

# Kontextsensitive Informationsfilter

Entwurf eines Nachrichtensystems für den Aktiven Campus

Diplomarbeit bei  
Professor Gudrun Klinker und  
Professor Donald Kossmann

Christof Söhngen (soehngen@nts-online.de)

# Überblick

1. Einführung
2. Modellierung
3. Implementierung
4. Zusammenfassung

# Überblick

**1. Einführung**

**2. Modellierung**

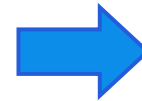
**3. Implementierung**

**4. Zusammenfassung**

# Hauptseminar: Context-based Presentation of Information for Mobile Users

### Szenario:

- Vorhandene Infrastruktur in Gebäuden
- Mobile Benutzer mit eigenen Geräten
- Interessenprofile können erstellt werden
- Zugriff auf Dienstleistungen



**Aktiver Campus**

### Themen:

- Mobile Kommunikation  
(*Protokolle, WaveLan, Bluetooth*)
- Datenbanken  
(*Repräsentation und Indexierung mobiler Objekte, Spatial Data Mining*)
- Augmented Reality  
(*Active Badges, Multimediale Datenbeschreibungen, Informationsfilter, Wearable Computing, DWARF*)

# Überblick

## 1. Einführung

## 2. Modellierung

2.1 Szenarien

2.2 Anforderungsanalyse

2.3 Modell und dessen Komponenten

2.4 Abläufe im Modell

2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

## 4. Zusammenfassung

# Überblick

## 1. Einführung

## 2. Modellierung

### 2.1 Szenarien

### 2.2 Anforderungsanalyse

### 2.3 Modell und dessen Komponenten

### 2.4 Abläufe im Modell

### 2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

## 4. Zusammenfassung

### Insgesamt wurden 17 Szenarien entworfen

- Verteilen von elektronischen Dokumenten
- Erinnerung der Studenten an Fristen durch die Verwaltung
- Warnung vor einer Baustelle
- Umleitung bei Überfüllung der Mensa
- Automatische Übermittlung des Mensa-Speiseplans
- Drucken eines Dokumentes auf nahegelegendem Drucker
- Anzeigen aller Bekannten Personen in der näheren Umgebung
- Definition eines Filters
- Information über den Ausfall einer Vorlesung

# Kontextsensitive Informationsfilter

Entwurf eines Nachrichtensystems für den Aktiven Campus

Diplomarbeit bei  
Professor Gudrun Klinker und  
Professor Donald Kossmann

Christof Söhngen (soehngen@nts-online.de)

### Szenario: Ausfall einer Vorlesung

**Szenario:** **benachrichtigungOrt**

**Beschreibung:** Vorlesungsteilnehmer werden über den Ausfall einer Vorlesung informiert

**Akteure:** Professor  
Studenten

**Kontext:** Ort (Nähe zum Hörsaal)  
Stundenplan (Hörer der Vorlesung)  
Historie (Zur Vorlesungszeit im Hörsaal)

**Flow of events:**

1. Ein Professor ist kurzfristig verhindert.
2. Die Studenten, die in seiner Vorlesung sind (Vorlesung im Stundenplan oder mindestens 3 Mal besucht), werden benachrichtigt, dass die Vorlesung ausfällt.
3. Die Personen, die sich im Hörsaal aufhalten, werden auch benachrichtigt.

# Überblick

## 1. Einführung

## 2. Modellierung

### 2.1 Szenarien

### 2.2 Anforderungsanalyse

### 2.3 Modell und dessen Komponenten

### 2.4 Abläufe im Modell

### 2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

## 4. Zusammenfassung

### **Die Kommunikation bzw. Informationsübermittlung ist Nachrichten-basiert**

- Unterstützung aller Teilnehmer des Universitätsbetriebes
- Jede (digital) verfügbare Information kann im System gespeichert werden
- Jeder Teilnehmer kann überall, unabhängig vom verwendeten Gerät am Aktiven Campus teilnehmen
- Informationen und Dienstleistungen werden dem Teilnehmer mit Hilfe von Nachrichten übermittelt
- Die Relevanz einer Nachricht wird vom Ersteller der Nachricht manuell definiert
- Jeder Teilnehmer hat die umfassende Möglichkeit zu entscheiden, welche Nachrichten er empfangen will
- Das System, mit dessen Hilfe die Nachrichten verwaltet werden, kann moderiert werden
- Jeder Teilnehmer kann die Informationen, die im System gespeichert werden, selbst beeinflussen
- Die Struktur der Daten, die im System gespeichert werden ist dynamisch
- Der Sender einer Nachricht hat die Möglichkeit, Quittungen der erfolgten Sendungen zu erhalten
- Der Nachrichten-Inhalt kann dynamisch sein, und auf Informationen auf verteilten Quellen verweisen

