future internet technologies
by FOKUS

Assessment for Future Internet

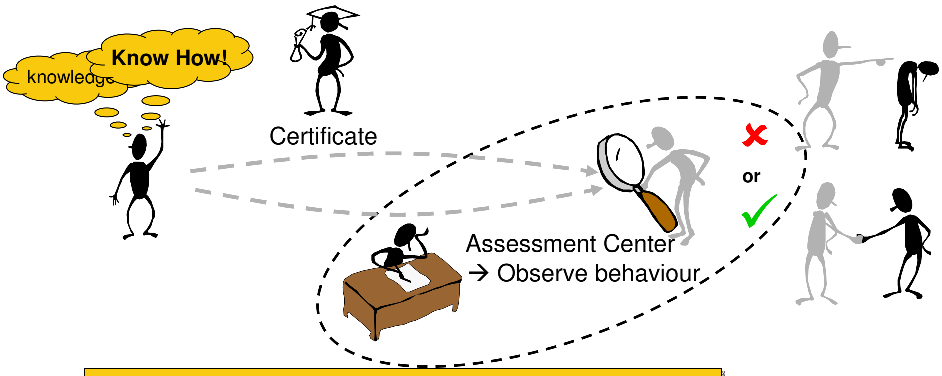
Towards Trust in Self-Managed Networks

Jens Tiemann, Dr. Mikhail Smirnov
Competence Center Network Research

Fraunhofer
FOKUS

Motivation: **Assessment Center / Business Simulation Game**

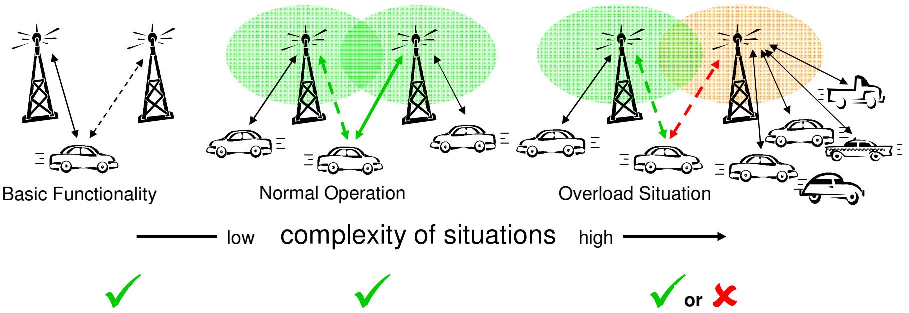
- Employing an **expert** for **project** work



➔ Simple rules can lead to complex situations

Motivation: **Selecting Self-Managed Network Components**

- Example: Self-Managed Flexible Base Station
  - Self-configuration, Load-balancing, Cell-outage compensation, …



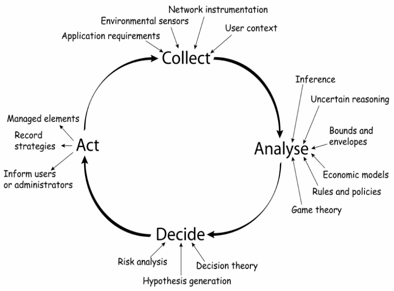
➔ Ready for complex, even unexpected situations?

Overview

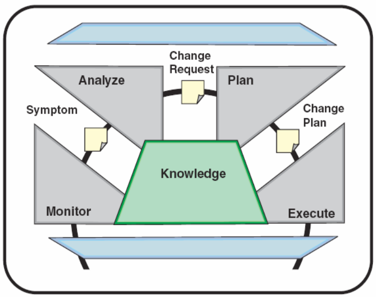
- Objectives for Evaluation of
  - FI Systems → Self-Managed Systems → Self-Adaptive Algorithms
- Expected System Behaviour
- Existing Approaches / What can we use?
  - Test and Evaluation of Communication Systems
  - Software Development and Benchmarking of Computer Systems
- System Model and Interfaces
- Assessment Framework
  - Illustrated by an example

Basis: **Self-Managed Networks / Components**

- **Network Management FCAPS** turns into
  - self-healing
  - self-configuration
  - self-optimization
  - self-protection



