



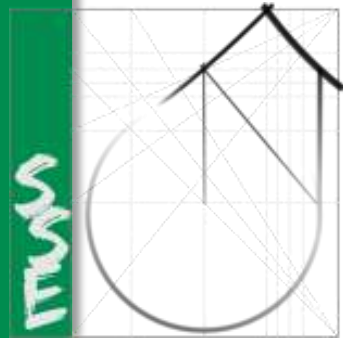
DAiSI – Infrastruktur für Dynamisch Adaptive Systeme

Verlässliche IT Ökosysteme am Beispiel eines
Rettungsassistenzsystems



Dirk Niebuhr

Technische Universität Clausthal
Institut für Informatik - Software Systems Engineering
Lehrstuhl von Prof. Dr. Andreas Rausch
Julius-Albert-Str. 4
38678 Clausthal-Zellerfeld



Dependable Dynamic Adaptive Systems

- In "Classical Systems" space of possible component bindings is defined at development time
- Dynamic System Generation is coming up (cf. Pervasive Computing, Ubiquitous Computing, Organic Computing, ULSS, ...): IT Ecosystems where components may permanently join or leave a system
 - *Flexible* Architectures dealing with these components are required
- ➡ Component binding needs to adapt at runtime
 - We need to be able, to decide, which components are *compatible* at runtime.



Emergency Assistance System

Our Task: Support Helpers during Catastrophes

- Present a quick overview showing
 - Number,
 - Position and
 - Severity of Injuryof casualties
- Monitor vital data of casualties to detect changes in condition and react accordingly
- Enable the Incident Command to instruct helpers efficiently



Emergency Assistance System

System components ...

- ... for the incident command



Emergency Assistance System

System components ...

- ... for helpers



- ... for casualties

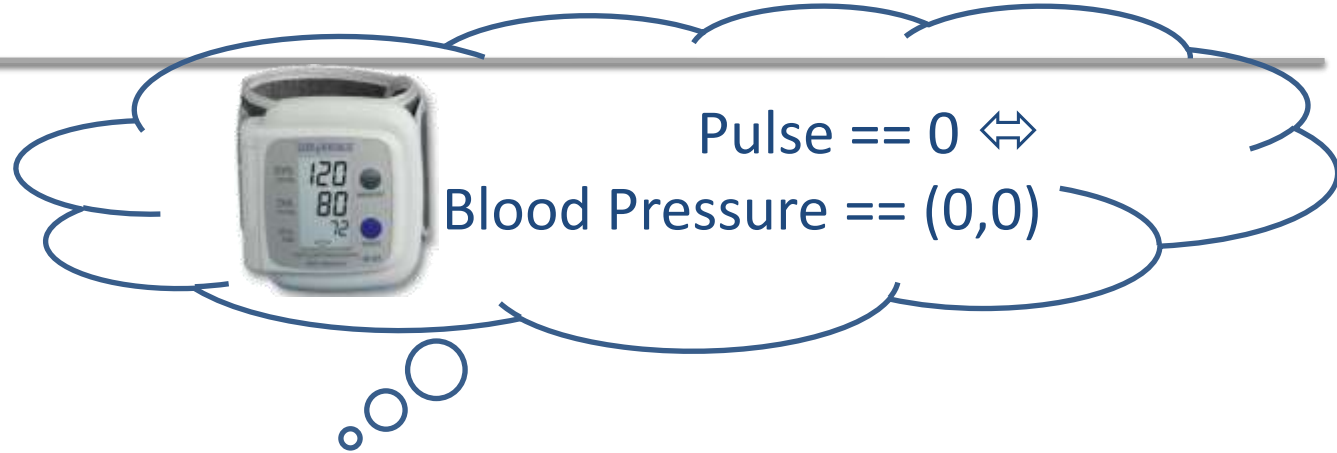


Different vendors!