# Überblick

## 1. Einführung

## 2. Modellierung

2.1 Szenarien

2.2 Anforderungsanalyse

2.3 Modell und dessen Komponenten

2.4 Abläufe im Modell

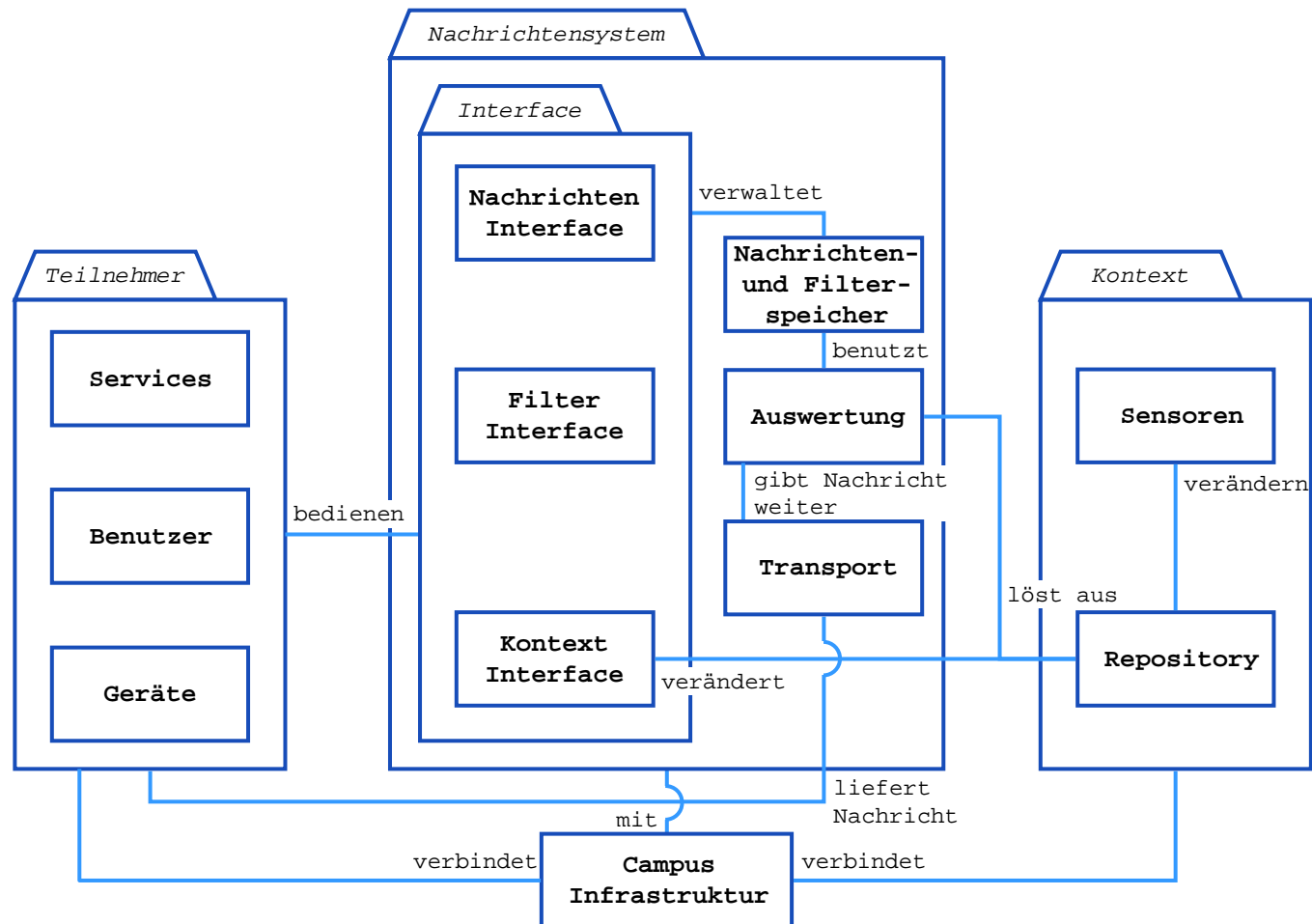
2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