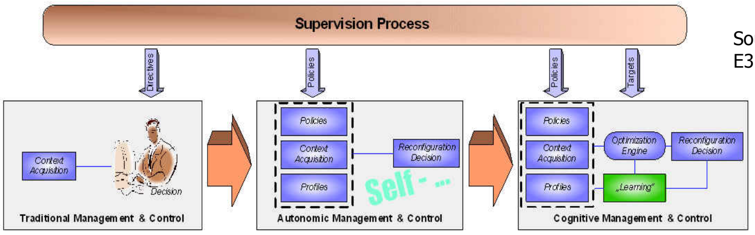
Autonomic Communication [Dobson]:
Collect-Analyze-Decide-Act



Autonomic Computing [IBM]:
MAPE **-** Monitor-Analyze-Plan-Execute

**➔ Central Control Loop:
Different terminology,
similar approaches**

Objectives



Source:
E3 Project

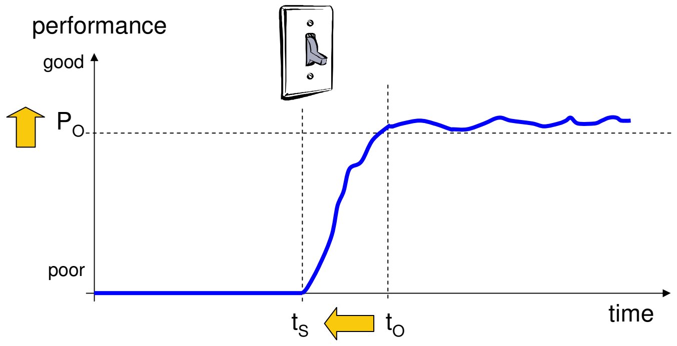
- Evaluation/Testing of self-adaptive systems
  - one approach to understand new system features,
    → important for design & implementation
  - prove the reliability of system operation
    → important for comparison & deployment
- Topics covered in this presentation:
  - Expected system behaviour & metrics ("what to measure?")
  - System Interfaces for evaluation purposes ("how to influence/interact?"))
  - Method ("how to evaluate?")

Observe System Behavior: **Key Performance Indicator (KPI)**

- aggregate of performance measurements related to long-term optimisation
- Viewpoint on System
  - External
    - monitor system behaviour from the outside, e.g. performance
  - Internal
    - measures use of internal resources, e.g. memory usage
  - Observer
    - attempt to describe adaptation towards a global, optimal solution

      → perfect solution might be not possible in dynamic environments
- Type of Metrics
  - Traditional performance metrics
  - New performance metrics from self-adaptive operation
    - describe new characteristics of systems,
      like the time to adapt or the quality of adaptation
  - Abstract metrics to summarize system characteristics
    - rate and compare systems at a first glance

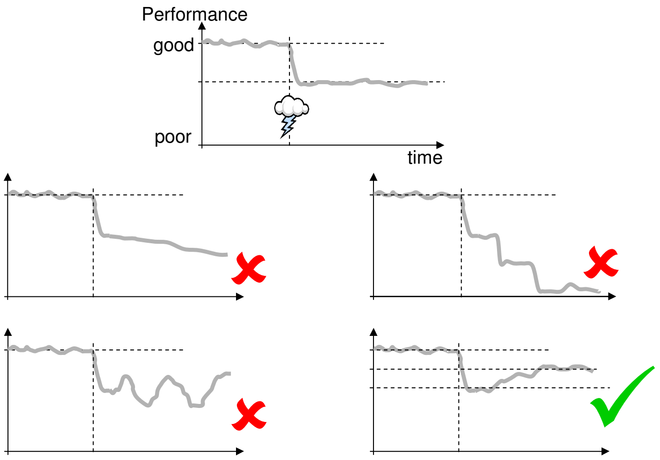
Example of System Operation: **Self-Configuration**



- system configures itself and reaches at t_O an expected/accepted min. performance P_O
- performance metric can be throughput, coverage, … → any KPI
- gain from self-adaptive algorithms / expected behaviour:

