

# Crowd Sensing und Social Data Mining mit Mobilen Agenten

Labor mit verteilten mobilen Netzwerken

PD Stefan Bosse

Universität Bremen - Fachbereich Mathematik und Informatik

SS2020

2020-07-23

sbosse@uni-bremen.de

## 1. Inhalt

<b>1. Inhalt</b>	4
<b>2. Überblick</b>	4
2.1. Der Kurs . . . . .	4
2.2. Zusammenfassung . . . . .	5
2.3. Grunddaten . . . . .	5
2.4. Zielgruppen des Kurses . . . . .	6
2.5. Materialien . . . . .	6
2.6. Inhalte und Themen . . . . .	7
2.7. Die Systemarchitektur . . . . .	7
2.8. Literatur . . . . .	9
2.9. Ziele . . . . .	13
2.10. Konzepte . . . . .	13
2.11. Forschungsaufgabe . . . . .	17
2.12. Hausaufgabe . . . . .	18
<b>3. Einführung in das Data Mining</b>	18
3.1. Begriffe . . . . .	19
3.2. Das Konzept . . . . .	20
<b>4. Data Mining</b>	24
4.1. Data Mining - Grundlagen . . . . .	25
4.2. Data Mining - Datenwarenhäuser . . . . .	27
4.3. Data Mining Funktionen . . . . .	28
4.4. Data Mining - Datenverarbeitung . . . . .	29
4.5. Data Mining - Datenvisualisierung . . . . .	40
4.6. Smart Environments . . . . .	41
<b>5. Crowdsensing</b>	44

5.1. Bedeutung und Historie . . . . .	45
5.2. Begriffe . . . . .	45
5.3. Crowdsourcing in der Produktion . . . . .	47
5.4. Crowdsensing Systemmodell . . . . .	48
5.5. Belohnung . . . . .	53
5.6. Mobiles Crowdsensing . . . . .	54
5.7. Klassifikation von Crowdsensing . . . . .	61
5.8. Semantische Modellierung von Mobilen Sensoren . . . . .	63
5.9. Semantische Modellierung von Trajektorien . . . . .	66
5.10. Semantische Modellierung von Aktivitäten . . . . .	71
<b>6. Introduction to Augmented Virtuality</b>	<b>72</b>
6.1. Simulation of Socio-Technical Systems . . . . .	72
6.2. Augmented Virtuality . . . . .	73
6.3. Challenges . . . . .	75
<b>7. Umfragen und Methoden</b>	<b>75</b>
7.1. Ziele . . . . .	76
7.2. Umfragengestaltung . . . . .	76
7.3. Die Zielpopulation . . . . .	77
7.4. Die Ziel- und Hilfsvariablen . . . . .	77
7.5. Populationsparameter . . . . .	78
7.6. Erhebungsfenster . . . . .	78
7.7. Störgrößen und Störfaktoren . . . . .	80
7.8. Repräsentative Umfragen . . . . .	80
7.9. Randomisierung . . . . .	82
7.10. Sequenzen . . . . .	86
7.11. Stichproben . . . . .	87
7.12. Schätzung . . . . .	89
<b>8. Gestaltung von Umfragen - Kochrezepte</b>	<b>90</b>
8.1. Die Umfrage . . . . .	91
8.2. Die Fragen . . . . .	91
8.3. Antworten . . . . .	92
8.4. Der Fragentext . . . . .	93
8.5. Antworttypenklassen (Formularlayout) . . . . .	96
8.6. Fragenkataloge . . . . .	97
8.7. Auswahl der Stichprobe . . . . .	100
<b>9. Crowdsensing Plattformen</b>	<b>100</b>
9.1. Zentralistische Netzwerke . . . . .	101
9.2. Energieeffizientes Mobiles Crowdsensing . . . . .	104
9.3. Identifikation . . . . .	105
9.4. Implementierungsprinzipien . . . . .	107
9.5. Implementierungsrandbedingungen . . . . .	107
<b>10. Sensorische Architekturen</b>	<b>108</b>
10.1. Sensoraggregation . . . . .	108
10.2. Lokalisation . . . . .	111
<b>11. Skalierbarkeit und Robustheit</b>	<b>111</b>
11.1. Skalierbarkeit . . . . .	112

11.2. Robustheit . . . . .	112
<b>12. Agenten</b>	113
12.1. Motivation . . . . .	113
12.2. Agenten . . . . .	114
12.3. ATG Modell . . . . .	117
12.4. DATG Modell . . . . .	119
12.5. Beispiel eines Agenten . . . . .	120
12.6. Agentenklassen . . . . .	121
12.7. Tupelräume . . . . .	123
12.8. Tupelräume - Datenmodell . . . . .	124
12.9. Tupelräume - Operationale Semantik . . . . .	125
12.10. Tupelräume - Produzenten und Konsumenten . . . . .	128
12.11. Signale . . . . .	128
12.12. Mobile Agenten . . . . .	130
<b>13. Agentenplattformen</b>	131
13.1. JavaScript Agent Machine . . . . .	131
<b>14. Programmierung</b>	137
14.1. JavaScript :: Daten und Variablen . . . . .	137
14.2. JavaScript :: Funktionen . . . . .	138
14.3. JavaScript :: Datenstrukturen . . . . .	138
14.4. JavaScript :: Objekte . . . . .	139
14.5. AgentJS . . . . .	140
14.6. Simulation . . . . .	147
<b>15. Demonstrator</b>	149
15.1. Smart City: Self-Organizing Light Control . . . . .	150
15.2. Conclusions . . . . .	151
<b>16. Sensoren</b>	152
16.1. Das IoT und Crowdsensing . . . . .	152
16.2. IoT Sensorknoten . . . . .	153
16.3. Intelligent Large Scale Sensing . . . . .	154
16.4. Der Nutzer als Sensor . . . . .	158
16.5. WEB basierte Dienste . . . . .	163
16.6. Agentenbasierter Dienst . . . . .	164
16.7. Dialoge . . . . .	167
16.8. Physische Sensoren . . . . .	168
16.9. Virtuelle Sensoren . . . . .	169
16.10. Aggregation und Fusion . . . . .	171
16.11. Aggregation und Maschinelles Lernen . . . . .	173
<b>17. Organisationsstrukturen</b>	173
17.1. Netzwerke . . . . .	174
17.2. Räumliche Nähe . . . . .	175
<b>18. Sicherheit</b>	175
18.1. Ebenen der Sicherheit . . . . .	175
18.2. Sicherheitsrisiken . . . . .	176
18.3. Multiagentsysteme und Plattformen . . . . .	176
18.4. Schutz privater Daten . . . . .	177

18.5. Explorationsstrategien mit MAS . . . . .	180
<b>19. Literatur und Referenzen</b> . . . . .	180
19.1. Bücher . . . . .	181
19.2. Publikationen . . . . .	181
19.3. Folien . . . . .	182

## 2. Überblick

### 2.1. Der Kurs

- Die Veranstaltung folgt dem **Konzept des forschenden Lernens** und bietet eine praktische und experimentelle Einführung in das Themenfeld Crowd Sensing und Social Data Mining mit Mobilen Agenten.
- In dieser Veranstaltung sollen Studenten den praktischen Umgang und die **Umsetzung** von verteilten Crowdsensing und Social Data Mining Anwendungen an einfachen Beispielen erlernen.
- Als verteiltes Datenverarbeitungs- und Kommunikationsmodell werden **mobile Agenten** eingesetzt.
- Die verwendete **JavaScript** basierte **Ausführungsplattform JAM** lässt sich auf einer Vielzahl von Hostplattformen ausführen, wie mobile Geräte (Smartphones), Desktoprechner, Server, Eingebettete Rechner und Sensornetzwerke, und im Internet mittels eines WEB Browsers.

### 2.2. Zusammenfassung

- Im Vordergrund steht eine
  - ❑ **heterogene verteilte Anwendung** mit
  - ❑ **Nutzerinteraktion** (HMI), das
  - ❑ **Agentenmodell** ist hier nur ein Werkzeug und dient als “Elementarzelle” eines selbstorganisierenden Systems, und
  - ❑ **Datenanalyse** (mittels *R*)
- Es ist daher nur Basiswissen von Agenten und verteilten Systemen erforderlich, die in der Veranstaltung noch einmal kurz zusammengefasst werden.
- Es erfolgt eine Einführung in die
  - ❑ Erhebung von Nutzerdaten (Umfragen),

- ❑ Grundlagen des mobilen Crowdsensing, der
- ❑ Probleme (wie motiviere ich Nutzer teilzunehmen usw.), der
- ❑ Sicherheit und Umgang mit privaten Daten, und der
- ❑ Aggregation und Analyse der Daten.

### 2.3. Grunddaten

---

---

VAK	08-29-4-FEM-1-e
Dozent	PD Stefan Bosse, sbosse@uni-bremen.de
Kategorie	Kurs: Vorlesung und integriertes Labor
Umfang	2 SWS
Art	Master / Bachelor Wahlpflicht/Ergänzung
Profil	Soziologie, Informatik, ...
ECTS	6
Leistung	Übungen, Mündliche Prüfung, Bericht
Wann	Sommer Semester 2020, NV
Wo	Online
Info	<a href="http://edu-9.de">http://edu-9.de</a>

---

### 2.4. Zielgruppen des Kurses

- Soziologen
- Informatiker
- Wirtschaftswissenschaftler, Ökonomen
- Produktionstechniker und Logistiker
- Systemingenieure (Systems Engineering)
- Psychologen
- Geologen



[tiridifilm/istockphoto.com]

## 2.5. Materialien

1. Die Vorlesungsinhalte (Skript, Folien) werden auf <http://edu-9.de> unter der Rubrik Lehre zusammengestellt und angeboten
2. Weitere Materialien (Tutorials, Übungen, Software) werden ebenfalls auf <http://edu-9.de> bereitgestellt
3. Die Videos sind über <http://edu-9.de> verlinkt und sind auf <http://ag-0.de> verfügbar (*opencast* Server)
4. Interaktion der Teilnehmer findet über einen Wiki statt! (*dokuwiki*). Dieser ist über <http://ag-0.de> erreichbar und in den jeweiligen Veranstaltungsseiten auf <http://edu-9.de> verlinkt.
5. Es wird noch einen online Chat geben.
6. Alle weiteren Hinweise und Einführungen (z.B. in Software) nur noch auf dem Wiki!!!

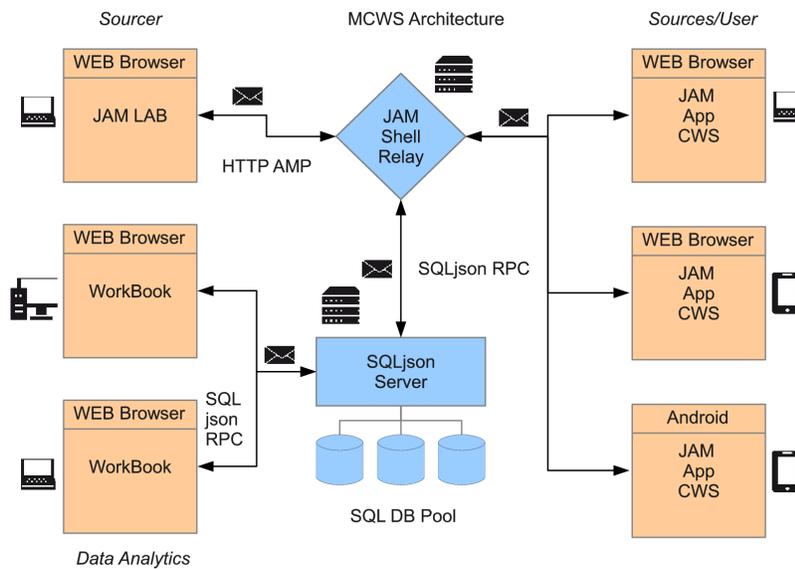
## 2.6. Inhalte und Themen

1. Sensoren, Digitale Sensordaten, Big Data
2. Data Mining
3. Umfragen
4. Crowdsensing und Crowdsourcing
5. Ein wenig Soziologie (oder doch mehr?) - Soziale Modelle (Netzwerke)
6. Umfragen und Datenerhebung
7. Datenanalyse
8. Mobiles Crowdsensing, Sensoren