## 4. Zusammenfassung



## Das Nachrichtensystem vermittelt zwischen den Teilnehmern und dem Kontext



# Überblick

## 1. Einführung

## 2. Modellierung

2.1 Szenarien

2.2 Anforderungsanalyse

2.3 Modell und dessen Komponenten

2.4 Abläufe im Modell

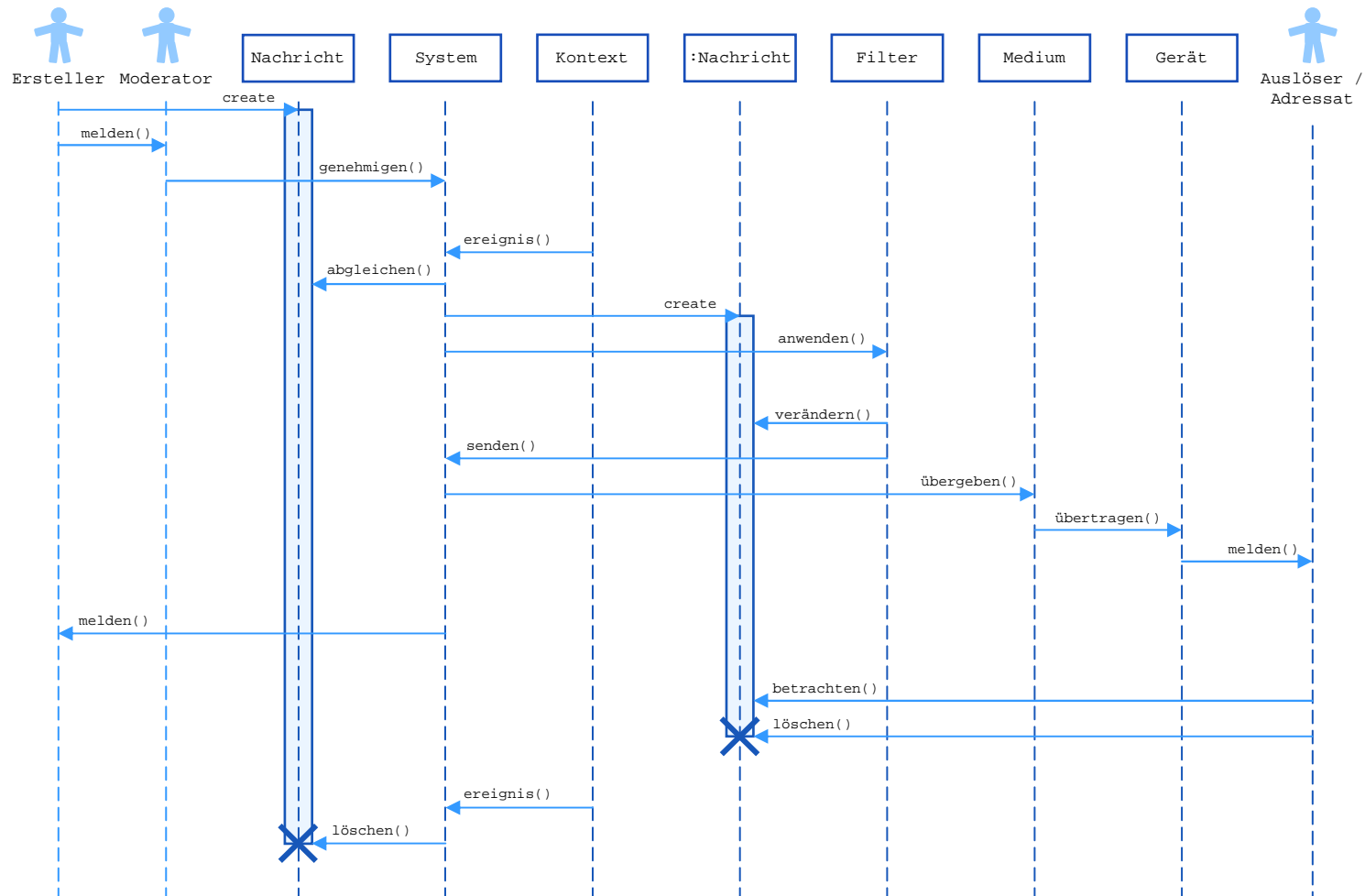
2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