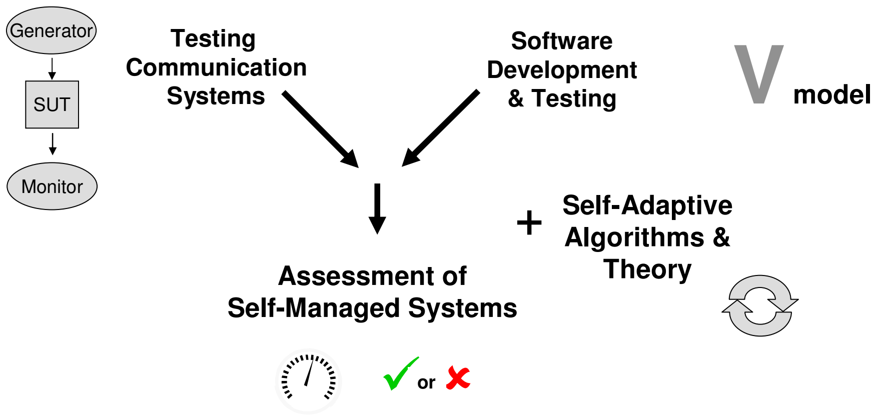
➔ improve level of performance, without human intervention

Examples of System Operation: **Fault and Self-Healing**



➔Metrics needed to describes ability to solve problems

Existing Approaches: **What can we use to start?**



Generator

SUT

Monitor

Testing Communication Systems

Software Development & Testing

V model

Assessment of Self-Managed Systems

+ Self-Adaptive Algorithms & Theory

✓ or ✗

➔ this reflects evolution of network devices

From Networking: **Testing Communication System**

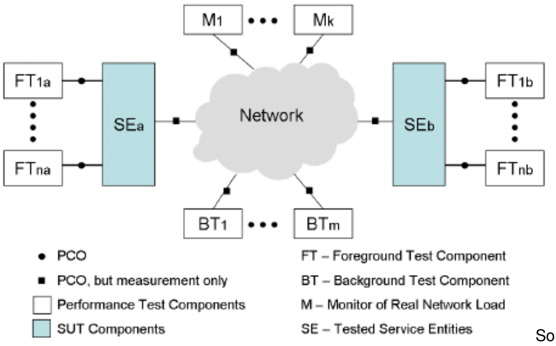
- Conformance Tests
- Performance Measurements

- Point Correctness vs. Process Correctness
  - any self-adaptation step needs to be correct

- Improved Test Configurations
  - Generation of packets, messages or traffic flows
  - new possibilities based on Context and Context Management

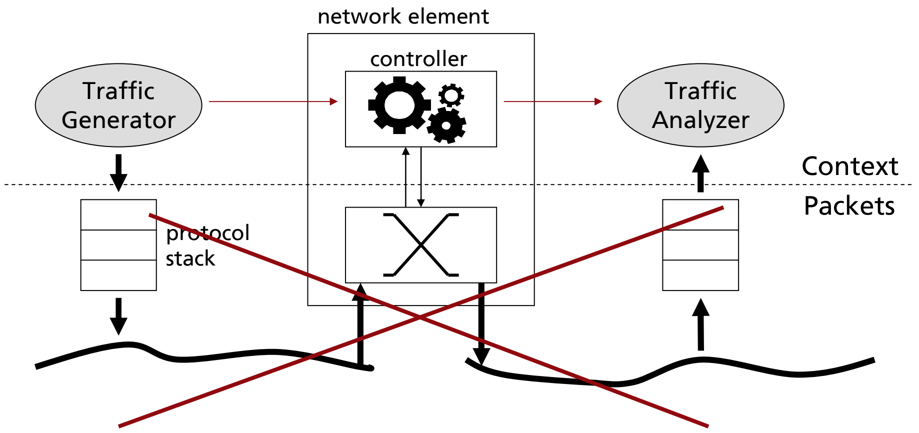
From Networking: **Testing Communication System**

- Test Configurations: Generation of Test Traffic and Monitoring
  - Background Tester – no interaction with the SUT
  - Foreground Tester – interacts with SUT, more expensive


M1
Mk
FT1a
FTna
SEa
Network
SEb
FT1b
FTnb
BT1
BTm
● PCO
▪ PCO, but measurement only
□ Performance Test Components
▨ SUT Components
FT – Foreground Test Component
BT – Background Test Component
M – Monitor of Real Network Load
SE – Tested Service Entities