9. Mobile Agenten, (Chat) Bots, und Agentenplattformen
10. Organisationsstrukturen
11. Labor und Experiment!!!

## 2.7. Die Systemarchitektur

### *Mobiles Crowd Sensing mit Agenten*

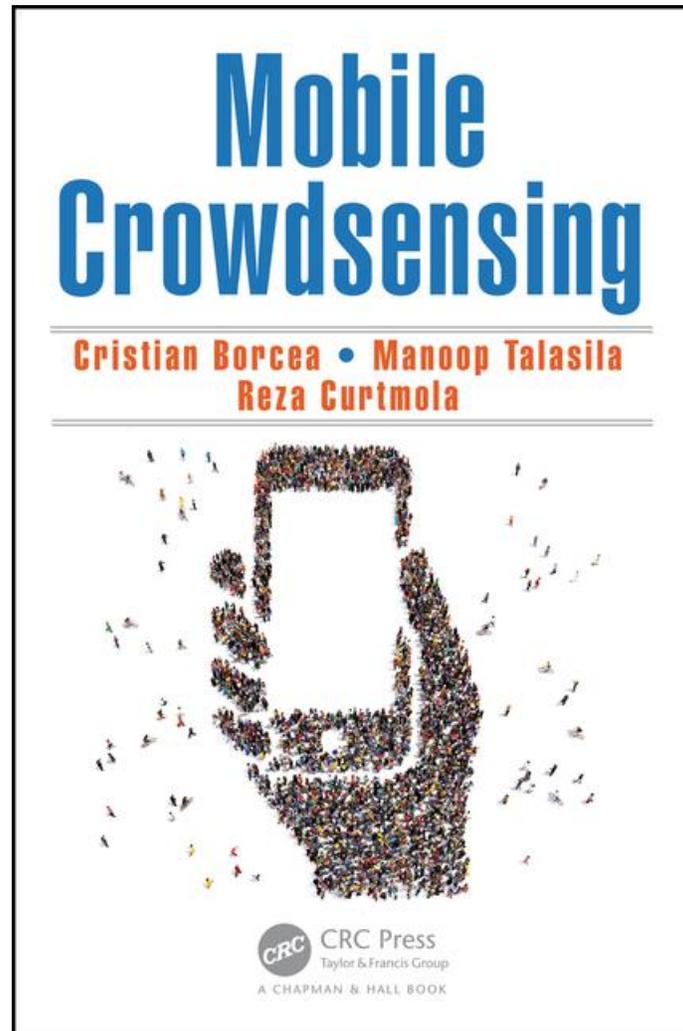


**Figure 1.** (Links) Datenerheber (Mitte) Vermittlung (Rechts) Nutzer und Datenquellen

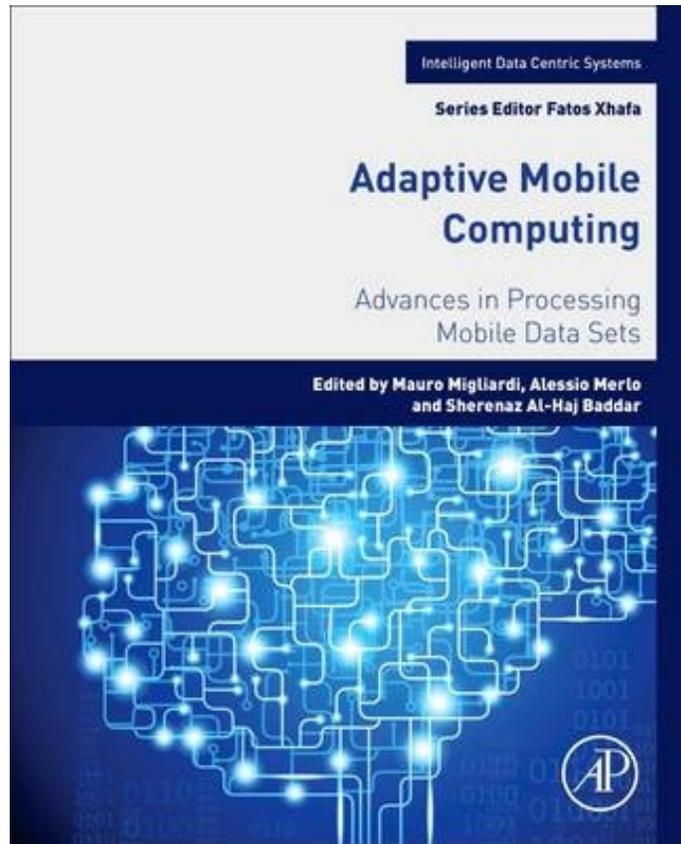
## 2.8. Literatur

- Empfohlene Literatur (neben Vorlesungsskript) zur Vertiefung

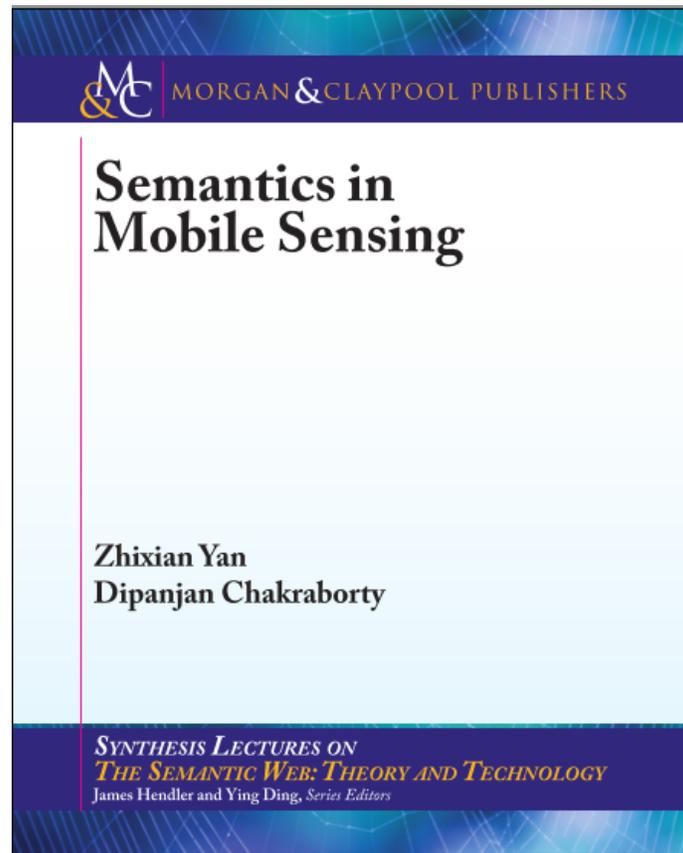
C. Borcea, M. Talasila, and R. Curtmola, *Mobile Crowdsensing*. CRC Press, 2017.



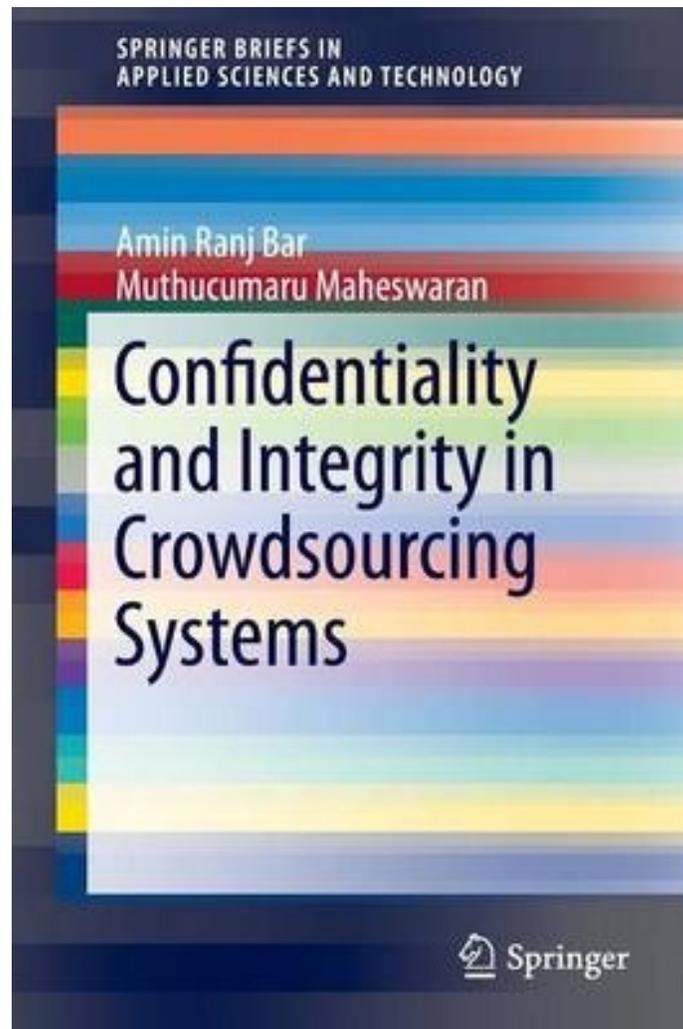
M. Migliardi, A. Merlo, and S. A.-H. Baddar, Eds., Adaptive Mobile Computing. Elsevier, 2017.



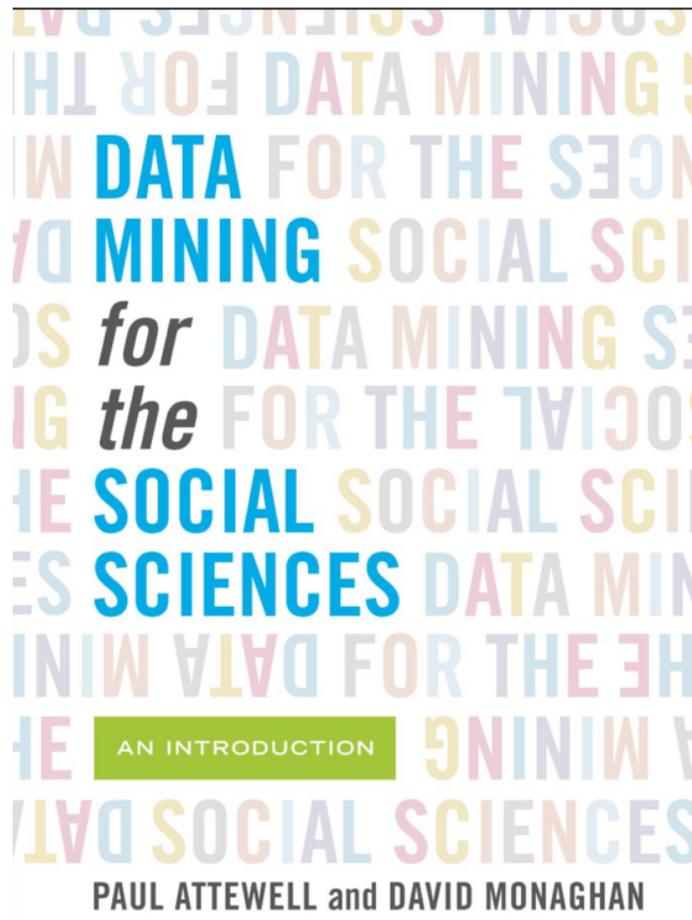
Z. Yan and D. Chakraborty, Semantics in Mobile Sensing. MORGAN&CLAYPOOL, 2014



A. Bar, M. Maheswaran, Confidentiality and Integrity in Crowdsourcing Systems, Springer, 2013



P. Attewell and D. B. Monaghan, Data mining for the social sciences : an introduction. University of California Press, 2015.



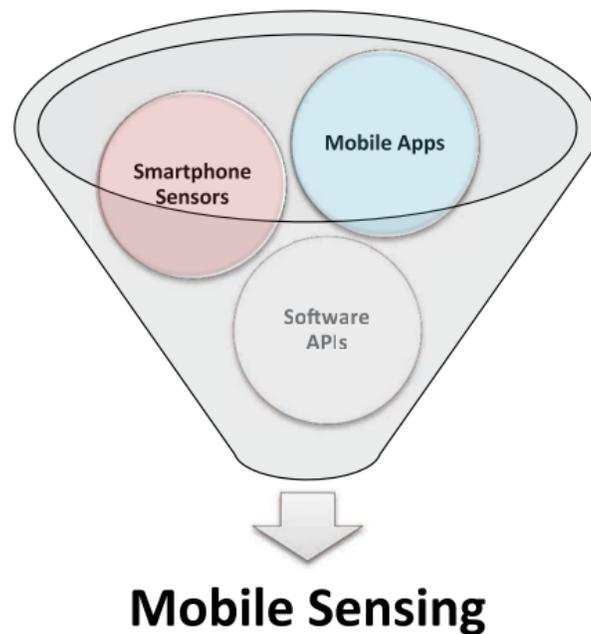
## 2.9. Ziele

1. Grundverständnis von verteilten und mobilen Plattformen für das Crowdsensing und Sourcing erwerben
  - um wesentliche Fragestellungen beim Entwurf und der Nutzung dieser Plattformen für großskalige und teilweise anonymisierte und mobile Cloud Netzwerke beantworten zu können.
2. Praktisch sollen Kenntnisse vom Einsatz (und ggfs. Programmierfertigkeiten) von agentenbasierten Crowdsensing mit einem Laborprototyp erworben werden
3. Es soll interdisziplinäres Arbeiten und Forschen erlernt werden und ausgewählte Fragestellungen der Soziologie erarbeitet werden

4. Verständnis und Anwendung von Sensoren, virtuellen Sensoren und Aggregation, selbstorganisierenden Systemen in einem ganzheitlichen Ansatz
5. Zentrale Fragen der Sicherheit, Schutz privater Daten, und Herausforderungen in solchen verteilten Systemen sollen erkannt werden können.

## 2.10. Konzepte

1. Umfragen - klassisch
2. Crowd Sensing und Mobiles Crowd Sensing → Umfragen mittels Crowd Sensing
3. Agenten und Chat Bots als Werkzeuge für die Erheber-Nutzer Schnittstelle
4. Datenanalyse (Statistik)



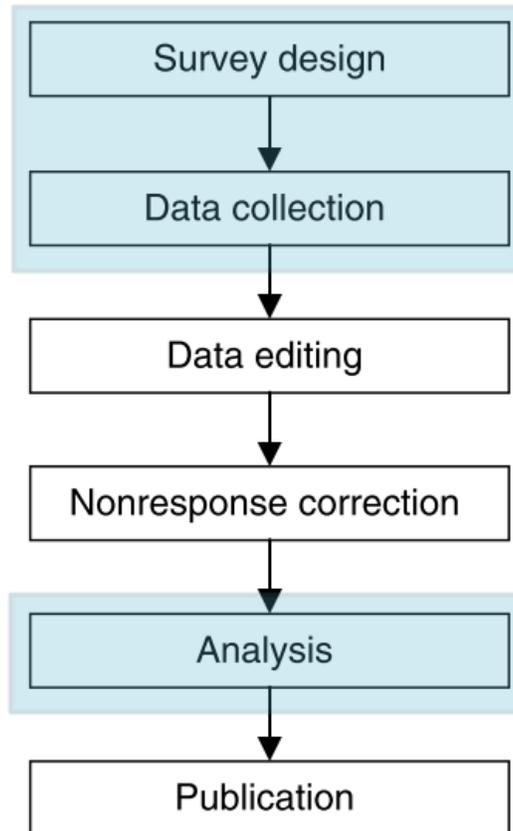
[Borcea, 2017]

### **Befragung (Umfrage, Survey)**

*“Befragung, eine häufig verwandte und universell ein- setzbare Erhebungsmethode, bei der Fragende und Befragte über unterschiedliche Medien in Austausch treten, um durch Fragen oder Aufforderung zu Stellungnahmen bzw. Erzählungen Informationen über den Befragten oder über andere Sachverhalte zu gewinnen.” [Methoden-Lexikon für die Sozialwissenschaften]*

Welche Themen sind dabei relevant?

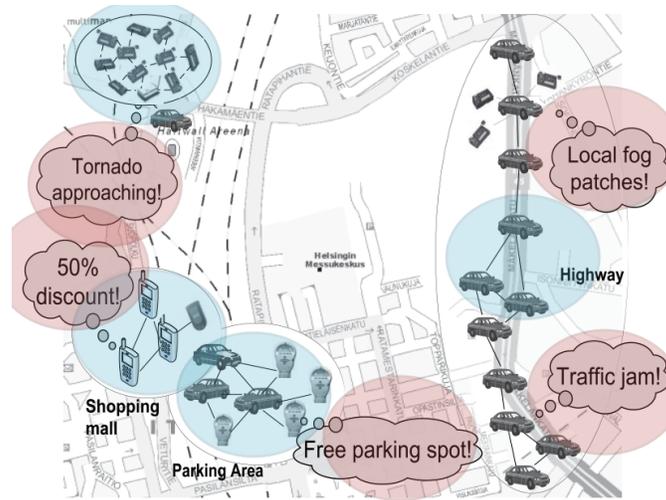
- Befangenheit → Neutralität
- Methodik:
  - ❑ Standardisierte und
  - ❑ offene (nicht-standardisierte, qualitative) Befragungen
- Konstellationen von Fragenden und Befragten
- Soziale und interaktive Beziehungen zwischen Fragenden und Befragten
- Erhebungsprobleme (jeglicher Art)
- Entwicklung von Befragungen (dynamisch vs. statisch)
- Theoretische Grundlagen der und Modelle als Grundlage für die Befragung



Ablauf von Umfragen - die blau markierten Ebenen werden hier behandelt [Bethlehem, 2009]

### ***Mobiles Crowd Sensing***

*Nutzer sind sowohl Quellen von Daten als auch Konsumenten der Daten bzw. daraus extrahierter Informationen!*



Smart City – Blau: Datenquelle, Rot: Datensenke [Borcea, 2017]

### Herausforderungen und Fragestellungen

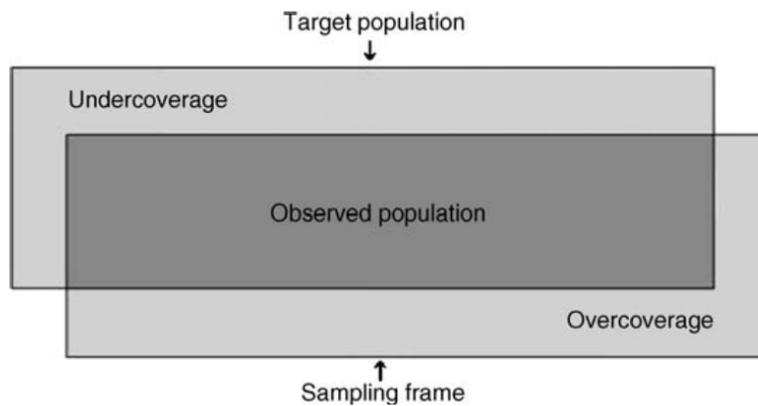
1. Architektur der Datenerhebung mit Mobilien Crowd Sensing → Zuverlässigkeit, Erreichbarkeit, Datensicherheit, Sicherheit bei der Datenverwendung, Ressourcenbedarf, Skalierbarkeit ...
2. Gestaltung von mobilen Umfragen und Repräsentanz → Verzerrung und Bias:

<b>Total participants</b>	58
<b>Males</b>	90%
<b>Females</b>	10%
<b>Age 16–20</b>	52%
<b>Age 21–25</b>	41%
<b>Age 26–35</b>	7%

**Figure 2.** Die Populationsverteilung einer mit einer mobilen App (McSense) durchgeführten Feldstudie [Borcea, 2017]

3. Qualität der Daten und Fusion von menschlichen mit technischen Daten (z.B. Smartphone Position)
4. Betrug und Verfälschung
5. Motivation an der Umfrage und Erhebung teilzunehmen!