## 4. Zusammenfassung



## Eine Nachricht wird durch ein weiteres passendes Ereignis gelöscht



# Überblick

## 1. Einführung

## 2. Modellierung

2.1 Szenarien

2.2 Anforderungsanalyse

2.3 Modell und dessen Komponenten

2.4 Abläufe im Modell

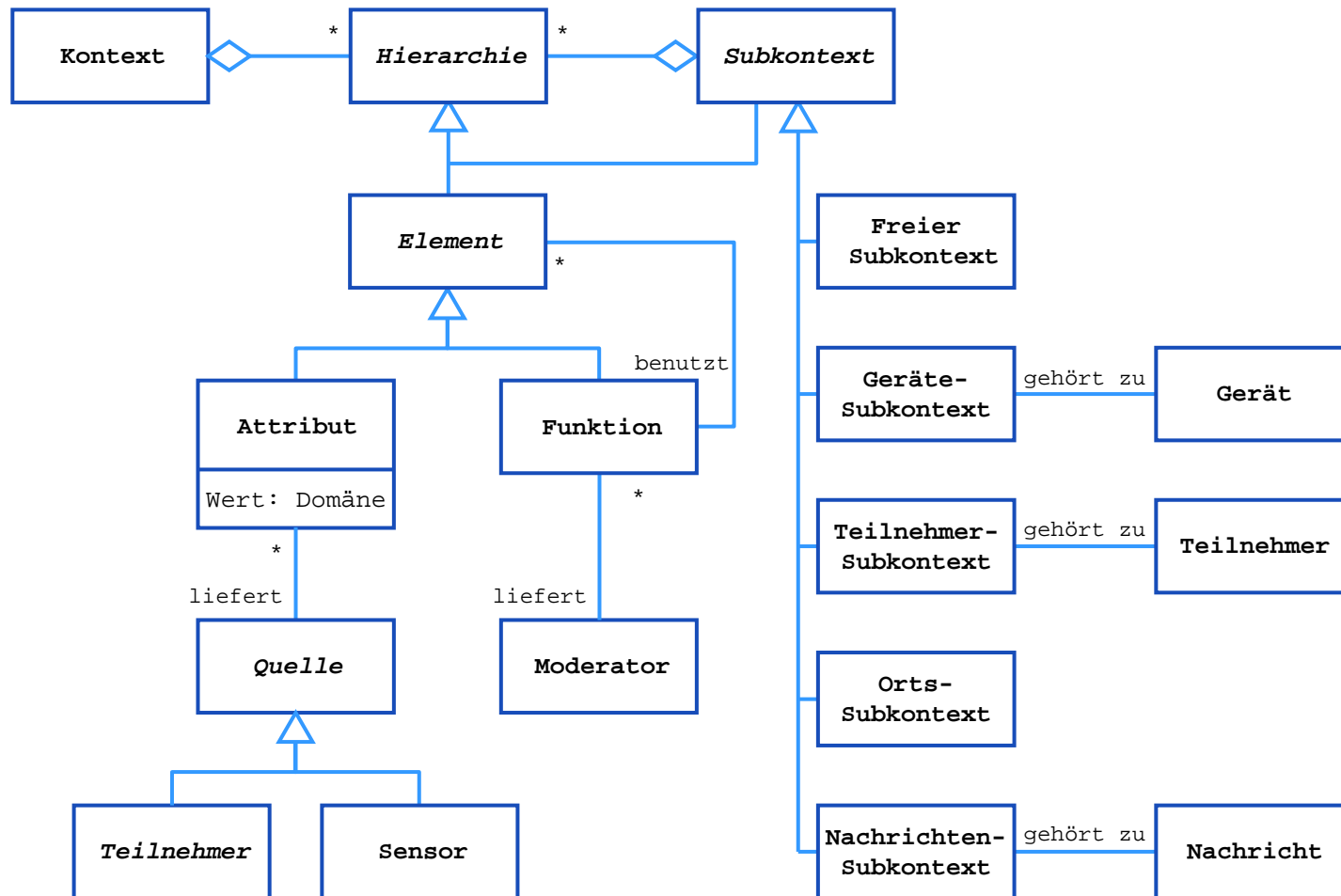
2.5 Details der Komponenten *Kontext*, *Nachricht*, und *Filter*