Source: Schieferdecker / PerfTTCN

Evaluation: **Packet Flow vs. Context Abstraction**

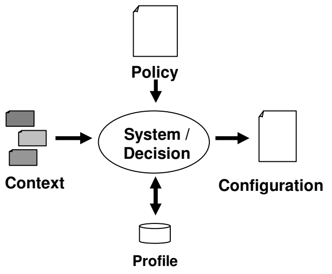


➔ Isolate Algorithm and use Context for Evaluation

From Software: **Software Development & Benchmarks**

- Software Development
  - structured approaches, esp. to cover complexity of systems
- Life-cycle of a self-adaptive system, reflected in evaluation
  - Design: Function blocks as substratum for later operation
  - Implementation: Basic algorithms and configurations
  - Operation: Guided by policies and depending on environment
- Benchmarks in computing
  - usually associated with performance characteristics of hardware
  - using real programs or develop synthetic benchmarks
  - in autonomic computing:
    introduction of events, system needs to react

Simplified System Model



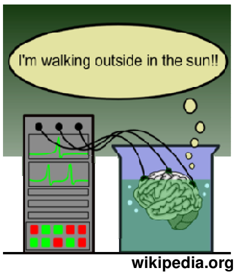
Source: ORACLE Project

- Policy:
  - management of device
- Context:
  - in general any information that can be used to characterize the situation of an entity
- Profile:
  - description/storage of device/user settings
- Configuration:
  - output to system e.g. update of routing table or configuration of radio interface
- System operation is based on information / context management

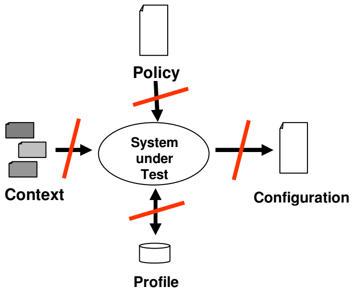
→ Context must be used for evaluation of self-* systems

System Model: **System Interfaces used for Testing**

Brain in a vat:



wikipedia.org

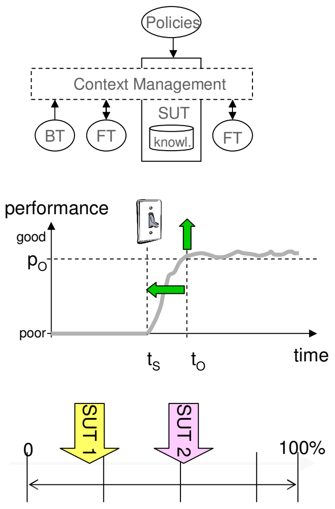


- Isolation of the self-adaptive system or (better) algorithms
- Generation of test data replacing (part of) real context
- Isolation and generation possibly at higher level of abstraction:
  - packets on the wire → identified data flows
  - context → situation

Bringing things together: **Assessment Process**

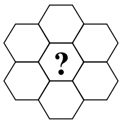
- Create an Environment
  - influence a context-aware system
- Metrics
  - measure system performance in response to complex situations
- Method
  - multiple benchmarks with increasing difficulty → shows problem solving