6. Unter- und Überabdeckung bei Umfragen (Samplingframe)



**Figure 3.** Samplingframe: Problem der Unter- und Überabdeckung bei einer Umfrage [Bethlehem, 2009]

## 2.11. Forschungsaufgabe

In diesem Kurs/Seminar soll von den Studenten in Gruppen folgende Fragen und Aufgaben bearbeitet werden:

1. Auswahl geeigneter Umfrageszenarien die sinnvoll (im Sinne der Soziologie) mit Mobilen Geräten und Crowd Sensing umgesetzt werden können.  
⇒ **Ich benötige Eure soziologische Expertise dafür!!!**
2. Erstellung einiger konkreter dynamischer Umfragen (d.h. Umfragebäume) die zwei Erhebungsstrategien umsetzen sollen:
  - Partizipatorische Umfrage (Zielgruppengerichtet)
  - Opportunistische Umfrage  
⇒ **Ich benötige Eure soziologische Expertise dafür!!!**
3. Nutzung, Test (!!!) und Evaluierung des Agentenframeworks für das Mobile Crowd Sensing
4. Auswertung der Umfragen mit statistischen und soziologischen Methoden  
⇒ **Ich benötige Eure soziologische Expertise dafür!!!**

## 2.12. Hausaufgabe

### **Aufgabe**

Führe eine Internetrecherche (allgemein und wissenschaftlich, z.B. mit Google Scholar) nach folgenden Fragestellungen durch:

1. Welche Anwendungsgebiete und Szenarien finden sich zum Thema Crowd-sensing und Crowdsourcing?
2. Welche Technologien werden eingesetzt?
3. Wie waren die Fragestellungen oder die Ziele?
4. Wurden Ergebnisse veröffentlicht?
5. Was ist der Unterschied zwischen Crowd Sensing und Crowd Sourcing? Gibt es einen?

*Hinweis: Nutze gängige Suchmaschinen für populistische Inhalte, nutze [ResearchGate](#), [Google Scholar](#) oder [Mendeley](#) für wissenschaftliche Inhalte.*

- Trage die Ergebnisse im Wiki ein!

---

## 3. Einführung in das Data Mining

### 3.1. Begriffe

#### **Sourcing**

Schöpfung von Mehrwert: Quelle von Informationen mit Bereitstellung von Daten; Aber auch Bereitstellung von Arbeitsleistung; Sammlung von Gütern

#### **Sensing**

Wahrnehmung von sensorischen Informationen

#### **Perzeption**

Wahrnehmung und Interpretation von sensorischen Informationen

#### **Sensoren**

Wandler die eine physikalische in eine andere physikalische Größe umwandeln (i.A. elektrisch oder digital)

#### **Virtuelle Sensoren**

Funktionen mit einer Eingangs- und Ausgangsschnittstelle zur Weiterverarbeitung und Fusion von Sensorsignalen

### Umfrage

Neben der Bereitstellung von sensorischen Daten  $\mathbf{X}=x_1,\dots,x_i$  (ggfs. passiv) kann es aktive Teilnahme an Datenerhebung geben i.A. mit einem zu testenden Modell, d.h.,

$$\exists M(\tilde{x}) = \tilde{x} \rightarrow \tilde{y}$$

### Data Mining

Ableitung von Informationen aus Daten (Variablen  $\mathbf{X}=x_1,\dots,x_i$ ) häufig ohne genaue Modelle (statistisch) und Korrelation der Variablen, d.h.:

$$\nexists M(\tilde{x}) = \tilde{x} \rightarrow \tilde{y}$$

### Crowd Sourcing (CWO)

Bereitstellung von Arbeitsleistung durch eine größere Gruppe von Menschen die nicht unmittelbar organisiert (sozial verbunden) sind. Aktive Teilnahme an Umfragen kann dazu gehören (d.h.  $CWS \subset CWO$ )

### Crowd Sensing (CWS)

Erhebung und Erfassung von sensorischen Informationen die durch eine größere Gruppe von Menschen, die nicht unmittelbar organisiert (sozial verbunden) sind, bereit gestellt werden.

### Mobiles Crowd Sensing (MCWS)

CWS mit mobilen Geräten und beweglichen Nutzern  $\rightarrow$  räumliche Veränderung und Ortskontext können eine wichtige Rolle spielen!

## 3.2. Das Konzept

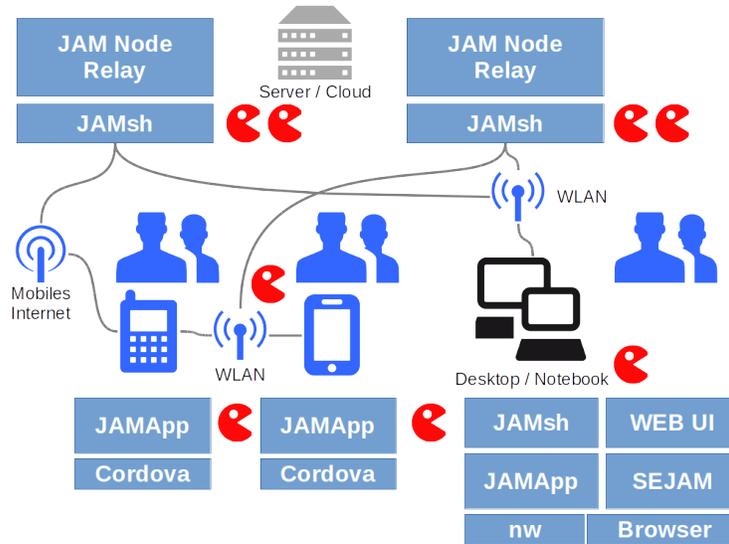
- Die automatisierte Wahrnehmung sensorischer Informationen soll durch **mobile Agenten** implementiert werden
- Diese Agenten können durch eine universelle **Ausführungsplattform** auf einer Vielzahl von Computern und mobile Geräten ausgeführt werden
- Die Plattformen spannen ein virtuelles **Netzwerk** auf, das es den Agenten ermöglicht verschiedene Geräte zu "besuchen"
- Dabei sollen die Agenten mit Menschen und Maschinen interagieren

□ **Mensch-Agent-Maschine Schnittstelle**

□ **Maschine-Agent-Maschine Schnittstelle**

□ **Agent-Agent Schnittstelle**

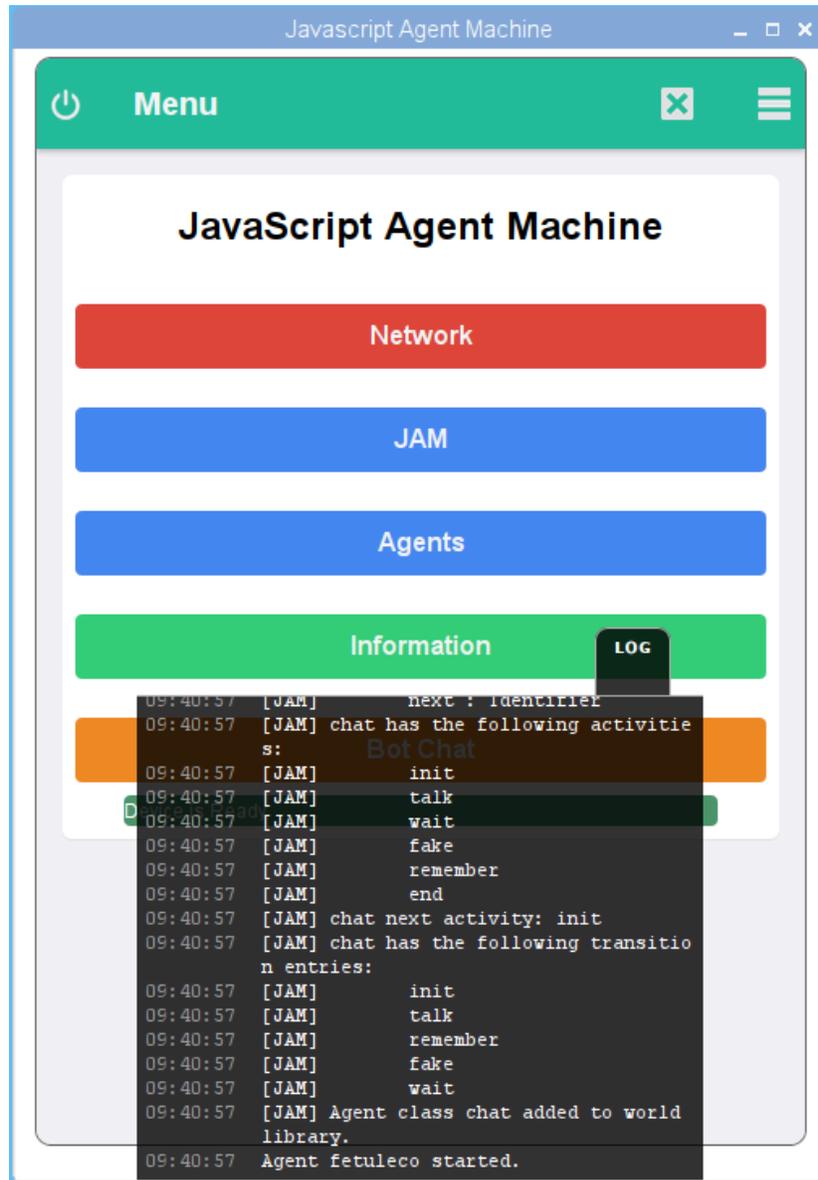
- Die Agenten werden in lose gekoppelten **Multiagentensystemen** durch Divide-and-Conquer und **Selbstorganisation** ihre Aufgaben erfüllen

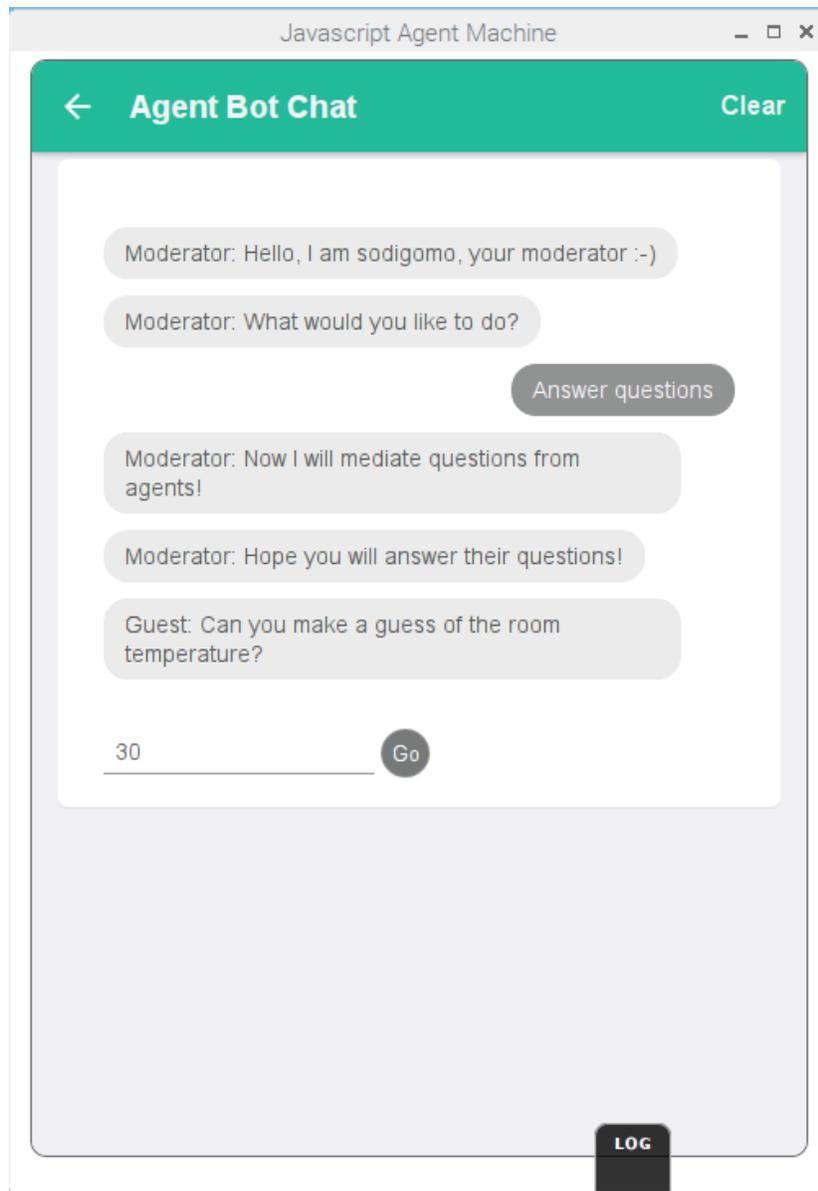


**Figure 4.** Das Konzept mit der JavaScript Agent Machine (JAM)

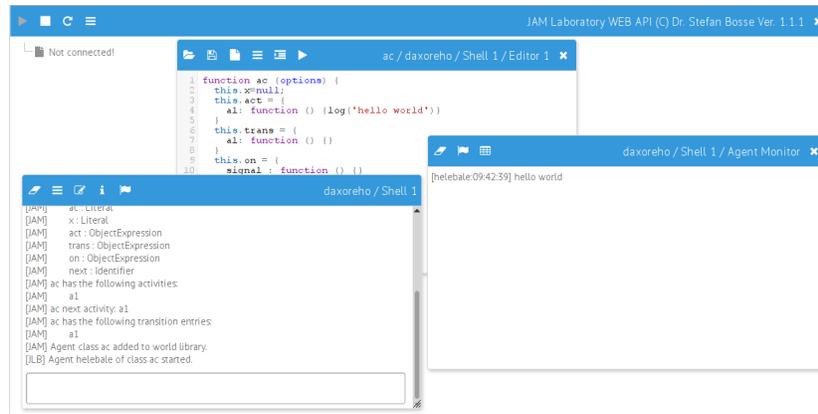
**Mobile JAMApp**

- Die JAMApp ist für mobile Geräte und Desktop/Notebook Rechner verfügbar (identische GUI)
- Die JAMApp hat Zugriff auf Gerätesensoren und bietet einen Chatdialog der von einem Agenten moderiert wird.