Emergency Assistance System



**Personal Data
+ Triage History**



Component
Binding
at Runtime

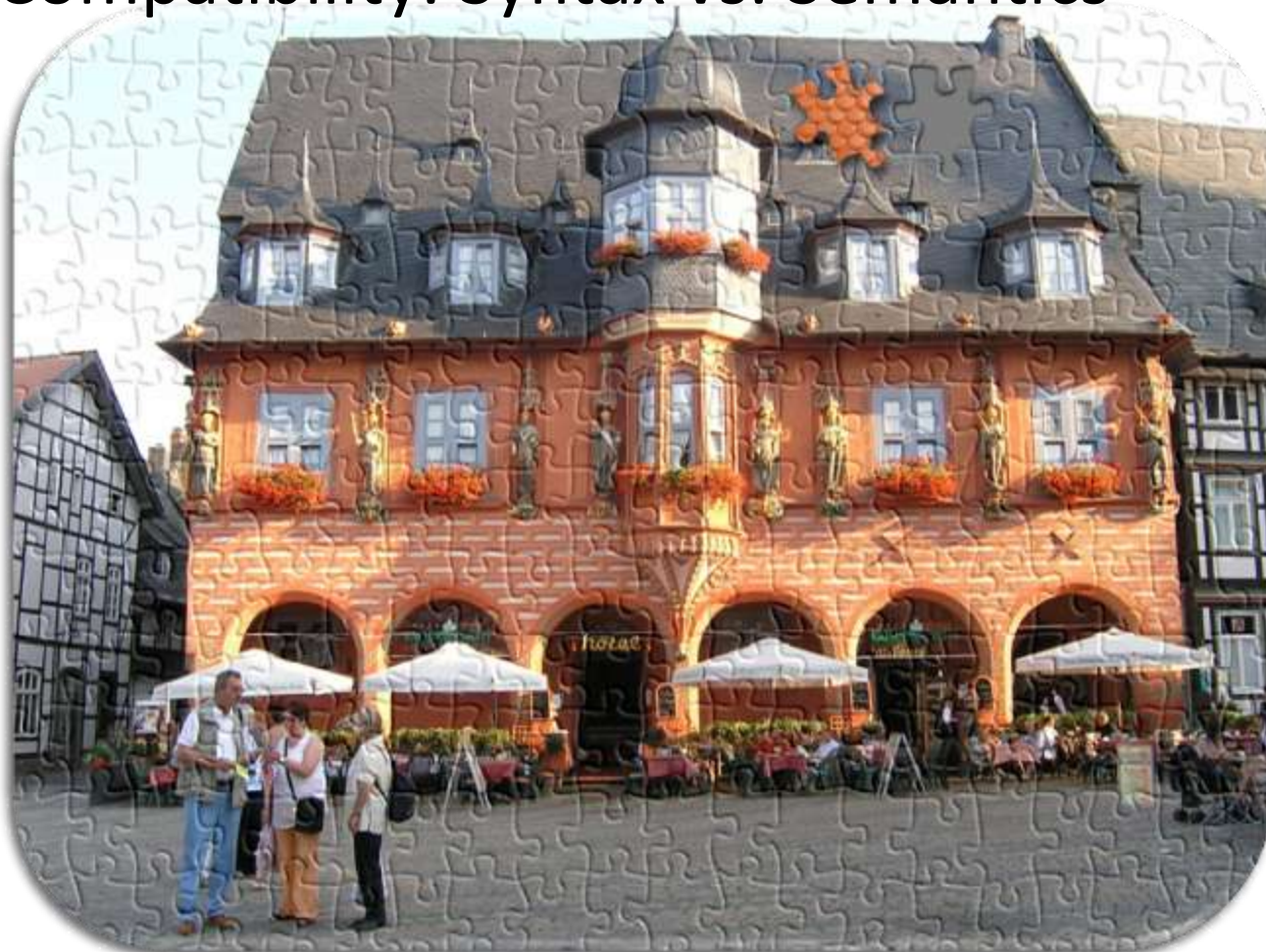
```
Dead: vitalDataMonitoring.getPulse()==0  
...
```



**Vital Data
Monitoring
+ Triage
Calculation**

Dependable Dynamic Adaptive Systems

- **Compatibility: Syntax vs. Semantics**



Dependable Dynamic Adaptive Systems

- Initial Situation: Middleware capable of dynamic component binding based on interface *syntax* existed (Dynamic Adaptive System Infrastructure DAiSI)
- *Semantic* compatibility of components needs to be addressed as well in order to get dependable Systems
 - Different vendors implement components
 - Components are developed at different times and have different lifecycles
 - Combinatorial explosion of possible component bindings within a system
- ➡ Proofing Compatibility at development time does not help
- ➡ Our solution: Use Runtime Testing to decide about compatibility