→ compare or rate systems

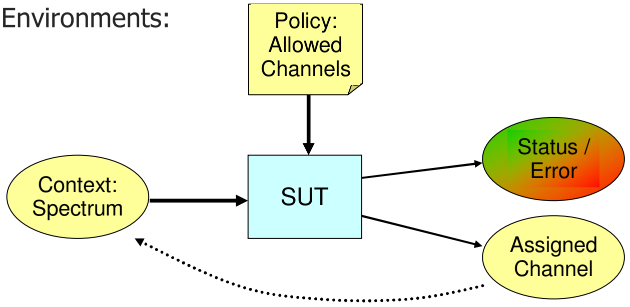


Assessment Example: **Channel Allocation**

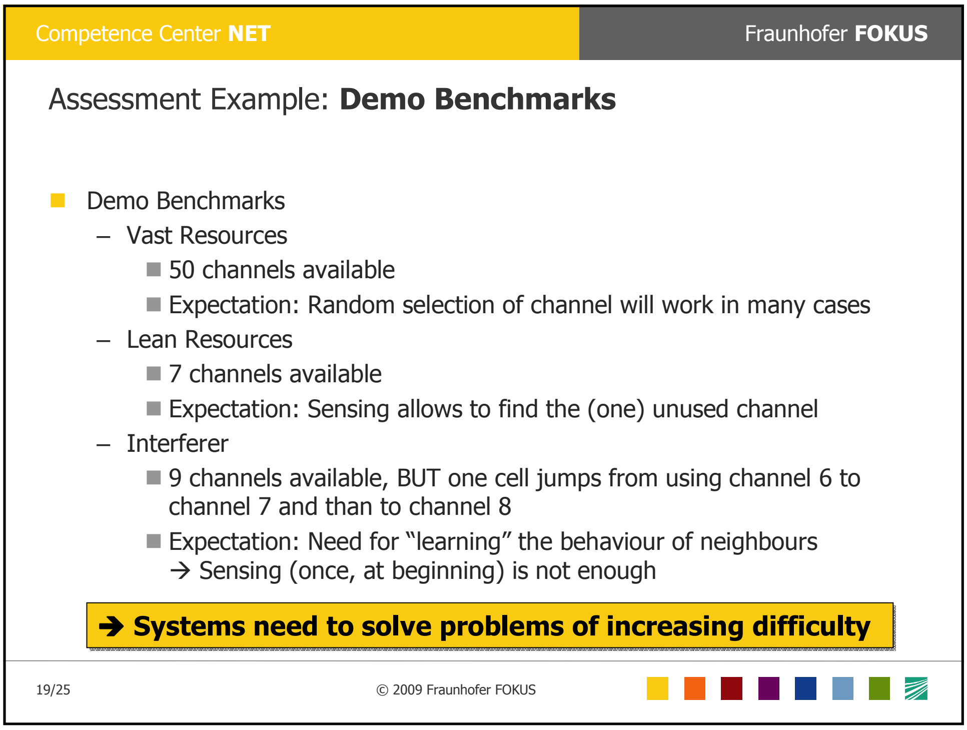
- Channel Allocation by a Base Station / Access Point for a cell



- System and Environments:



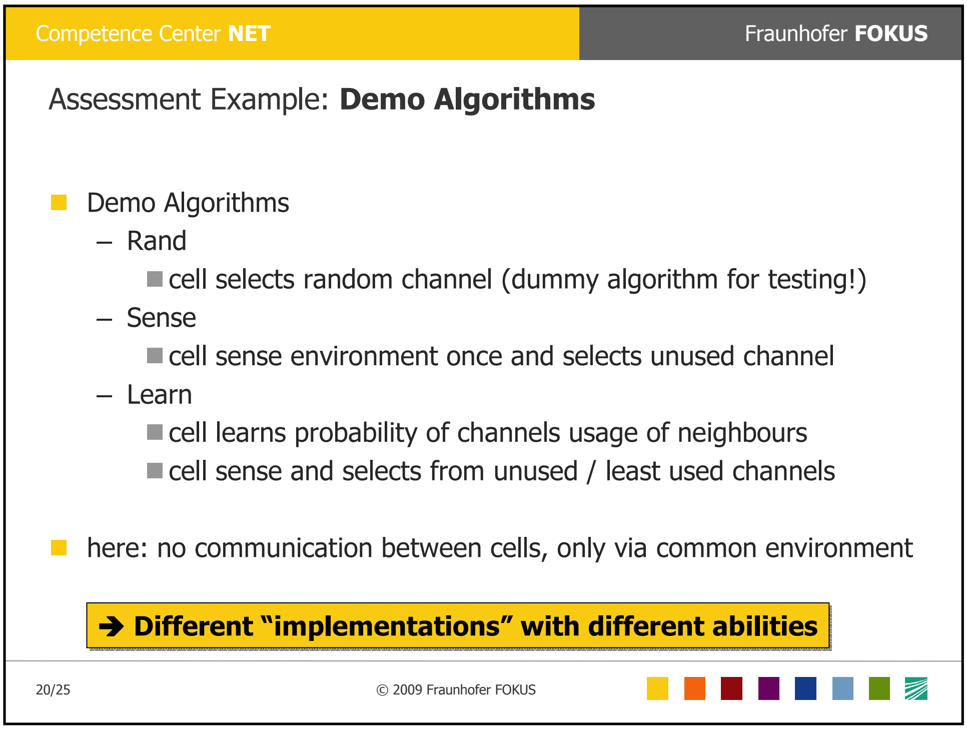
→ Goal of Example: Illustration of Assessment Process