## 3. Implementierung

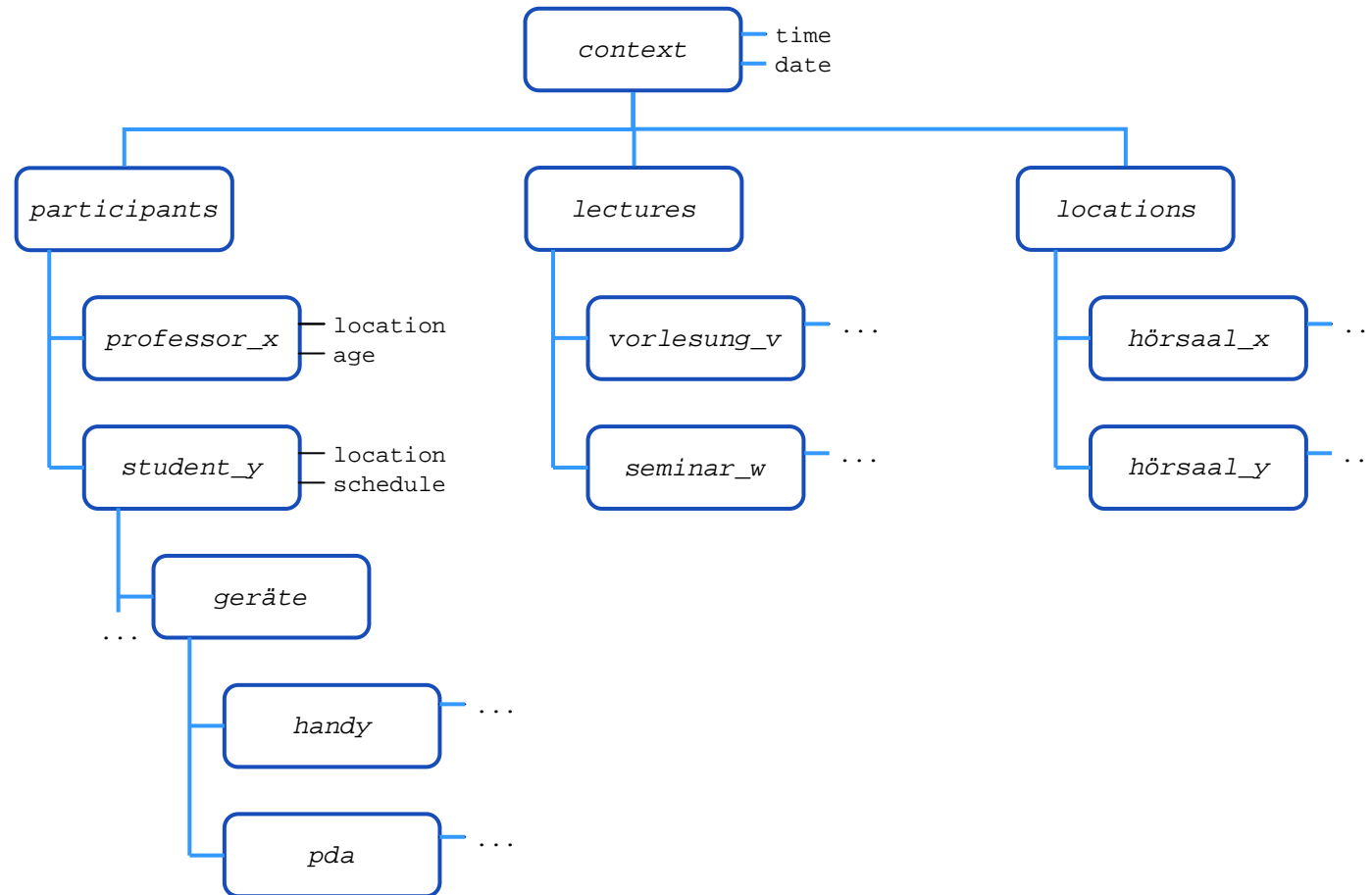
## 4. Zusammenfassung



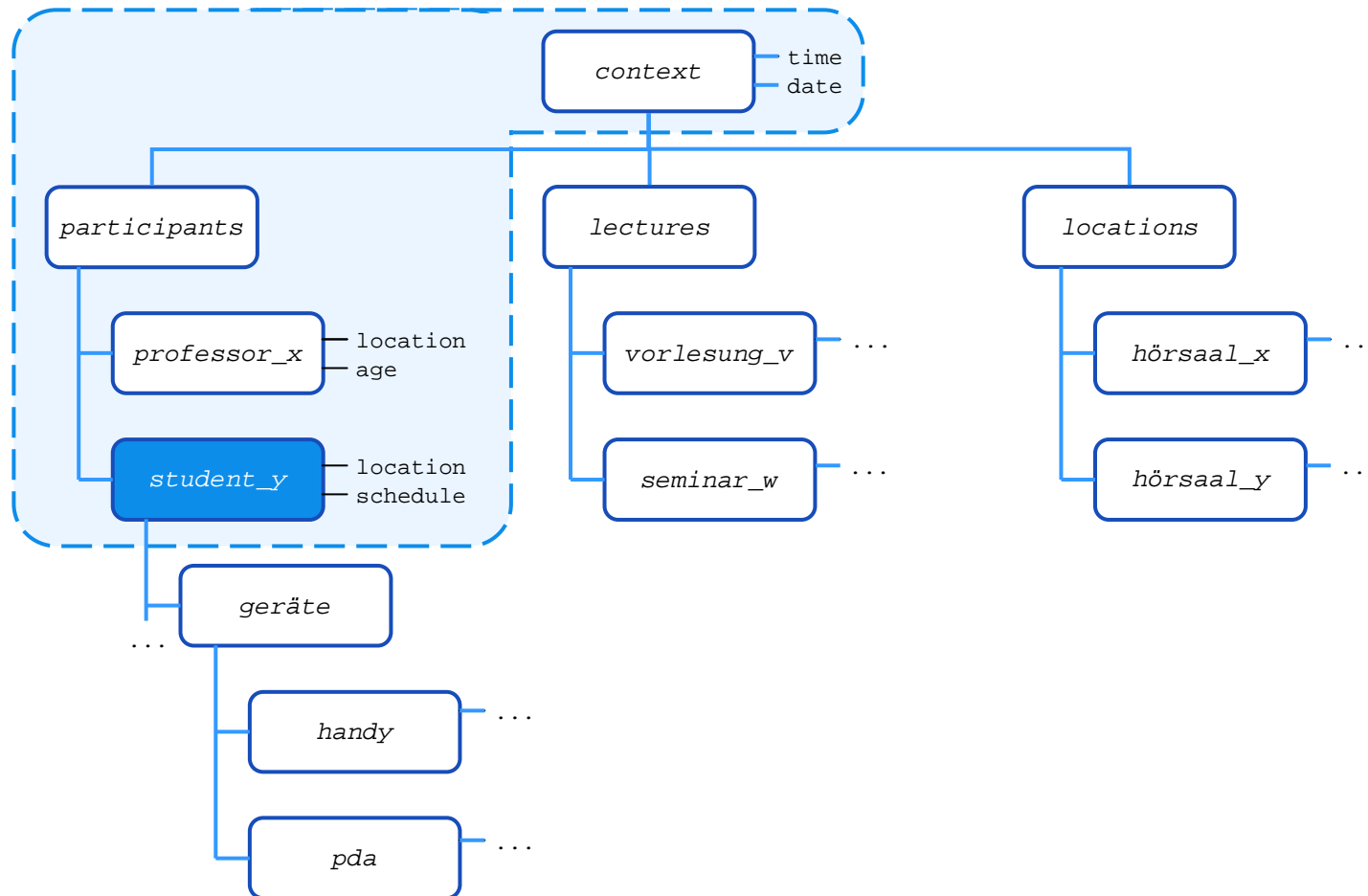
Der Kontext speichert alle Informationen, die im Aktiven Campus verfügbar sind



### Beispiel des Kontexts



### Sichtbarkeit der Subkontexte



### Eine Nachricht speichert eine Kommunikations- / Informationswunsch

```
<message> ::=  
  <condition>  
  <lifetime>  
  <iteration>  
  { <content> <impact> }+  
  <addressee>  
  <feedback>  
  <originator>
```

### Beispiel für eine Nachricht: Szenario BenachrichtigungOrt

**Szenario:** **benachrichtigungOrt**

**Beschreibung:** Vorlesungsteilnehmer werden über den Ausfall einer Vorlesung informiert

**Akteure:** Professor  
Studenten

**Kontext:** Ort (Nähe zum Hörsaal)  
Stundenplan (Hörer der Vorlesung)  
Historie (Zur Vorlesungszeit im Hörsaal)

**Flow of events:**

1. Ein Professor ist kurzfristig verhindert.
2. Die Studenten, die in seiner Vorlesung sind (Vorlesung im Stundenplan oder mindestens 3 Mal besucht), werden benachrichtigt, dass die Vorlesung ausfällt.
3. Die Personen, die sich im Hörsaal aufhalten, werden auch benachrichtigt.

### Beispiel für eine Nachricht: Szenario BenachrichtigungOrt

**Nachricht:**

*Bedingung:* (context.message.actuator.is\_at\_location(context.locations.hoersaal\_y)) OR  
(context.message.actuator.schedule.contains(context.events.lecture\_y)) OR  
(context.message.actuator.number\_of\_visits(context.events.lecture\_y) GE 3)

*Adressat:* context.message.actuator

*Inhalt:* Vorlesung fällt aus, da Professor X krank ist.