Dependable Dynamic Adaptive Systems

- When do we need to test?
 - Initial test (binding-time)
 - Test, whether the components are compatible before binding them
 - Compatibility may change over time!
 - ➡ Runtime-Compliance test (runtime)
 - Define equivalence classes for a component binding (i.e. state spaces) where equivalent behavior is assumed
 - Cannot be defined by component user / provider alone, needs to be combined from the user and provider view
 - Test whenever the equivalence class of a component binding changes



Our Approach applied to the example


Personal Data
+ Triage History
User

Equivalence Class Definitions

EC1: `vDM.getPulse()==0 && vDM.getBloodPressure()==(0,0)`
EC2: `vDM.getPulse()==0 && vDM.getBloodPressure()>(0,0) || vDM.getPulse()>0 && vDM.getBloodPressure()==(0,0)`
EC3: `0<vDM.getPulse()<=30 || 0<vDM.getBloodPressure()<=(40,30)`
...

Equivalence Class Combinations @ Runtime

(EC1,ECA), (EC2,ECA), (EC3,ECB), ...

Equivalence Class Definitions

ECA: `getTriage()==Dead`
ECB: `getTriage()==Seriously_Injured`
...

Provider


Vital Data Monitoring
+ Triage Calculation

Dependable Dynamic Adaptive Systems

- Who defines test cases?
 - Service User (he knows, what he requires from a service)
- How do we react on test results?
 - Pass: Bind components / keep them bound
 - Fail: Unbind components / keep them unbound
- What about side-effects of testing?
 - Components directly or indirectly affected by test case execution are notified, when test case execution starts and when it is finished



Our Approach applied to the example


Personal Data
+ Triage History
User

```
Pass(EC1): vDM.getTriage()==Dead  
Pass(EC2): vDM.getTriage()==Unknown  
...
```

**Test Case
Definitions**

Situations
at
Runtime:

✓ Pulse = 80
Blood Pressure = (120,80)

Fingerclip slips off

⚡ Expected: Unknown
Returned: Dead
Pulse = 0
Blood Pressure = (120,80)

```
public TriageClass getTriage(){  
    if (pulse==0) {  
        return Dead;  
    }  
    ...  
}
```

**Component
Realization**

Provider

**Vital Data
Monitoring
+ Triage
Calculation**



Lessons learned

- DAiSI is an infrastructure designed especially for dynamic adaptive systems, where components permanently join or leave a system.
- DAiSI automatically updates the component bindings in these systems.
- Dependability in dynamic adaptive systems is hard to achieve.
- DAiSI executes test cases defined by the application components during system runtime, to decide, whether components are compatible.



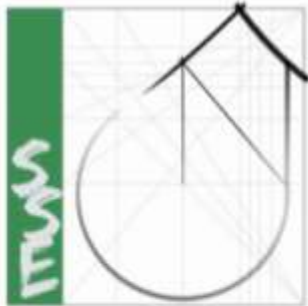
I need more infos!

Further informations...

- ... regarding the Emergency Assistance System
 - <http://www2.in.tu-clausthal.de/Rettungsassistenzsystem>
 - Exhibit at Hall 9, B22.
- ... regarding DAiSI and its specific features
 - Niebuhr, D.; Klus, H.; Anastasopoulos, M.; Koch, J.; Weiß, O.; Rausch, A. 2007. **DAiSI - Dynamic Adaptive System Infrastructure**. Technical Report. <http://publica.fraunhofer.de/starweb/servlet.starweb?path=pub0.web&search=N-58784>
 - Klus, H., Niebuhr, D., and Rausch, A. 2007. **A component model for dynamic adaptive systems**. In International Workshop on Engineering of Software Services For Pervasive Environments: in Conjunction with the 6th ESEC/FSE Joint Meeting (Dubrovnik, Croatia, September 04 - 04, 2007). ESSPE '07.
- ... regarding NTH School for IT Ecosystems:
 - <http://www.it-ecosystems.org/>



Partners



supported by

SIEMENS

SINOSYS



Deutsches Rotes Kreuz 
Rettungsschule Niedersachsen