**Das JavaScript Agenten Labor (webui)**



**Figure 5.** Die JAM Shell ist als Agentenlabor in jedem WEB Browser verfügbar

## 4. Data Mining



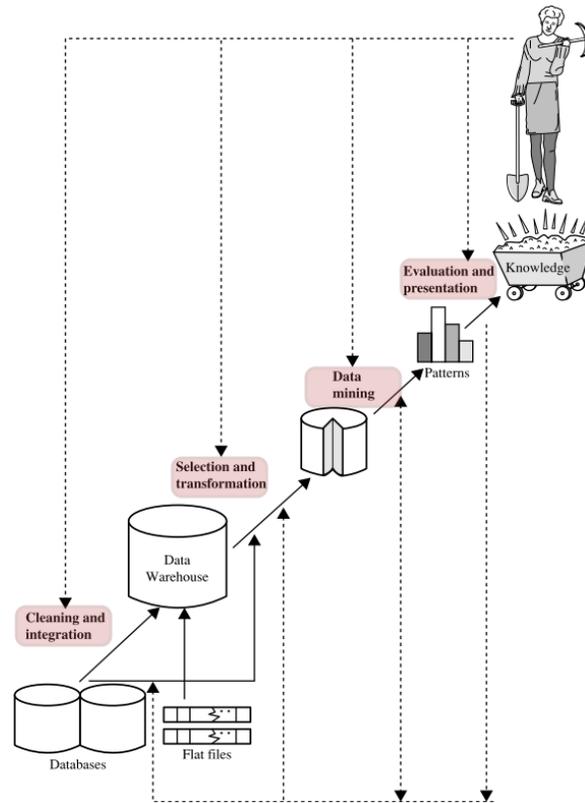
#### **4.1. Data Mining - Grundlagen**

Data Mining  $\Leftrightarrow$  Knowledge discovery from data (KDD)

##### ***Schritte im Data Mining***

1. Datenbereinigung und Filterung (Rauschen entfernen)
2. Datenintegration (Kombination)
3. Datenselektion (Relevante Daten)
4. Datentransformation (Aggregation)
5. Datensuche / Data Mining (KI Methoden)
6. Evaluierung von Mustern (Identifikation von interessanten Mustern)

## 7. Wissensrepräsentation



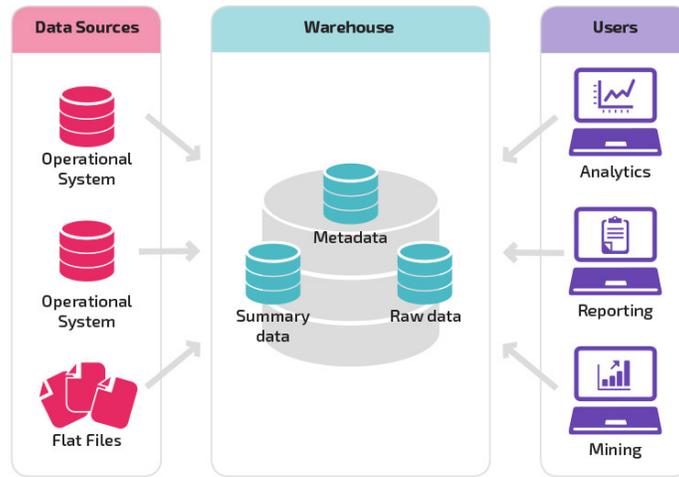
[E]

### ***Welche Daten und Quellen?***

1. Datenbanken
2. Waren und Digitale Warenhäuser
3. Transaktionen
4. Dokumente (Text, Tabelle, Grafiken, Bilder)
5. Suchmaschinen: Index, Einträge, Statistiken
6. Allgemeine Statistiken
7. Diskussionsplattformen
8. Soziale Netzwerke

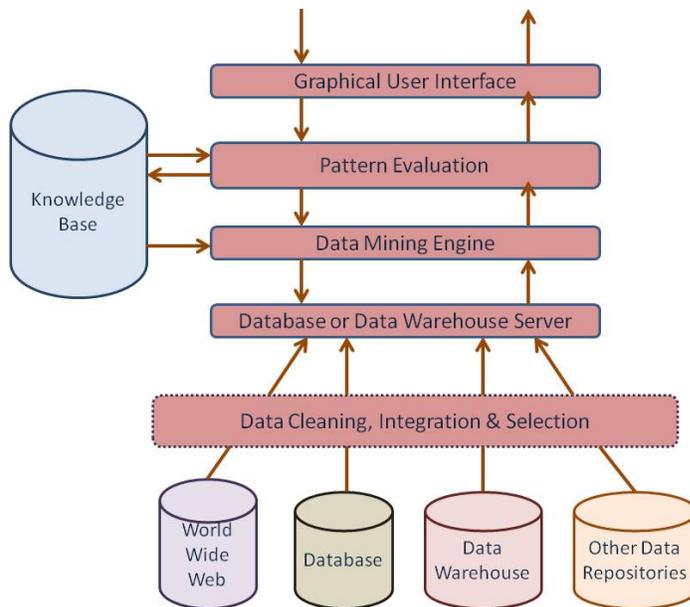
9. Wissenschaftliche Publikationen

10. Nachrichten



[www.panoply.io]

### Architektur

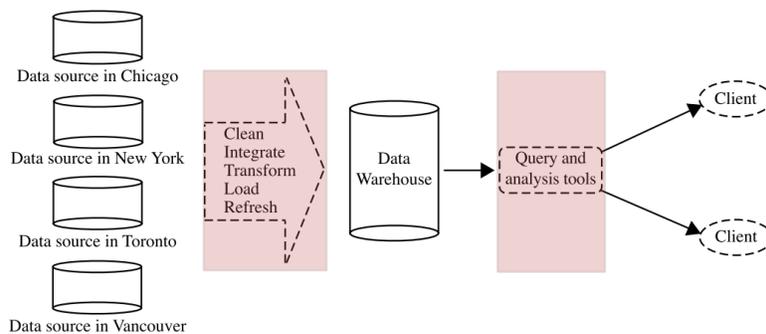


**Figure 6.** Allgemeine Data Mining Architektur [www.wideskills.com]

## 4.2. Data Mining - Datenwarenhäuser

### Begriff

- Ein Datenwarenhaus (Data Warehouse, DW) ist ein Sammlung (Pool) mit Informationen, die aus mehreren Quellen gesammelt, unter einem einheitlichen Schema gespeichert werden und sich normalerweise an einem einzigen Standort befinden.
- Datenwarenhäuser werden durch einen Prozess der Datenbereinigung, Datenintegration, Datentransformation, Datenspeicherung und periodische Datenaktualisierung erstellt.



**Figure 7.** Typisches Framework und Datenverarbeitungsarchitektur in einem Datenwarenhaus [E]

### Eigenschaften

#### Zentrale Datenverwaltung

Die Daten sind zentral gespeichert und nicht über viele verschiedene Systeme verteilt → erhöht Performance und den Überblick.

#### Archivierung der Daten

Auch historische Daten, die für die Operativen Systeme meist bedeutungslos sind, werden in einem DW gespeichert, um den Analysen mehr Aussagekraft zu verleihen.

#### Schneller Zugriff

Mehrdimensionale Abfragen auf dem DW sind effizienter als einzelne Abfragen auf den Quellsystemen.

#### Zugriff

Das DW ist für viele Nutzer zugänglich.

## **Unabhängigkeit**

Unabhängigkeit zwischen Datenquellen und Analysesystemen

### **4.3. Data Mining Funktionen**

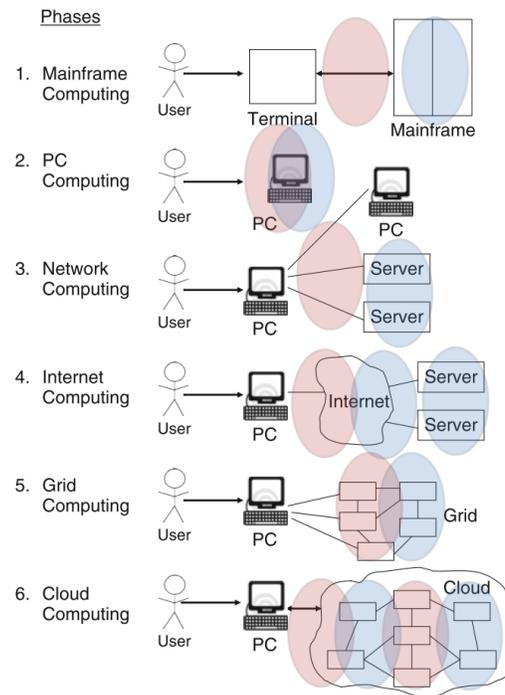
- ▶ Es gibt eine Reihe von Data Mining-Funktionen. Dazu gehören
  - ❑ Charakterisierung und Diskriminierung;
  - ❑ Die Gewinnung häufiger Muster, Assoziationen und Korrelationen;
  - ❑ Klassifizierung und Regression;
  - ❑ Clusteranalyse; und
  - ❑ Ausreißeranalyse.
- ▶ Data Mining-Funktionen werden verwendet, um die Arten von Mustern anzugeben, die in Data Mining-Aufgaben zu finden sind.
- ▶ Im Allgemeinen können solche Aufgaben in zwei Kategorien unterteilt werden:
  - ❑ **Beschreibende Kategorie**; und
  - ❑ **Vorhersagende Kategorie**.
- ▶ **Beschreibende** Mining-Aufgaben charakterisieren die Eigenschaften der Daten in einem Zieldatensatz.
- ▶ **Vorhersagende** Mining-Aufgaben führen eine Reduzierung der aktuellen Daten durch, um Vorhersagen zu treffen.

### **4.4. Data Mining - Datenverarbeitung**

#### ***Datenverarbeitungsparadigmen***

- ▶ Die Datenverarbeitung kann klassifiziert werden nach Daten- und Instruktionsströmen (DS/IS)
  - ❑ Ursprünglich nach **Flynn** für Parallelrechner angewendet
  - ❑ Art und Anzahl der Daten- und Instruktionsströme bestimmt die Performanz des Gesamtsystems

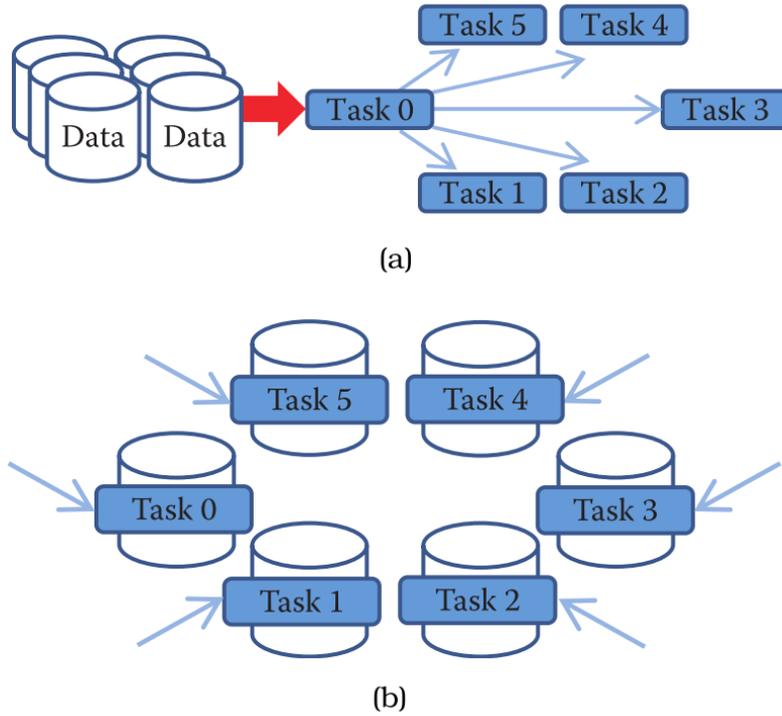
- ❑ Je stärker Daten- und Instruktionsströme lokalisiert sind (d.h. keine oder nur geringe Netzwerkkommunikation benötigen), um so hier wird die Performanz
  - ❑ Große Datenmengen müssen gespeichert werden → Verteiltes Speichermodell vs. Geteiltes Speichermodell
- Kann für netzwerkgekoppelte und verteilte Rechnersysteme adaptiert werden:
- ❑ Wo sind die Daten?
  - ❑ Wohin müssen Daten übertragen werden (DS)?
  - ❑ Wo werden die Daten verarbeitet (IS)?



**Figure 8.** Klassifikation der verteilten Datenverarbeitung nach Daten- (rot) und Instruktionsströmen (Blau)[G]

### **Verteilte Datenverarbeitung**

- ▶ Klassischer Ansatz (1) : Zentrale Recheninstanz mit paralleler Datenverarbeitung
  - ❑ Erfordert Datenkollektion mit großen Datenströmen, Daten müssen zur Berechnung gebracht werden → nicht (gut) skalierbar
  - ❑ Beim Data Mining (und Big Data) sind die Daten bereits verteilt!
- ▶ Klassischer Ansatz (2) : Verteilte Recheninstanzen (Cloud) mit paralleler Datenverarbeitung
  - ❑ Auch hier sind Daten und Berechnung u.U. noch getrennt und müssen zusammengeführt werden
- ▶ Neuer Ansatz (z.B. Hadoop) : Verteilte Recheninstanzen (Cloud) mit paralleler Datenverarbeitung mit Datenkopplung
  - ❑ **Die Berechnung wird zu den Daten gebracht!**



**Figure 9.** Das traditionelle parallele Rechenmodell, bei dem Daten zu den Rechenknoten gebracht werden. (b) das parallele Rechenmodell von Hadoop: Berechnen der Daten [F].

## Map-Reduce Algorithmen

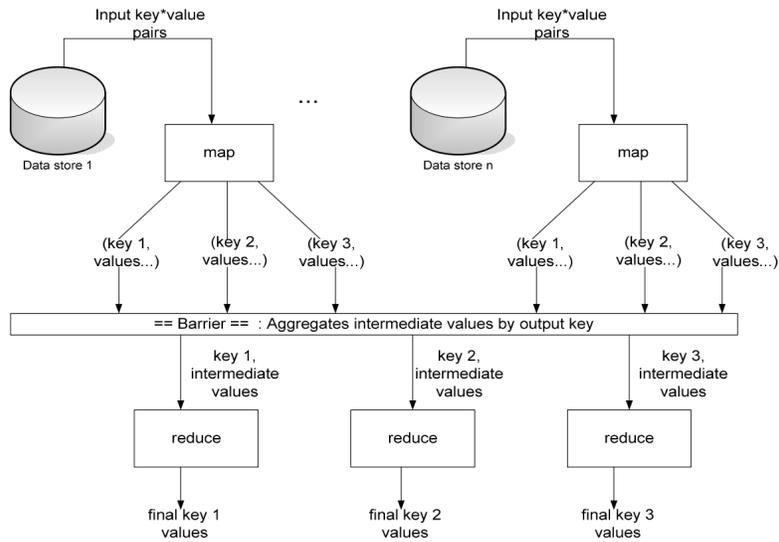


Figure 10. Generelles Map-Reduce Framework und Datenfluss [a]

### Motivation

- Großskalige Datenverarbeitung
  - ❑ Viele Daten verarbeiten (> 1 TB)
  - ❑ Parallelisierung auf Hunderttausende von CPUs
  - ❑ Einfache Handhabung und Umsetzung
- Map-Reduce bietet:
  - ❑ Automatische Parallelisierung und Verteilung
  - ❑ Fehlertoleranz
  - ❑ Stellt Status- und Überwachungstools bereit.
  - ❑ Abstraktion für Programmierer

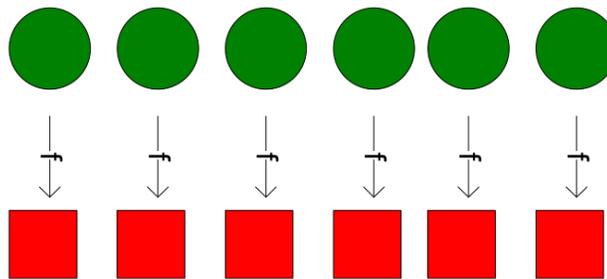
## **Funktionale Programmierung**

- ▶ Map-Reduce Algorithmen, wie sie z.B. im Hadoop System verwendet werden, haben Ähnlichkeit mit funktionalen Programmiersprachen (wenn auch Map&Reduce kein spezifisches funktionales Problem ist):
  - ❑ Funktionale Operationen verändern keine Datenstrukturen: Sie erstellen immer neue;
  - ❑ Originaldaten sind unverändert vorhanden;
  - ❑ Datenfluss ist im Programmdesign implizit;
  - ❑ Reihenfolge der Operationen spielt keine Rolle.
  - ❑ Funktionale Programmierung wendet Operationen auf Listen an

### **Map0**

```
function (f: 'a->'b, 'a list) -> 'b list
```

Ursprüngliche funktionale *map* Funktion. Erzeugt eine neue Liste durch Anwendung einer Funktion *f* auf jedes Element der Eingangsliste (oder Arrays)  
→ **Abbildung**

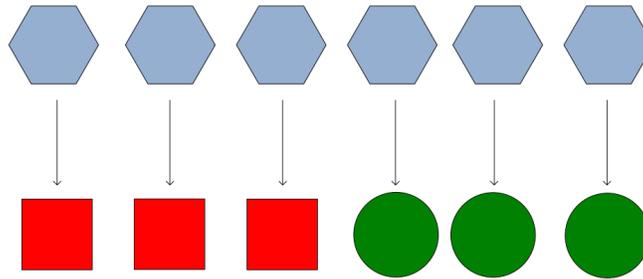


[a]

### **Map**

```
function (in_key,in_value) -> (out_key,intermediate_value) list
```

Adaption der ursprünglichen *map* Funktion. Datensätze von einer Datenquelle (Zeilen aus Dateien, Zeilen einer Datenbank usw.) werden als Schlüssel / Wert-Paare in die Abbildungsfunktion eingegeben: z.B. (Dateiname, Zeile). *map* erzeugt aus der Eingabe einen oder mehrere Zwischenwerte zusammen mit einem Ausgabeschlüssel.