*Wirkung:* DISPLAY

*Lebensdauer:* context.functions.distance(context.events.lecture\_x.endtime(context.date), context.time) LE 5min

*Wiederholung:* false

*Rückmeldung:* COUNTER

**Anmerkungen:** Die besonderen Funktionen müssen vorher im Kontext definiert werden.

### Ein Filter speichert einen Wunsch auf Kontrolle über Informationen

```
<filter> ::= order name <decay> <condition> <modifications>
```

```
<modification> ::= <weight> (
```

```
  <modifyDelivery> |
```

```
  <modifyAddressee> |
```

```
  <modifyFeedback> |
```

```
  <modifyImpact> |
```

```
  <modifyIteration> |
```

```
  <modifyContent> )
```

### Beispiel für einen Filter: Szenario eigenerFilter

**Szenario:** **eigenerFilter**

**Beschreibung:** Student definiert Filter

**Akteure:** Student  
Service

**Kontext:** Ort (Nähe zum Hörsaal)

**Flow of events:**

1. Ein Student betritt einen Hörsaal.
2. Sein mobiles Gerät zeigt ab sofort die Nachrichten nur noch mit einem grafischen Symbol, nicht mehr akustisch.

**Filter:**

*Name:* Keine Störung während Vorlesung

*Verfall:* false

*Bedingung:* `context.message.actuator.is_at_location(context.groups.public.locations.hoersaele)`

*Modifikationen:* `context.message.impact = MIN(context.message.impact, STORE)`

**Anmerkungen:** Hörsaal muss eine Sammlung von Orten sein, die vorher definiert wurde.

# Überblick

## 1. Einführung

## 2. Modellierung

## 3. Implementierung

3.1 Heuristik

3.2 Algorithmus

3.3 Weitere Optimierungen

## 4. Zusammenfassung

# Überblick

**1. Einführung**

**2. Modellierung**

**3. Implementierung**

3.1 Heuristik

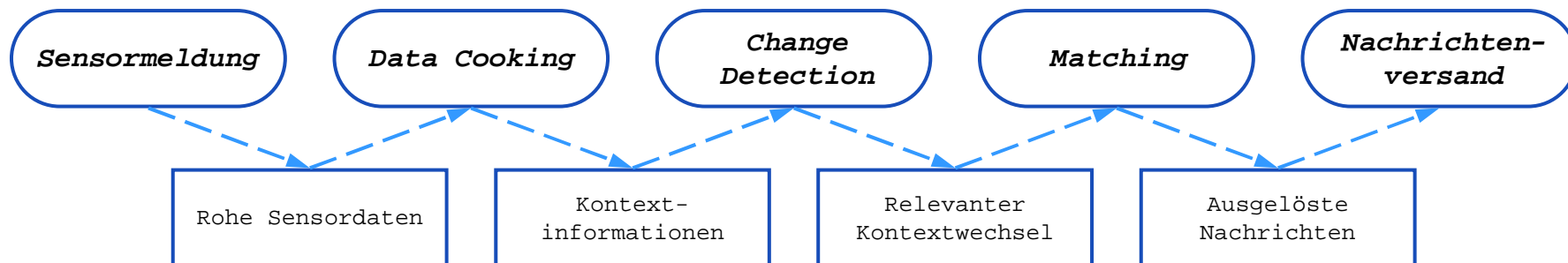
3.2 Algorithmus

3.3 Weitere Optimierungen

**4. Zusammenfassung**



Die ausgelösten Nachrichten werden nach Anwendung aller Filter verschickt



# Überblick

## 1. Einführung

## 2. Modellierung

## 3. Implementierung

### 3.1 Heuristik

### 3.2 Algorithmus

### 3.3 Weitere Optimierungen

## 4. Zusammenfassung



## Hauptidee des Algorithmus ist die Vermeidung von unnötigen Operationen

Nachrichten (bzw. deren Bedingungen):

$$M_0: [P_0 \wedge P_1 \wedge P_2] \vee P_3$$

$$M_1: P_4 \wedge P_5$$

Auslöser:

$$T_0: P_0 \wedge P_1 \wedge P_2$$

$$T_1: P_3$$

$$T_2: P_4 \wedge P_5$$

Auslöser, mit geordneten Prädikaten:

niedrig Update Ratio hoch

$$T_0: \underbrace{P_2 \wedge P_1}_{\text{dashed box}} \wedge P_0$$

$$T_1: \underbrace{P_3}_{\text{dashed box}}$$

$$T_2: \underbrace{P_5 \wedge P_4}_{\text{dashed box}}$$

Prädikate und zugehörige Attribute:

