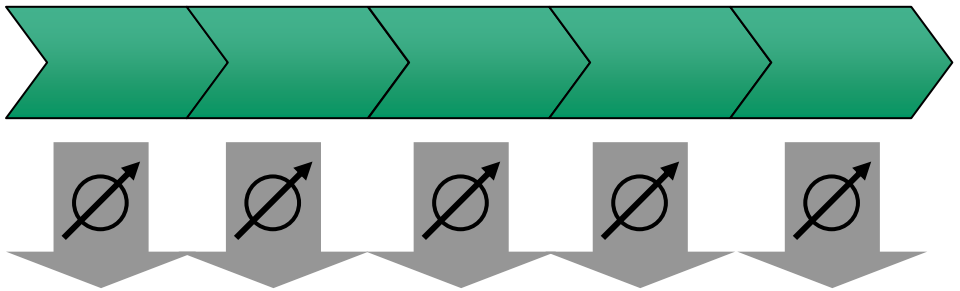
- Which methods and techniques are used for quality assurance
- To which artifacts quality assurance techniques are applied
- To which extent quality assurance activities are performed
- Who performs quality assurance activities

How to Achieve Dependability? (3/5)

- **Planning** of constructive and analytic quality engineering activities (time/resources)
  - Includes setting explicit quality objectives for intermediate development artifacts
- **Optimizing** individual activities according to context
- **Integrating** different activities into holistic quality approach
- **Measuring** the success of the approach ➔ Continuous learning
  - Unless results of quality assurance can be measured there is no way to know if plan is working

How to Achieve Dependability? (4/5)

- Define **quality indicators** that "measure" the desired target qualities during the entire development process
- Key-concepts
  - Use information gathered in early phase quality assurance to steer/focus quality assurance activities in later phases (➔ Defect classifications and defect flow models
  - Establish holistic "quality view" over life cycle phases



Dependability

How to Achieve Dependability? (5/5)

Prerequisites

- Understand relationships between development artifacts qualities and the dependability attributes of end product
- Definition of different quality attributes in the different life cycle phases
  - e.g., what means security in a component specification?

References (1/2)

- A. Avizienis, J.-C. Laprie, and B. Randell. *Dependability of Computer Systems: Fundamental Concepts, Terminology, and Examples*. Technical report, LAAS-CNRS, October 2000.
- A. Avizienis, J.-C. Laprie, and B. Randell. *Fundamental Concepts of Dependability*. Research Report N01145, LAAS-CNRS, April 2001.
- V. Basili, P. Donzelli, S. Asgari. *Modeling Dependability: The Unified Model of Dependability*. Computer Science Department, University of Maryland, College Park, MD, Technical Report CS-TR-4601 – UMIACS-TR-2004-43, June 2004.
- P. Costa, I. Rus. *Characterizing Software Dependability from Multiple Stakeholders' Perspective*. Software Tech News, 6(2), December 2003.

References (2/2)

- J.-C. Laprie (ed.). *Dependability: Basic Concepts and Terminology*. In Dependable Computing and Fault Tolerance, Vienna, Austria, Springer-Verlag, December 1992.
- IFIP WG 10.4. *Dependability: Basic Concepts and Terminology.* IFIP Working Group on Dependable Computing and Fault Tolerance, October 1990.
- I. Rus, S. Komi-Sirvio, P. Costa. *Software Dependability Properties: A Survey of Definitions, Measures and Techniques*. Fraunhofer Center for Experimental Software Engineering, College Park, MD. Technical Report 03-110, January 2003.