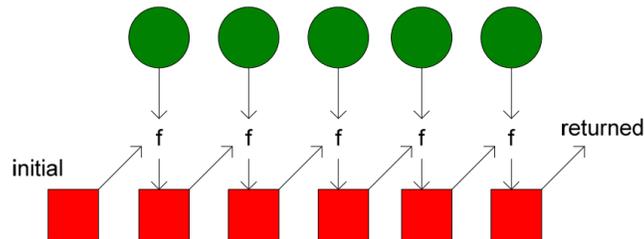


[a]

### Fold

```
function (f:'a*'b->'b', x0:'b', lst;'a list) ->'b
```

Ursprüngliche funktionale *fold* Funktion. Iteriert über eine Liste und wendet  $f$  auf jedes Element plus einen Akkumulator an (Anfangswert  $x_0$ ).  $f$  gibt den nächsten Akkumulatorwert zurück, der mit dem nächsten Element der Liste kombiniert wird → **Reduktion**.

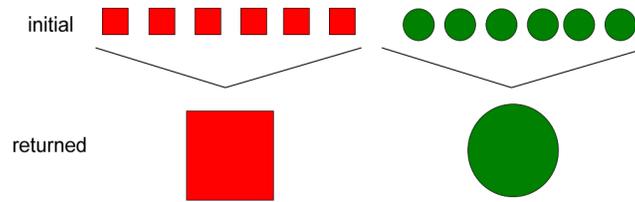


[a]

### Reduce

```
function (out_key, intermediate_value list) -> out_value list
```

Adaption der ursprünglichen *fold* Funktion. Nach der Beendigung der Abbildungsphase werden alle Zwischenwerte für einen bestimmten Ausgabeschlüssel zu einer Liste zusammengefasst. *reduce* kombiniert diese Zwischenwerte zu einem oder mehreren Endwerten für denselben Ausgabeschlüssel (in der Praxis normalerweise nur ein Endwert pro Schlüssel)

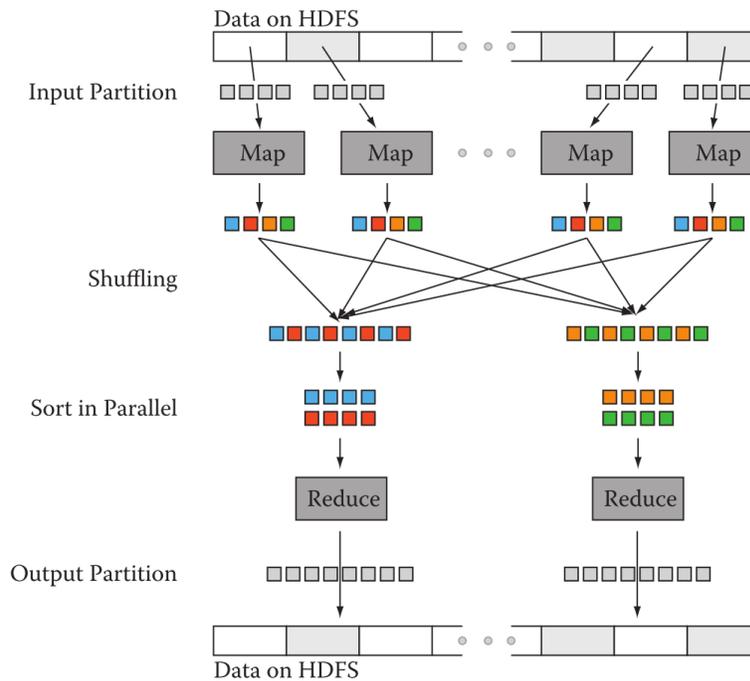


[a]

### ***Map-Reduce Beispiel in JavaScript***

### ***Map-Reduce Methode in Hadoop***

- Daten werden verteilt im HDFS (Hadoop Distributed File System) abgelegt
- Datenblöcke werden mehreren Mappern zugewiesen, die Schlüssel-Wert-Paare ausgeben, die parallel gemischt und sortiert werden.
- Der Reduzierungsschritt gibt ein oder mehrere Paare (mit Daten die den gleichen Schlüssel haben) aus, wobei die Ergebnisse im HDFS gespeichert werden



**Figure 11.** Datenübertragung und Kommunikation eines MapReduce-Jobs in Hadoop [F].

### Limitierungen von Hadoop

#### Leistung

- Hadoop hat sich als skalierbare Implementierung erwiesen, die auf Tausenden von Cores laufen kann.
- Es ist jedoch auch bekannt, dass der Aufwand für die Einrichtung von Jobs relativ hoch ist und die Laufzeit nicht optimal ist.
- Eine leere Aufgabe in Hadoop (d.h. Ohne Mapper oder Reduktion) kann selbst in einem modernen Cluster etwa 30 Sekunden dauern.
- Dieser Aufwand macht ihn für Echtzeitdaten oder interaktive Jobs ungeeignet.
  - ❑ Das Problem rührt hauptsächlich von der Tatsache her, dass Hadoop-Überwachungsprozesse nur innerhalb eines Jobs leben.
  - ❑ Daher muss dieser Prozess bei jeder Übergabe eines Jobs gestartet und gestoppt werden, was wiederum zu einem erheblichen Overhead

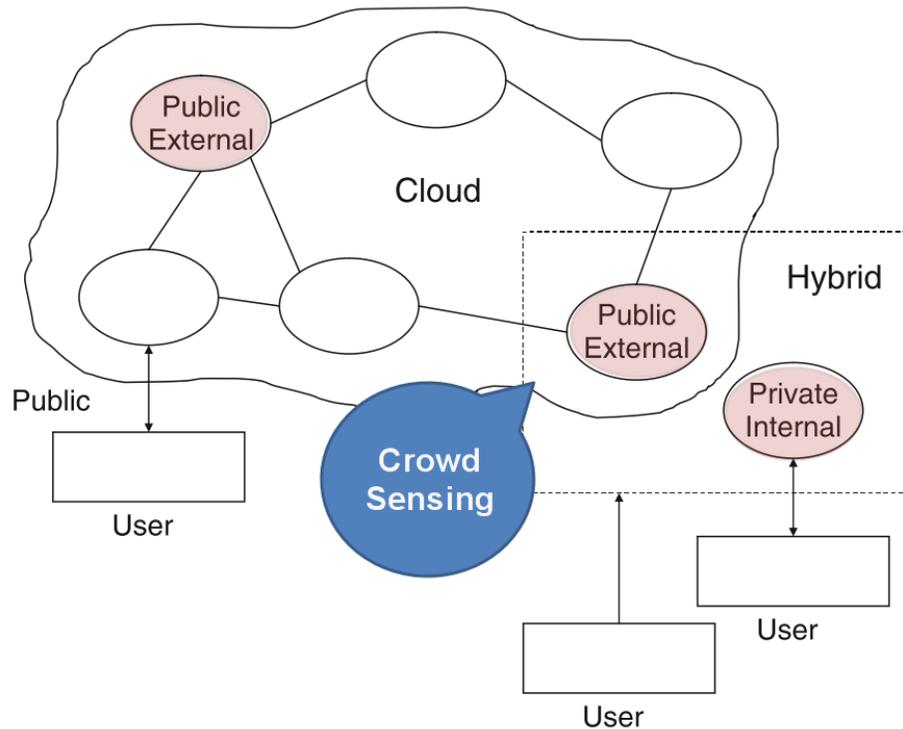
führt.

### **Echtzeitanwendungen**

- In der aktuellen Konfiguration unterstützt Hadoop nur Batch-Datenverarbeitungsjobs. Dies ist beabsichtigt, daher ist es keine Einschränkung für Hadoop selber.
- Da jedoch immer mehr Anwendungen großvolumige Datensätze in Echtzeit verarbeiten, wächst die Community, die Map-Reduce für die Echtzeitverarbeitung verwendet, ständig weiter.
- Die Unterstützung von Streaming oder Echtzeitdaten ist für Hadoop eindeutig ein Nachteil gegenüber anderen Implementierungen.

### **Cloud Computing**

- Cloud Computing bedeutet die lose Kopplung von Rechnern in einem Netzwerk (Internet) bei dem sich die Konfiguration (Art und Anzahl der Rechner) sowie die Vernetzung dynamisch ändern können
- Eine Cloud kann Datenverarbeitungsdienste bereit stellen die über eine (evtl. käufliche) Schnittstelle nutzbar sind → Cloud Services.
- Virtualisierung von Speicher und Berechnung spielt eine große Rolle beim Cloud Computing → Ziel: Mobile Agenten!
- Beim Cloud Computing, welches auch beim Crowdsensing eingesetzt wird, unterscheidet man verschiedene Klassen:
  1. **Öffentliche Cloud**
  2. **Private Cloud**
  3. **Hybride Cloud**
- Crowdsensing an der Schnittstelle von öffentlicher und privater Cloud



**Figure 12.** Überlappung von öffentlicher und privater Cloud in Hybriden Umgebungen [G]

### **Cloud Computing-Funktionen**

- Cloud Computing bringt im Vergleich zu anderen Computing-Paradigmen eine Reihe neuer Funktionen mit sich (Wang et al., 2008; Grossman, 2009).

### **Skalierbarkeit und On-Demand-Services**

Cloud Computing stellt Ressourcen und Services für Benutzer zur Verfügung. Die Ressourcen sind über mehrere Rechenzentren skalierbar.

### **Benutzerzentrierte Benutzeroberfläche**

Cloud-Schnittstellen sind ortsunabhängig und können von jedem durch etablierte Schnittstellen wie Webdienste und Internetbrowser genutzt werden.

### **Garantierte Quality of Service (QoS)**

Durch Cloud-Berechnungen kann QoS für Benutzer in Bezug auf Hardware / CPU, Leistung, Bandbreite und Speicherkapazität garantiert werden.

### Autonomes System

Die Cloud-Computing-Systeme sind autonom und transparent. Software und Daten in Clouds können jedoch je nach den Anforderungen des Benutzers automatisch rekonfiguriert und zu einer einfachen Plattform konsolidiert werden.

### Preisgestaltung

Cloud Computing erfordert keine Investitionen. Benutzer zahlen für Dienste und Kapazität, wenn sie benötigt werden.

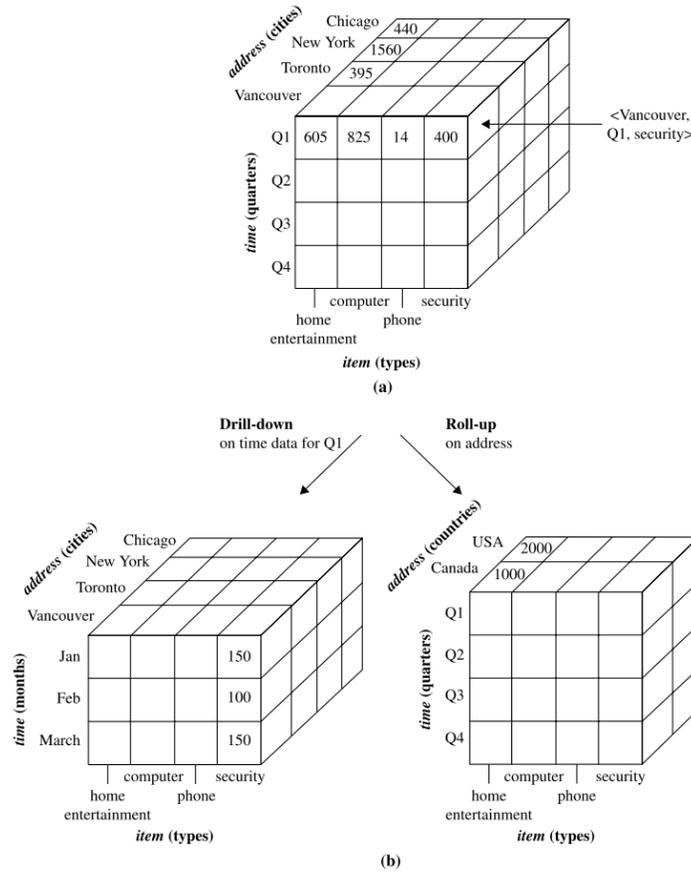
Company	Cloud computing platform	Year of launch	Key offerings
Amazon.com	AWS (Amazon Web Services)	2006	Infrastructure as a service (Storage, Computing, Message queues, Datasets, Content distribution)
Microsoft	Azure	2009	Application platform as a service (.Net, SQL data services)
Google	Google App. Engine	2008	Web Application Platform as a service (Python run time environment)
IBM	Blue Cloud	2008	Virtualized Blue cloud data center
Salesforce.com	Force.com	2008	Proprietary 4GL Web application framework as an on Demand platform
Bosse et al.	JavaScript Agent Machine (JAM)	2016	Scalable > 1M Devices Mobile Platform, Open Source Agents as Service, WEB/JS

**Figure 13.** Key Player bei Cloud Computing Plattformen [G]

## 4.5. Data Mining - Datenvisualisierung

- Die Daten eines komplexen Datenbanksystems (wie eines digitalen Warenhauses) werden in der Regel durch eine mehrdimensionale Datenstruktur modelliert, die als Datenwürfel bezeichnet wird:
  - ❑ Jede Dimension wird einem Attribut oder einer Gruppe von Attributen im Schema entsprechen
  - ❑ Jede Zelle gibt den Wert irgendeiner aggregierten Kennzahl wie z.B. *count* oder *sum* (z.B. von Verkaufszahlen) an.
- Ein Datenwürfel bietet eine mehrdimensionale Ansicht der Daten und ermöglicht die Vorberechnung und den schnellen Zugriff auf zusammengefasste Daten.

- Es können verschiedene Ansichten des Würfels aus aggregierten Daten abgeleitet werden

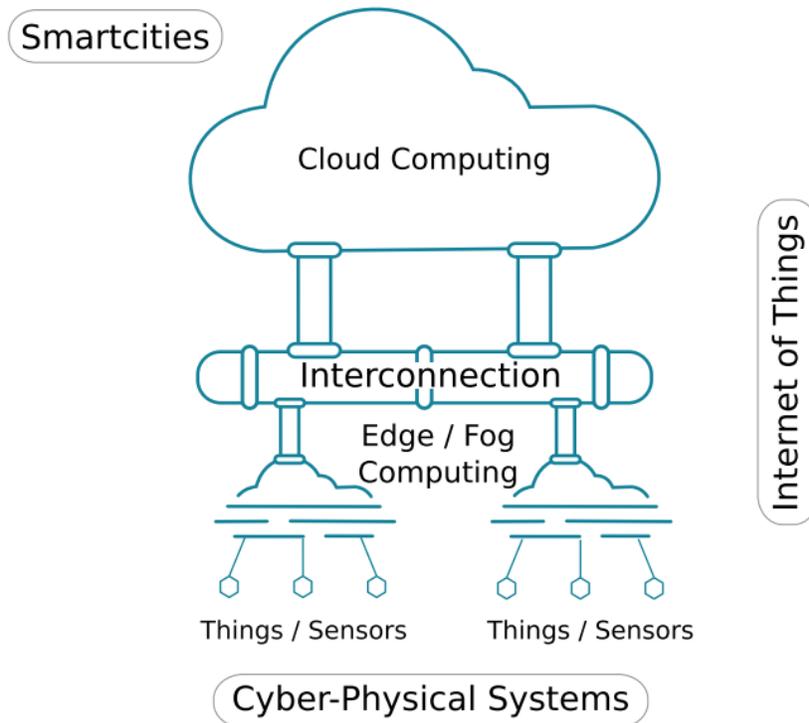


**Figure 14.** Ein mehrdimensionaler Datenwürfel. (a) Zusammengefasste Daten (b) Abgeleitete Daten, die sich aus Drilldown- und Rollup-Vorgängen des Cubes in (a) ergeben [E].