Assessment Example: **Demo Benchmarks**

- Demo Benchmarks
  - Vast Resources
    - 50 channels available
    - Expectation: Random selection of channel will work in many cases
  - Lean Resources
    - 7 channels available
    - Expectation: Sensing allows to find the (one) unused channel
  - Interferer
    - 9 channels available, BUT one cell jumps from using channel 6 to channel 7 and than to channel 8
    - Expectation: Need for "learning" the behaviour of neighbours
      → Sensing (once, at beginning) is not enough

➔ Systems need to solve problems of increasing difficulty



Assessment Example: **Demo Algorithms**

- Demo Algorithms
  - Rand
    - cell selects random channel (dummy algorithm for testing!)
  - Sense
    - cell sense environment once and selects unused channel
  - Learn
    - cell learns probability of channels usage of neighbours
    - cell sense and selects from unused / least used channels
- here: no communication between cells, only via common environment

➔ Different "implementations" with different abilities

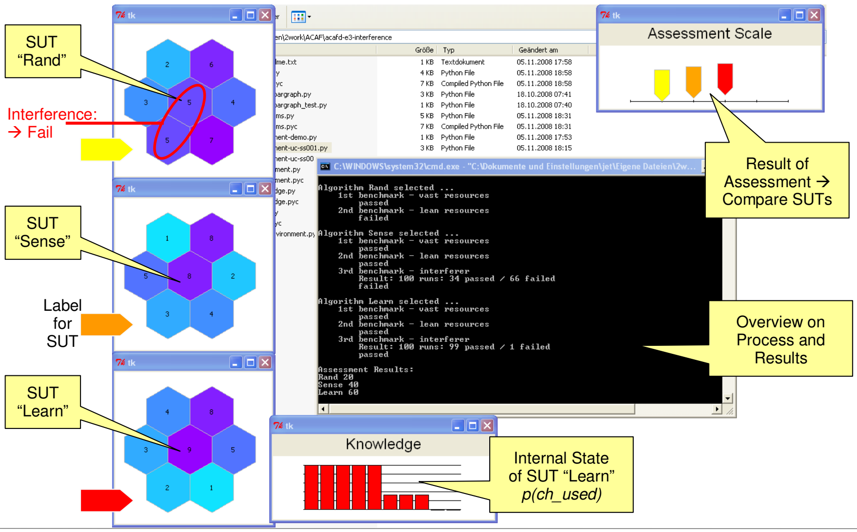
Assessment Example: **Assessment Process**

- Benchmarks with increasing difficulty
- System "implementations" with different capabilities
- Assessment Process:
  - Benchmark is "Pass"ed with >50% good runs
  - Rate SUT with most difficult benchmark passed

		SUT Rand	SUT Sense	SUT Learn
-	**Benchmark Vast resources**	✓	✓	✓
Complexity ↓	**Benchmark Lean resources**	✗	✓	✓
+	**Benchmark Interferer**	✗	✗	✓

➔ System need to solve problems of increasing difficulty

Assessment Example: **Demo**


SUT "Rand"
Interference: → Fail
SUT "Sense"
Label for SUT
SUT "Learn"
tk
Assessment Scale
Result of Assessment → Compare SUTs
Overview on Process and Results
Knowledge
Internal State of SUT "Learn" *p(ch_used)*

en\2work\ACAF\acafd-e3-interference

	Größe	Typ	Geändert am
lme.txt	1 KB	Textdokument	05.11.2008 17:58
y	4 KB	Python File	05.11.2008 18:58
yc	7 KB	Compiled Python File	05.11.2008 18:58
bargraph.py	3 KB	Python File	18.10.2008 07:41
bargraph_test.py	1 KB	Python File	18.10.2008 07:40
ms.py	5 KB	Python File	05.11.2008 18:31
ms.pyc	7 KB	Compiled Python File	05.11.2008 18:31
nent-demo.py	1 KB	Python File	05.11.2008 17:53
nent-uc-ss001.py	3 KB	Python File	05.11.2008 18:15
nent-uc-ss00			
ment.py			
ment.pyc			
dge.py			
dge.pyc			
y			
yc			
vironment.py			