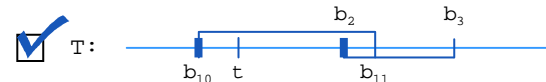
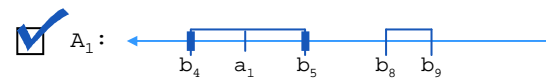
$$P_0: b_0 < A_0 < b_1 \quad P_3: b_6 < A_2 \leq b_7$$

$$P_1: b_2 \leq T < b_3 \quad P_4: b_8 \leq A_1 \leq b_9$$

$$P_2: b_4 < A_1 \leq b_5 \quad P_5: b_{10} < T < b_{11}$$

Attribute:

aktiv



Werte der Prädikate zum Zeitpunkt t:

$$P_0: \text{false} \quad P_1: \text{false} \quad P_2: \text{true}$$

$$P_3: \text{false} \quad P_4: \text{false} \quad P_5: \text{true}$$

# Überblick

## 1. Einführung

## 2. Modellierung

## 3. Implementierung

3.1 Heuristik

3.2 Algorithmus

3.3 Weitere Optimierungen

## 4. Zusammenfassung

## Die Auswertung lässt sich (vor allem durch Expertenwissen) weiter verbessern

- Virtuelle Prädikate/Attribute (Interpolation)

Bsp:

- Temperatur im Winter
- Langer Fußweg

- Virtuelle Grenzen (z.B. Schlüsselworte in Nachrichten/Filtern)

Bsp:

- `message.actuator.location`

# Überblick

**1. Einführung**

**2. Modellierung**

**3. Implementierung**

**4. Zusammenfassung**

4.1 Verbesserungsmöglichkeiten

4.2 Ausblick

# Überblick

**1. Einführung**

**2. Modellierung**

**3. Implementierung**

**4. Zusammenfassung**

4.1 Verbesserungsmöglichkeiten

4.2 Ausblick

### **Zu den vorgestellten Ansätzen bieten sich weitere Vertiefungsmöglichkeiten an**

- Empirische Ermittlungen über Szenarien, Kontext, Geräte und Anwendungen, ...
- Erweiterung der Heuristik für die Sortierung der Attribute
- Berücksichtigung vorhandener Filter in der Auswertung
- Implementierung von Subkontexten mit spezieller Semantik
- Ergänzung der Interval Skip List um eine doppelte Verkettung
- Die Komponenten für Transport und Interface müssen implementiert werden

# Überblick

**1. Einführung**

**2. Modellierung**

**3. Implementierung**

**4. Zusammenfassung**

4.1 Verbesserungsmöglichkeiten

4.2 Ausblick

## Aktuelle Forschungsergebnisse können in den Aktiven Campus einfließen

### Beispiele aus dem Bereich Datenbanken:

- Automatische Nachrichtenerstellung (Data Mining)
- Gemeinsamkeiten im Kontext entdecken (Clustering)
- Aufzeigen von Alternativen (Outlier Detection)

## **Folgende Quellen erlauben einen tieferen Einblick**

- [CBP01] „Hauptseminar Context-based Presentation of Information for Mobile Users “;  
Prof. Dr. Uwe Baumgarten, Prof. Gudrun Klinker Ph. D., Prof. Dr. Donald Kossmann; WS 01/02  
<http://www.bruegge.in.tum.de/teaching/ws01/CBP-Hauptseminar/index.html>
- [Fo77] Charles Forgy, John P. McDermott. International Joint Conference on Artificial Intelligence (IJCAI):  
OPS, A Domain-Independent Production System Language, 1977
- [GoLe99] Andrew R. Golding und Neal Lesh. 3rd International Symposium on Wearable Computing:  
Indoor Navigation using a set of cheap, wearable sensors, 1999
- [Ha02] Florian Haftmann. Bachelor Thesis: Management Of Dynamic User Context, Juli 2002
- [HaJo94] Eric N. Hanson und Theodore Johnson. Selection Predicate Indexing for Active Databases  
Using Interval Skip Lists, April 1994
- [Soe03] „Kontextsensitive Informationsfilter“, Diplomarbeit; 15.02.2003; Christof Söhngen

# Intervall Skip List: Beispiel

a: [ 2,17 ]  
b: ] 17,20 ]  
c: [ 8,12 ]  
d: [ 7,7 ]  
e: [ -∞,17 [