#### 4.6. Smart Environments

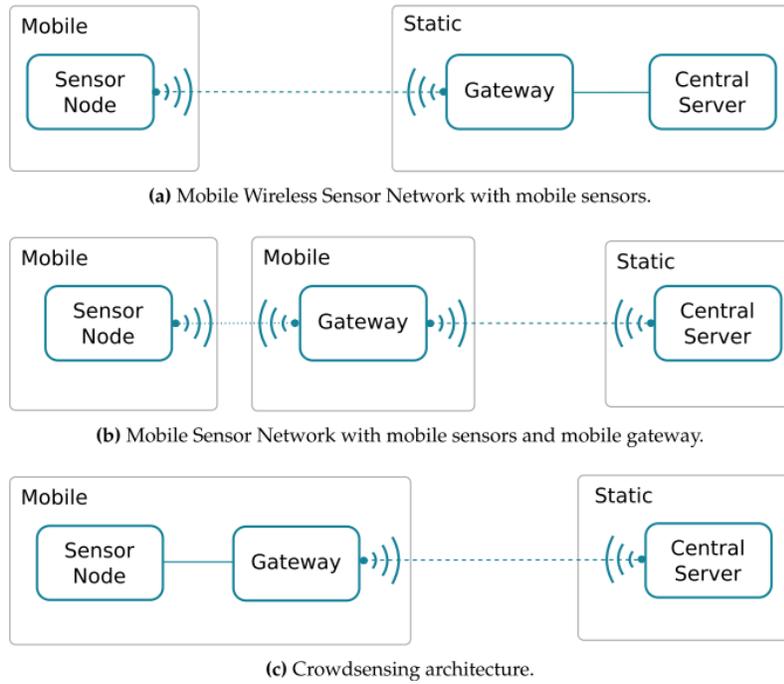
- Z.B. Smart Cities bestehen aus heterogenen Informations- und Kommunikationsarchitekturen und Technologien
- Baumstruktur:
  1. Cloud Computing mit großen Datenbanken und Servern
  2. Internet und Mobile Geräte

3. Edge/Fog Computing
  4. Geräte und Sensoren
  5. Aktuatorische Systeme (Cyber-Physical Systems)
- Existierende Smart Environments stellen eine Plattform für das Crowd-sensing dar
    - ❑ Mit Menschinteraktion
    - ❑ Reine Gerätedaten → Internet der Dinge!
  - Traditionelle Sensornetzwerke (die anwendungsspezifisch sind) und Crowdsensingsysteme (die eher geenerisch sind) sind ähnlich und können sich ergänzen (Sensordatenfusion)



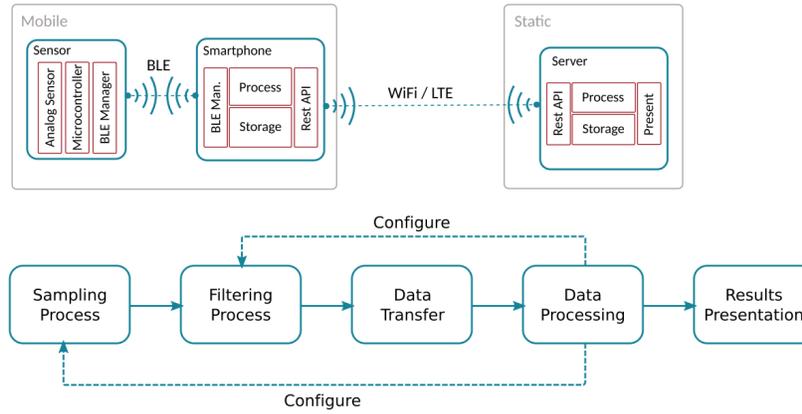
**Figure 15.** Genereller Aufbau eines Smart Environments [4]

**Mobile Netzwerke**



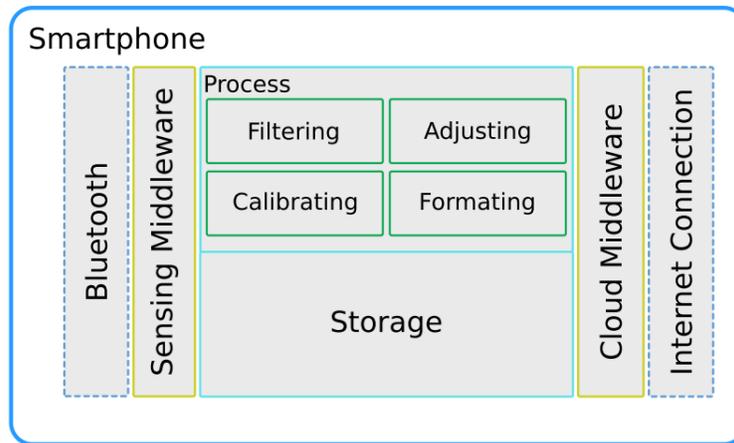
**Figure 16.** Verschiedene Architekturen von mobilen Sensornetzwerken [4]

**Crowdsensing Architektur - Der Klassiker**



**Figure 17.** Crowdsensing mit Gerätesensoren: (Oben) Architektur 1.0 (Unten) Verarbeitungsschritte [4]

**Smartphone - Softwarearchitektur**



**Figure 18.** Crowdsensing mit Smartphones: Cloud basierte Softwarearchitektur ohne Nutzerinteraktion [4]

**5. Crowdsensing**



## 5.1. Bedeutung und Historie

- ▶ Was allgemein als Crowdsourcing betrachtet wird - es **technisch** jedoch **nicht** ist [C]:
  - ❑ Open Source Software Entwicklung
  - ❑ “Commons-Based Peer Production”: Sozioökonomischer Produktionsansatz mit einer großen Anzahl von Personen die kooperativ an einem Projekt arbeiten
  - ❑ “Brand Engagement” bildet eine emotionale oder rationale Bindung zwischen einem Verbraucher und einer Marke. Es umfasst Aspekte der Markenführung.
  - ❑ Die Erstellung des Oxford English Dictionary in den 1800er Jahren war ein frühes Beispiel für Crowdsourcing?

*Crowdsourcing hat seinen Ursprung in der Ökonomie, Wirtschaft, Produktion, Marketing, usw., um den Mehrwert von Produkten zu steigern*

### **Wertsteigerung [D]**

- ▶ **Kostenreduktion**
- ▶ **Innovation**
- ▶ **Authentizität**
- ▶ Zunehmend erkennen aber auch die Soziologie und “Politologie” den Nutzen von Crowdsourcing!
- ▶ Zunehmend wird Crowdsourcing für die freiwillige Erstellung von Geographischen Wissensdatenbanken und Karten verwendet!

## **5.2. Begriffe**

- ▶ *Die Begriffe Crowdsourcing und Crowdsensing sind nicht einheitlich definiert worden.*
- ▶ Man unterscheidet: **Sensorik (Perzeption) und Aktuatorik (Aktion)**

Daher folgende **Klassifikation**:

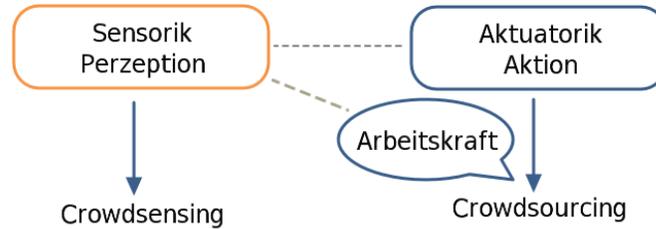
### **Crowdsourcing**

Zunächst kann man unter Crowdsourcing die Nutzbarmachung von menschlichen Arbeitsressourcen mit kollektiven Ziel verstehen, d.h. **Aktuatorik**.

Weiterhin die Erhebung von Nutzerdaten (**Sensorik**) ohne oder mit unmittelbarer Mitwirkung des Nutzers oder die öffentliche selbstorganisierende Verfügbarmachung von Dienstleistungen von Nutzern. Der Nutzer ist sich der Erhebung seiner Daten nicht unbedingt bewusst.

### **Crowdsensing**

Die reine Erhebung von Nutzerdaten mit oder ohne Mitwirkung der Nutzer. Der Nutzer kann sich der Erhebung seiner Daten bewusst sein und aktiv mitwirken. Crowdsensing kann als eine Unterklasse des Crowdsourcings verstanden werden [4].



**Figure 19.** Sensorik und Aktuatorik im Crowdsourcing und Crowdsensing und deren Zusammenhang

**Major Crowdsourcing Systeme** Verschiedene Arten von Crowdsourcing Systemen [H]

- ▶ Kollektives Wissensmanagement → Wikipedia → <http://ag-0.de/dokuwiki>
- ▶ Kollektive Kreativität → Kunst und Fotografie
- ▶ Kollaboratives Online Gaming
- ▶ Kollaboratives Abstimmen → Bewertungssysteme
- ▶ Soziale Netzwerke

### 5.3. Crowdsourcing in der Produktion

- ▶ Einsatz von Crowdsensing und Crowdsourcing zur Optimierung von Prozessflüssen (Produktion usw.)
- ▶ **Serviceorientierte Architekturen (SoA)** im Crowdsourcing:
  - ❑ **Softwarebasierte Services (SBS)**
  - ❑ **Services von Personen (HPS)**
- ▶ Der **Prozessfluss (PFL)** kann sich aus einzelnen Aufgaben zusammensetzen, die entweder von entsprechenden Web-Services verarbeitet oder verantwortlichen Personen zugewiesen werden.
- ▶ In diesem Szenario kann eine Aufgabe (Task) an die Crowd ausgelagert werden.

- Zu diesem Zweck wird eine **Supportanfrage** (RFS) vorbereitet, die verschiedene von der Crowd zu verarbeitende Aufgaben und zusätzliche Metadaten wie z.B. Zeiteinschränkungen und Komplexität der Aufgabe enthält.

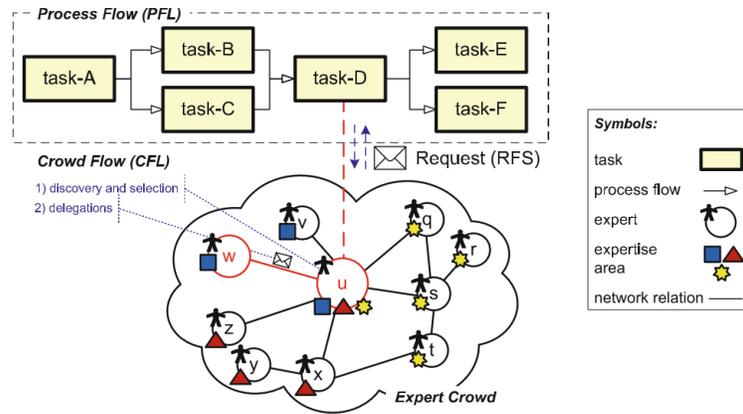


Figure 20. Crowdsourcing in Prozessflüssen [I]

## 5.4. Crowdsensing Systemmodell

- Ein Crowdsensing System dient häufig kommerziellen Anwendungen und besteht aus:
  - **Crowdsourcer (Erheber)** der die Daten empfängt, aggregiert, und i.A. mittels Data Mining aufgabenorientiert analysiert → besteht aus einem oder mehreren sensorverarbeitenden Servern
  - **Crowd** die aus einer großen Zahl von **Nutzern (Datengeber)**, deren sensorische und persönliche Daten i.A. mit mobilen Geräten wie Smartphones erhoben werden.
  - Beide verbunden über eine **Cloudumgebung**

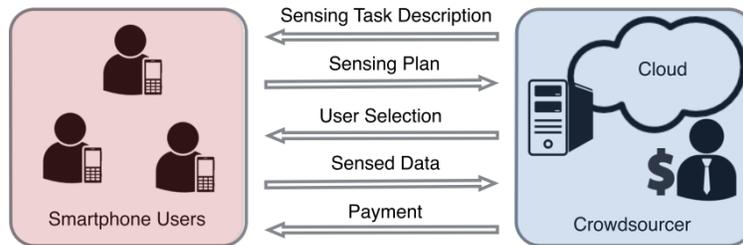


Figure 21. Allgemeine Architektur eines Crowdsensing Systems [1]

### Bidirektionale Verbindungen und Ablauf

1. Die Nutzer bekommen eine Beschreibung und Aufgabenstellung der **Erhebungsaufgabe**
2. Die Nutzer haben einen **Erhebungsplan** zu erfüllen (direkt bewusst oder indirekt durch Anreize)
3. Der Crowdsourcer (**Erheber**) wählt die Nutzer aus (oder gewinnt sie durch ein öffentliches Ausschreibeverfahren)
4. Die Nutzer liefern sensorische Daten für eine bestimmte Zeit  $t \rightarrow$  **Erhebungsdienstleistung**
5. Die Nutzer erhalten **Belohnungen**: direkte Vergütungen oder geldwerte Vorteile - oder nehmen freiwillig im Eigeninteresse teil

Es muss eine **Erhebungsstrategie** geben, die eine hohe Effizienz der Erhebung erreichen soll!

### Systemmodell und Crowdsensingprozess

- Es gibt eine Menge von sensorischen **Erhebungsaufgaben** (Tasks)  $\Gamma$
- Es gibt eine Menge von **Teilnehmern**  $U=\{1,2,\dots,n\}$  die an der Teilnahme der Erhebung interessiert sind *nachdem* sie die Aufgaben kennen (mit  $n \geq 2!$ )
- Jeder Nutzer der an der Erhebung teilnimmt verursacht **Kosten**; der Nutzer erwartet eine **Belohnung** (Bezahlung!?)

- Basierend auf Kosten und Belohnung wird der einzelne Nutzer einen **Erhebungsplan** haben, der wiederum vom Auftraggeber eingesammelt wird und eine globale Bewertung erfolgt mit anschließender Bezahlung/Belohnung des Nutzers (Auftragserteilung)!
- Auftraggeber und Nutzer haben eine **Nützlichkeitsfunktion**
  - ❑ Der Auftraggeber (Erheber) ist nur an der **Maximierung** seiner eigenen Nützlichkeitsfunktion  $u_0$  interessiert
  - ❑ Der Nutzer (Datengeber) ist ebenfalls an der Maximierung seiner eigenen Nützlichkeitsfunktion  $u_i$  interessiert, wird aber erst an der Erhebung teilnehmen wenn der Anreiz groß genug ist!

### **Aufgabe**

1. Welche nicht monetären Belohnungen könnte es noch in Crowdsensing Applikationen geben?
  2. Ist Nutzen schon das Erlangen von Informationen oder Vorteile gegenüber anderen Personen?
  3. Welche Anreize müsste man für Crowdsensing Applikationen geben? Gibt es zielgruppenabhängige Faktoren?
- 

### **Vertiefung**

*D. Yang, G. Xue, X. Fang, and J. Tang, Incentive Mechanisms for Crowdsensing: Crowdsourcing with Smartphones, IEEE/ACM Transactions on Networking, vol. 99, 2015.*  
*F. Hou, Y. Pei, and J. Sun, Mobile Crowd Sensing: Incentive Mechanism Design. Springer, 2018.*

- Man kann Crowdsensing mathematisch modellieren ...