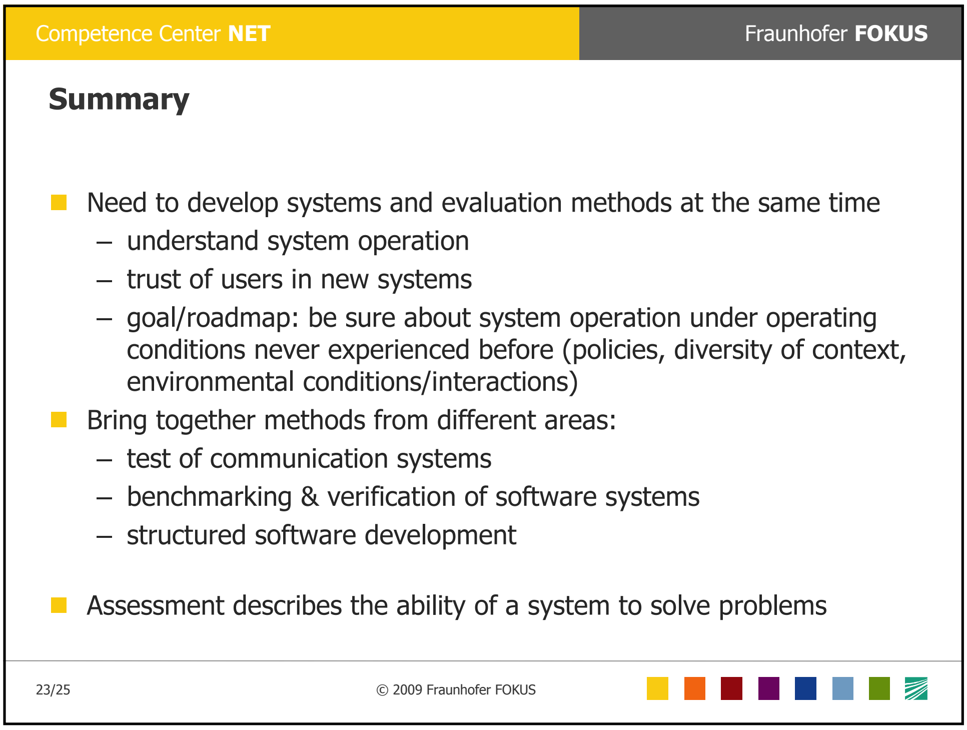
C:\WINDOWS\system32\cmd.exe - "C:\Dokumente und Einstellungen\jet\Eigene Dateien\2w...

```
Algorithm Rand selected ...
    1st benchmark - vast resources
        passed
    2nd benchmark - lean resources
        failed

Algorithm Sense selected ...
    1st benchmark - vast resources
        passed
    2nd benchmark - lean resources
        passed
    3rd benchmark - interferer
        Result: 100 runs: 34 passed / 66 failed
        failed

Algorithm Learn selected ...
    1st benchmark - vast resources
        passed
    2nd benchmark - lean resources
        passed
    3rd benchmark - interferer
        Result: 100 runs: 99 passed / 1 failed
        passed

Assessment Results:
Rand 20
Sense 40
Learn 60
```



Summary

- Need to develop systems and evaluation methods at the same time
  - understand system operation
  - trust of users in new systems
  - goal/roadmap: be sure about system operation under operating conditions never experienced before (policies, diversity of context, environmental conditions/interactions)
- Bring together methods from different areas:
  - test of communication systems
  - benchmarking & verification of software systems
  - structured software development
- Assessment describes the ability of a system to solve problems

future internet technologies by FOKUS

Thank You!

Contact: *jens.tiemann@fokus.fraunhofer.de*

This work was partially performed in project E3 which has received research funding from the Community's Seventh Framework programme.

Fraunhofer FOKUS

Future Internet Tournament

- Have Fun with Future Internet!

  If network elements can manage themselves,
  they can compete with each other:

  Join us to create the first **Future Internet Tournament**

➔ visit http://www.fit-2010.net/CMS

- or attend the afternoon session