Notation	Description
$\mathcal{U}, \mathcal{U}_{-i}$	set of users and set of users except user $i$
$n$	number of users
$R$	reward of the crowdsourcer
$t_i, t, t_{-i}$	sensing time/strategy of user $i$ , strategy profile of all users, strategy profile excluding user $i$ 's strategy
$\kappa_i$	cost unit of user $i$
$\beta_i(t_{-i})$	best response of user $i$ given $t_{-i}$
$\bar{u}_i, \bar{u}_0$	utility function of user $i$ and the crowdsourcer in the crowdsourcer-centric model
$\lambda$	system parameter in $\bar{u}_0$
$\Gamma, \Gamma_i, \tau_j$	set of tasks, set of user $i$ 's tasks and task
$m$	number of tasks
$\nu_j$	value of task $\tau_j$
$c_i, b_i$	cost and bid of user $i$
$\mathcal{S}$	set of selected users
$p_i$	payment to user $i$
$v(\mathcal{S})$	total value of the tasks by $\mathcal{S}$
$\tilde{u}_i, \tilde{u}_0$	utility function of user $i$ and the crowdsourcer in the user-centric model

**Figure 22.** Notation und Bewertungskriterien für ein Crowdsensing Systemmodell [1]

### *Erheberzentriertes Modell*

- In diesem Modell gibt es nur **eine Erfassungsaufgabe**.
- Der Crowdsourcer (Erheber) kündigt eine Gesamtbelohnung  $R \geq 0$  an und motiviert  $n$  Benutzer, am Crowdsensing teilzunehmen, während jeder Benutzer seinen Beteiligungsgrad auf der Grundlage der Belohnung festlegt.
- Der Erfassungsplan des Benutzers  $i$  wird durch  $t_i \geq 0$  dargestellt, die Zeitdauer, die er bereit ist, den sensorischen Erfassungsdienst bereitzustellen oder an Umfragen teilzunehmen.
- Wenn ein Benutzer  $t_i = 0$  setzt, zeigt der Benutzer  $i$  an, dass er nicht am Crowdsensing teilnehmen wird.
- Die **Erfassungskosten** des Benutzers  $i$  sei  $\kappa_i t_i$ , wobei  $\kappa_i \in \Theta$  Einheitskosten sind und  $\Theta = \{\theta_1, \theta_2, \dots, \theta_l\}$  sei die Menge der Einheitskosten.

- Wir nehmen an, dass der Erheber  $\Theta$  und die Wahrscheinlichkeitsverteilung der Benutzer mit den Einheitskosten  $\theta_j$  kennt. Dies kann aus der Analyse der historischen Daten gelernt werden.
- Sei  $n_j$  die Anzahl der Nutzer mit den Einheitskosten  $\theta_j$ , d.h.  $n = \sum n_j$
- Angenommen, die von Benutzer  $i$  erhaltene Belohnung ist proportional zu  $t_i$ . Dann ist der Nutzen  $\bar{u}_i$  des Benutzers  $i$  die Belohnung abzüglich der Kosten:

$$\bar{u}_i = \frac{t_i}{\sum_{j \in u} t_j} R - t_i \kappa_i$$

- Daher würden alle Nutzer mit den gleichen Einheitskosten den gleichen Erfassungsplan wählen! **Stimmt das?**
- Die Nützlichkeit des Erhebers  $\bar{u}_0$  ist dann gegeben durch eine Bewertungsfunktion  $g$  (die von den tatsächlichen Wahrnehmungszeiten  $\tilde{t}_i$  der Nutzer abhängt) abzüglich der gesamten Belohnung  $R$ :

$$\bar{u}_0 = g(\tilde{t}_1, \tilde{t}_2, \dots, \tilde{t}_l; n_1, n_2, \dots, n_l) - R$$

- Vereinfachung und Feststellung: Nutzer mit den gleichen Einheitskosten werden die gleiche Erhebungszeit bereitstellen!
- Bei diesem Modell besteht also das Ziel des Crowdsourcer darin, den optimalen Wert von  $R$  so zu bestimmen, dass er  $\bar{u}_0$  maximiert.
- Jeder Benutzer  $i$  entscheidet eigenständig über seine Erfassungszeit  $t_i$ , um den angegebenen Wert von  $R$  zu maximieren.
- Da kein rationaler Benutzer bereit ist, einen Erhebungsdienst für eine negative Nützlichkeit bereitzustellen, wird der Benutzer  $i$  auf  $t_i = 0$  setzen wenn  $R \leq \kappa_i \sum t_j$  für alle  $i \neq j$ .

### **Nutzerzentriertes Modell**

- In diesem Modell kündigt der Erheber eine Menge  $\Gamma = \{\tau_1, \tau_2, \dots, \tau_m\}$  von Aufgaben zur Auswahl durch die Benutzer an.
- Jedes  $\tau_j \in \Gamma$  hat für den Erheber einen Aufgabenwert  $v_j > 0$ .
- Jeder Benutzer  $i$  wählt je nach Präferenz eine Teilmenge von Aufgaben  $\Gamma_i \subseteq \Gamma$  aus (basierend auf seinen Präferenzen).
- Dementsprechend hat der Benutzer  $i$  auch damit verbundene Kosten  $c_i$ , die privat sind und nur ihm selbst bekannt sind.

- ▶ **Auktion:** Benutzer  $i$  übermittelt dann ein Auftrag-Gebot Paar  $(\Gamma_i, b_i)$  an den Crowdsourcer, wobei  $b_i$  das Gebot und der Mindestpreis ist, für den der Benutzer seinen Service verkaufen möchte.
- ▶ Beim Empfang der Auftragspakete von allen Benutzern wählt der Erheber eine Teilmenge  $S$  von Benutzern als Gewinner aus und bestimmt die Zahlungen  $p_i$  für jeden gewinnenden Benutzer  $i$ .
- ▶ Der Nutzen des Benutzers  $i$  ist dann:

$$\tilde{u}_i = \begin{cases} p_i - c_i, & \text{wenn } i \in S \\ 0, & \text{sonst} \end{cases}$$

- ▶ Der Nutzen des Crowdsourcers ergibt sich dann aus:

$$\tilde{u}_0 = v(S) - \sum_{i \in S} p_i, \quad v(S) = \sum_{\tau_j} v_j$$

Für das benutzerzentrierte Modell kann man einen Anreizmechanismus entwerfen, der die folgenden vier Eigenschaften erfüllen sollte:

1. **Recheneffizienz:** Das Ergebnis der Auktion kann in Polynomialzeit berechnet werden.
  2. **Individuelle Rationalität:** Jeder teilnehmende Benutzer hat einen nicht negativen Nutzen, wenn er seine wahre Bewertung vorlegt.
  3. **Rentabilität:** Der Wert, den die Gewinner bringen, sollte mindestens der Summe der an sie gezahlten Beträge entsprechen.
  4. **Wahrhaftigkeit:** Kein Bieter kann seinen Nutzen verbessern, indem er ein Angebot abgibt, das von der tatsächlichen Bewertung (die Kosten sind) abweicht
- ▶ Das erheberzentrierte Belohnungsmodell kann als ein **Stackelberg Spiel** verstanden werden.
    1. In der ersten Phase gibt der Erheber die Belohnung bekannt
    2. In der zweiten Phase (nicht kooperativ) werden die Nutzer ihre Erhebungszeit strategisch planen um ihre Nützlichkeit zu maximieren
  - ▶ Frage 1: Gibt es für eine gegebene Belohnung  $R$  eine Reihe stabiler Strategien im Spiel, so dass kein Benutzer durch einseitige Änderung seiner aktuellen Strategie etwas gewinnen kann?
  - ▶ Frage 2: Wenn die Antwort auf F1 positiv ist, ist die stabile Strategie eindeutig? Wenn sie eindeutig ist, können Benutzer sicher sein, die Strategien aus demselben stabilen Strategiesatz auszuwählen?

- Frage 3: Wie kann der Crowdsourcer den Wert von  $R$  auswählen, um seinen Nutzen zu maximieren?

## 5.5. Belohnung

- Im vorherigen Abschnitt wurde die monetäre Belohnung durch Verhandlung eingeführt.
  - ❑ Es ist eine Nützlichkeitsfunktion zu erfüllen → Die Erhebungszeit spielt eine wesentliche Rolle
  - ❑ Sowohl Nutzer (Datenanbieter) als auch der Erheber (Datensammler) haben eine eigene Nützlichkeitsfunktion → ggfs. gegenläufig!
- Die Belohnung soll drei Ziele bei der Datenerhebung erreichen:
  1. Große Anzahl von Teilnehmern → Anreize schaffen
  2. Große Flächenabdeckung (inkl. Repräsentanz)
  3. Datenqualität (Abdeckung der Erhebungsziele)

### **Belohnungsklassen**

- A. Anreiz der Teilnahme durch **monetäre Belohnung** → Makro- und Mikrozahlungen
  - B. Anreiz der Teilnahme durch **Spiele**
  - C. Anreiz der Teilnahme durch **Verbesserung sozialer Kontakte**
- Belohnung ist Motivation ...
  - Man unterscheidet zwei Arten der Motivation:

### **Intrinsische Motivation**

Spaß, Herausforderung, Lernen

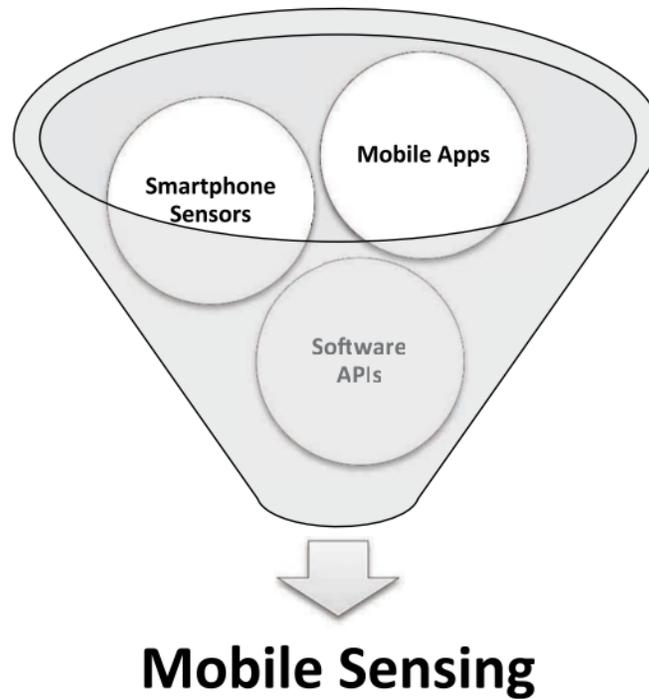
### **Extrinsische Motivation**

Monetäre Ziele, Sozialer Druck, Ruhm

- Intrinsische Motivation kann bedeutender als extrinsische für das Crowdsensing sein
- Beiden Arten gemeinsam ist die individuelle Optimierung der Nützlichkeitsfunktion (des Nutzens) für den einzelnen Nutzer

## 5.6. Mobiles Crowdsensing

- Beim mobilen Crowdsensing handelt es sich um eine Technologie, die eine großräumige, kostengünstige Erfassung der physischen Welt ermöglicht.
- Beim mobilen Crowdsensing werden Menschen, die sich ihrem Alltag widmen, zu „Sensoren“.
- Ihre mobilen Endgeräte wie Smartphones oder Smartwatches sind mit einer Vielzahl von Sensoren ausgestattet, mit denen Daten aus der Umwelt, dem Transport, der Gesundheit, und Sicherheit gesammelt werden können bald.
- Crowdsourcing- und Crowdsensing-Techniken ermöglichen eine auf Menschen ausgerichtete Wahrnehmung, um eine umfassende, kollektive Wahrnehmung zu schaffen.
- In Verbindung mit statischen Sensornetzwerken und Internet-of-Things-Geräten (IoT) kann mobiles Crowdsensing die Vision der allgegenwärtigen Wahrnehmung Wirklichkeit werden lassen.
- Es gibt im wesentlichen drei Komponenten die als Grundlage für das mobile Crowdsensing gesehen werden könne:
  1. Mobile Geräte mit Sensoren (Smartphones)
  2. Mobile Applikationsprogramme
  3. Vereinheitlichte Softwareschnittstellen



**Figure 23.** Mobiles Crowdsensing: Kombination aus Komponenten [A]

#### *Serverzentrierte Systeme*

- ▶ Smartphones “sensorieren” einen räumlichen Bereich und liefern Sensordaten an Server
- ▶ Der Server stellt aber auch einen Service mit aggregierten Daten für die Nutzer zur Verfügung

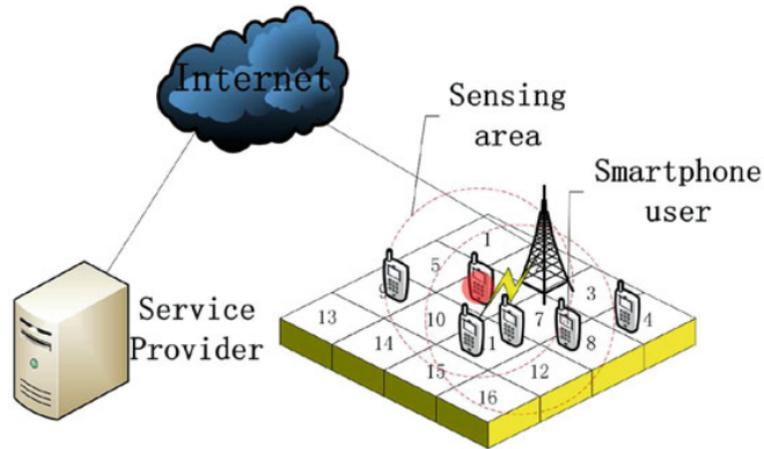


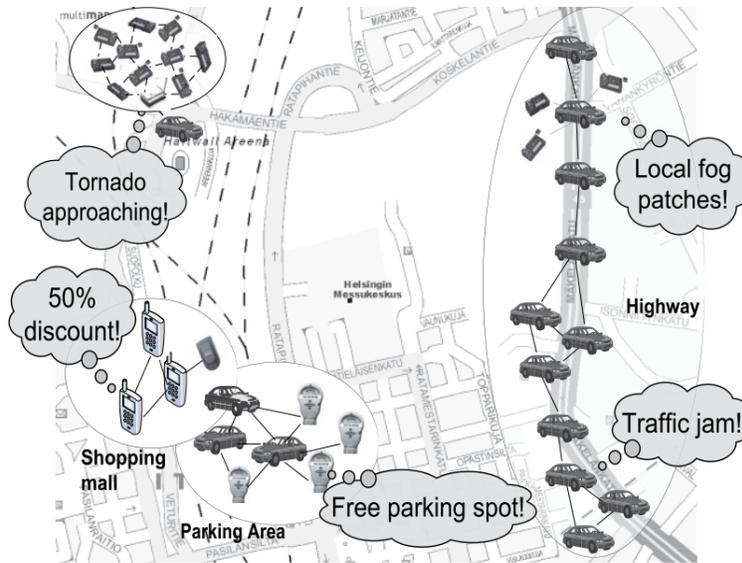
Figure 24. Serverzentriertes Mobiles Crowdsensing: S [L]

*Mobile Crowdsensing: Die Leute sind sowohl Konsumenten als auch Produzenten der sensorischen Daten!*

### **Straßenverkehr und Transport**

- Transportabteilungen können fein aufgelöste und großräumige Daten über das Verkehrsverhalten im Land / Stadt anhand von Standort- und Geschwindigkeitsdaten sammeln, die von Smartphones bereitgestellt werden.
- Dieselben Informationen könnten verwendet werden, um eine individualisierte Verkehrsumleitung zur Vermeidung von Verkehrsstaus bereitzustellen oder um Autofahrer zu freien Parkplätzen zu leiten.
- Daten über die Straßenqualität könnten ebenfalls erhoben werden, um den Kommunen zu helfen, die Straßen rasch zu reparieren.
- In ähnlicher Weise können Fotos (d.h. Kamerasensordaten), die von Personen während / nach Schneestürmen aufgenommen wurden, automatisch analysiert werden, um das Reinigen der Straßen zu priorisieren.

*Geographical Information Systems (GIS) spielen im Mobilem Crowdsensing eine große Rolle da sie räumliche geografische Information mit anderen Sensoren zu einem neuen Kontext kombinieren!*



**Figure 25.** Mobiles Crowdsensing: Die Menge ist gleichzeitig Produzent und Konsument von aggregierten Daten [A]

### Sensortypen und Funktionalität

- Es gibt eine große Anzahl verschiedener Sensortypen in mobilen Geräten [A]

Motion/Position sensors	Environmental sensors	Radios	Other hardware
Accelerometer, Magnetometer, Gyroscope, Proximity sensor, Pedometer	Ambient light sensor, Barometer, Temperature sensor, Air humidity sensor, Radiation sensor	GPS, Bluetooth, WiFi, Cellular	Microphone, Camera, Camera flash, Touch sensor, Fingerprint

### Beschleunigung (ACC)

Ein Beschleunigungsmesser zeichnet die Beschleunigung in drei zueinander senkrechten Achsen  $x$ ,  $y$ ,  $z$  auf. Wenn das Smartphone horizontal auf einem

Tisch angeordnet ist, zeigt eine Achse auf den Vektor der Schwerkraft ( $g$ ). Beschleunigungsmesser spielen eine wesentliche Rolle bei der Messung körperlicher Aktivitäten. In der Regel werden sie zum Erkennen der Ausrichtung eines Smartphones verwendet und werden heutzutage häufig für *verschiedene Spiele- und Aktivitätserkennungsanwendungen verwendet*.

### **GPS**

Dies ist einer der bekanntesten Sensoren des Smartphones. Er berechnet die Position des Smartphones anhand von Satelliten und deren Bodenstationen als Referenz. Die Ausgabe ist in der Regel ein (Breitengrad, Längengrad, Höhen-Tupel), zusammen mit anderen Informationen wie Geschwindigkeit und Richtung. GPS kann (im Smartphone) abgeschaltet sein und steht nicht immer zur Verfügung.

### **Gyroskop (GYP)**

Mit einem Gyroskop wird die Ausrichtung des Smartphones anhand des Drehimpulses gemessen. Es kann die Änderungsrate des Winkels entlang einer bestimmten Achse messen und ist in der Regel zuverlässig, um kurzfristige und sprunghafte Orientierungsänderungen genau zu erkennen. *Smartphones verwenden normalerweise eine Kombination aus Beschleunigungssensor und Gyroskop, um sich gegenseitig zu kalibrieren sowie Orientierungsänderungen, Lokomotionszustände usw. zuverlässig zu erkennen.*

### **Licht**

Bei den Lichtsensoren handelt es sich i.A. um Fotodioden, die die Lichtintensität über einen elektrischen Strom messen. *Lichtsensoren werden verwendet, um die Bildschirmhelligkeit unter verschiedenen Lichtsituationen zu steuern, oder um den Touchscreen zu sperren, wenn ein Benutzer das Telefon an sein Ohr hält.*

- Der Lichtsensor kann auch mit anderen Sensoren für intelligente Dienste gekoppelt werden, z.B. zur besseren Lokalisierung durch die Kombination von Ton und Telefonpositionserfassung (z.B. in der Tasche, auf dem Tisch in der Hand) zusammen mit dem Beschleunigungssensor.

### **Externe Sensoren**

Zusätzlich zu den internen Smartphone-Sensoren kann die Erfassung auch durch externe Sensoren erfolgen, *wie z. B. vom Körper getragene Sensoren im Gesundheitsbereich zur Überwachung von Patienten, Sensoren an Fahrrädern zur Überwachung der täglichen Bewegungsabläufe oder Radrouten sowie Sensoren in Fahrzeugen.*

- Auf diese externen Sensoren wird im Allgemeinen über verschiedene drahtlose Technologien zugegriffen (vor allem Bluetooth ist mit vielen externen Sensorgeräten kompatibel). Diese externen Sensoren werden in Zukunft

mit zunehmender Anzahl und Komplexität der Sensoranwendungen an Bedeutung gewinnen.

### **Kommunikation**

Die Smartphones verfügen über GPS, Bluetooth, WLAN und GSM (Mobilfunk), die üblicherweise für Standort- und Kommunikationsfunktionen verwendet werden.

*Diese Kommunikationsgeräte können in Kombination mit anderen Sensoren die mobile Erfassung des Benutzers verbessern und einen großen Beitrag zu den Standortdaten der Benutzer leisten. Insbesondere wird der GPS-Standort fast immer mit allen Erfassungsdaten gekennzeichnet, um die Daten aussagekräftiger und kontextbezogener zu machen.*

### **GSM**

Globales mobiles Funknetz bestehend aus Basisstationen die sich i.A. im Außenbereich befinden (Ausnahmen: Funknetzrepeater wie z.B. in Zügen). Reichweite ca. 100-2000m.

### **WIFI**

Lokales stationäres Funknetz bestehend aus Basisstationen (WLAN Access-points), die sich sowohl im Innen- als auch AUssenbereich befinden können. Reichweite ca. 10-100m.

### **Bluetooth**

Bluetooth wird verwendet, um Daten über kurze Entfernungen zwischen zwei Geräten über Funkübertragungen im ISM-Band von 2400–2480 MHz auszutauschen.

- Bluetooth kann daher andere Bluetooth-fähige Peer-Geräte in seiner Umgebung erfassen und sich mit diesen verbinden.
- Dank der neuen Bluetooth LE-Technologie (Low Energy) können Smartphones Proximity-Daten effizienter erfassen.
- Diese Eigenschaft wurde in der Forschung häufig verwendet, um soziale Echtzeitbedingungen zu schätzen und zu erfassen, z.B. Schätzen der Bevölkerungsdichte, Berechnen von Gruppenbildung und Ableiten sozialer Kontexte und Kontakte. Reichweite ca. 1-10m.

### **Software APIs**

1. Googles Android API
2. Appels iOS API
3. Microsoft Windows Mobile API
4. andere?

### **Hürden**

#### **Zuverlässigkeit der Daten**

Die von den Teilnehmern übermittelten erfassten Daten sind nicht immer zuverlässig, da sie falsche Daten zur Kompensation übermitteln können, ohne die eigentliche Aufgabe auszuführen. Daher ist es wichtig, die erfassten Daten zu validieren.

- Weiterhin sind Sensoren in mobilen Geräten A.) Unzuverlässig B.) Ungenau!

#### **Belohnung**

Es ist nicht immer wirtschaftlich für Organisationen (z.B. lokale / staatliche Regierungsbehörden oder Forschungsgemeinschaften) oder Einzelpersonen, für jede Art von Erfassungsaufgabe, die öffentliche Dienstleistungen verbessert, die Entdeckung von Wissen unterstützt oder unseren Alltag erleichtert, eine **finanzielle Kompensation zu leisten**.

- Daher ist es notwendig, kostengünstige Lösungen zu finden, um die Teilnehmer dazu anzuregen, die sensorische Erhebung durchzuführen.

#### **Energiebedarf / Batterie**

Bei der Entwicklung und Implementierung mobiler Anwendungen ist der **Energieverbrauch eine wesentliche Randbedingung**, mit dem Entwickler konfrontiert sind.

- Daher ist das Hauptziel, den Energiebedarf des mobilen Gerätes zu minimieren, die Aktionen der Sensoren zu steuern und bei Bedarf auszusetzen.

#### **Sicherheit und Privatssphäre**

Häufig ist der Datenschutz eine große Hürde für die mobile Datenerfassung. Wenn die Sensing-Anwendung Sensing-Dienste von Drittanbietern verwendet, ist der Datenschutz für den Smartphone-Benutzer ein wichtiges An-

liegen, da es schwierig ist, den Eigentümern von Servern, auf die die Sensordaten hochgeladen werden, zu vertrauen.

- Darüber hinaus können diese Server auch von Hackern angegriffen werden. Die Benutzer fühlen sich jedoch sicher, solange die erfassten Daten auf ihrem mobilen Gerät bleiben.
- Ohne die Sensordaten an Sensing Services zu senden, können die App-Entwickler jedoch möglicherweise keine komplexen Benutzeraktivitäten erkennen.
- Darüber hinaus kann dies zu einem hohen Energieverbrauch oder einer langsamen Reaktionszeit führen, auch wenn dies am Telefon möglich ist.
- Daher müssen die App-Entwickler die Anwendungsfunktionen für Benutzer und den Datenschutz abwägen.

## 5.7. Klassifikation von Crowdsensing

- Das mobile Crowdsensing-Modell behandelt jede Person, die ein mobiles Gerät trägt, als potenziellen Sensorknoten und bildet ein großes Sensornetzwerk, das die gesamte Bevölkerung nutzt.
- Crowdsensing lässt sich grob in zwei Hauptkategorien einteilen:
  - ❑ Partizipative manuelle Erfassung, bei der der manuelle Eingriff der Teilnehmer für bestimmte Eingaben erforderlich ist.
  - ❑ Opportunistische automatische Erkennung, bei der die Erkennung auf den Smartphones der Teilnehmer im Hintergrund ausgeführt wird.

### ***Partizipative manuelle Erhebung***

- Bei der partizipativen Wahrnehmung bezieht jede Erhebung den Teilnehmer direkt ein, um bestimmte Aktionen auszuführen, beispielsweise die Aufnahme von Fotos von Orten oder Ereignissen.
- Daher haben die Teilnehmer bei diesem Verfahren die größte Kontrolle darüber, wann, wo und was zu erfassen ist, indem sie entscheiden, welche Erhebungsaufgaben angenommen und abgeschlossen werden sollen
- Bei der partizipativen Erfassung ist es möglich, dass die Erfassung automatisch durchgeführt wird, die Teilnehmer können jedoch die Zeit, zu

der die Erfassung durchgeführt werden kann, die Freigaberichtlinien usw. steuern.

### **Opportunistische automatische Erhebung**

- Bei der opportunistischen Erfassung wird die tatsächliche Erfassung automatisch durchgeführt, und der Teilnehmer ist nicht an der Erfassungsaktivität beteiligt.
- Die Smartphones (Erfassungsssoftware) treffen die geeigneten Entscheidungen und initiieren die Erfassung und Weitergabe von Erfassungsdaten.
- Dieses Sensormodell überträgt Arbeitsleitung von den Teilnehmern auf ein zentrales Sensorerfassungssystem (oder auf ein Ad-hoc-Netzwerk mobiler Geräte), um zu bestimmen, wann, wo und auf welchen Smartphones eines Teilnehmers eine automatische Erkennung durchgeführt wird, um die Anforderungen der Erhebung zu erfüllen.
- Das System berücksichtigt die sozialen Aspekte der Teilnehmer, wenn Entscheidungen bezüglich der Aufgabenplanung für die Erfassung getroffen werden.
- Zu diesen Aspekten gehören der
  - ❑ Aktuelle Standort,
  - ❑ ein möglicher nächster Standort,
  - ❑ die aktuelle Aktivität,
  - ❑ regelmäßige tägliche Muster und ggfs.
  - ❑ der Gesundheitszustand.
- Dieser Art der Erfassung wird als personenzentriertes Erkennen bezeichnet.

### **Beispiele**

- Beispiele für **Partizipative manuelle Erhebung**:
  - ❑ Foto- und Videojournalismus (Berichterstattung durch Bürger)
  - ❑ Straßentransporterfassung
  - ❑ Datenteilung in sozialen Netzwerken
- Beispiele für **Opportunistische automatische Erhebung**:

- ❑ Verhaltens- und Sozialverhaltens Erfassung
- ❑ Verkehrs- und Straßentransporterfassung
- ❑ Innenbereichslokalisierung mit GPS und Gebäudekarten

## 5.8. Semantische Modellierung von Mobilen Sensoren

- Es gibt eine Vielzahl physischer und virtueller Sensoren im Crowdsensing, die heutige Smartphones verwenden.
- Der Schwerpunkt liegt auf der semantischen Analyse von drei repräsentativen Sensoren, die eine wichtige Rolle bei der mobilen Erfassung spielen:
  - ❑ Positionssensoren (wie GPS, WiFi) und
  - ❑ Bewegungssensoren (wie Beschleunigungsmesser, Gyroskop).
- Der grundlegende Schritt zum Extrahieren der Semantik aus diesen Sensordaten ist die Bereitstellung eines **semantischen Modells** zur Darstellung von Sensordatenströmen.
- Es gibt zwei wesentliche **Modellierungstechniken zur Modellierung von Sensordaten**:

### Konzeptionelle Modellierung

Die konzeptuelle Modellierung wird genutzt, um einzelne Aspekte der physischen und sozialen Welt, die uns umgeben, zum Verständnis und zur Kommunikation formal zu beschreiben.

- Konzeptuelle Modelle werden bei der Darstellung von Sensordaten verwendet, z.B. den Konzepten “Stopp”, “Bewegen” und “Pfad” zum Modellieren der Abstraktion von GPS-Koordinaten.
- Eine Reihe von konzeptionellen Modellen können bei Sensoren im Ubiquitous Computing-Paradigma verwendet werden, wobei der Fokus insbesondere auf kontextsensitiven Anwendungen gerichtet wird.

### Ontologien

Im SemanticWeb sind Ontologien die Schlüsseltechnologie zur Modellierung der Konzepte innerhalb einer Domäne, die “formale, explizite Spezifikation einer gemeinsamen Konzeption” bieten.

- Ontologien spielen auch eine wichtige Rolle bei der Modellierung von Sensoren und deren Semantik auf höheren Abstraktionsebenen, z.B. Trajek-

torien für GPS-Daten, Aktivitäten für Beschleunigungsmesserdaten, Kontextmodellierung und Argumentation, usw.

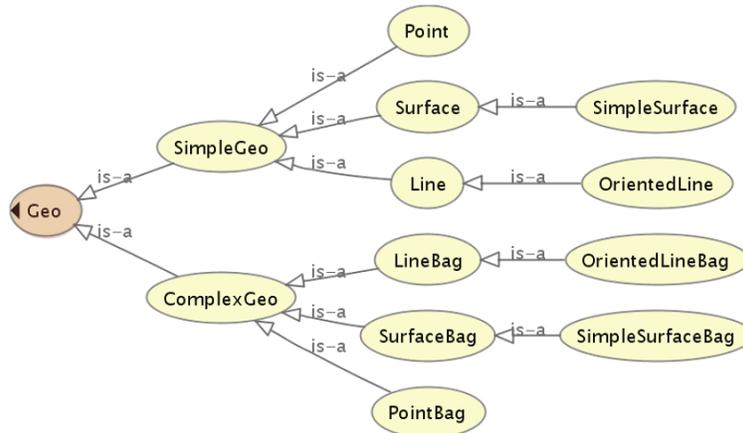


Figure 26. Ontologie für räumliche Beziehungen [H]

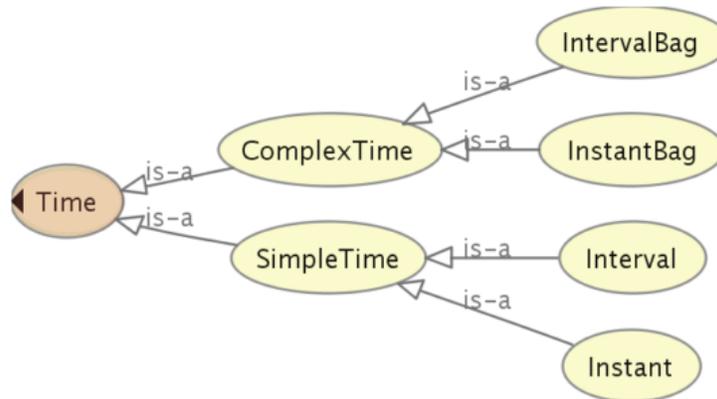


Figure 27. Ontologie für temporale Beziehungen [H]

- Temporale Logiken können bei der semantischen Modellierung insbesondere durch die Einbeziehung von Prädiktion (Abschätzung und Vorhersage) nützlich sein!
- Erweiterung von temporalen Logiken mit räumlicher Annotation ...

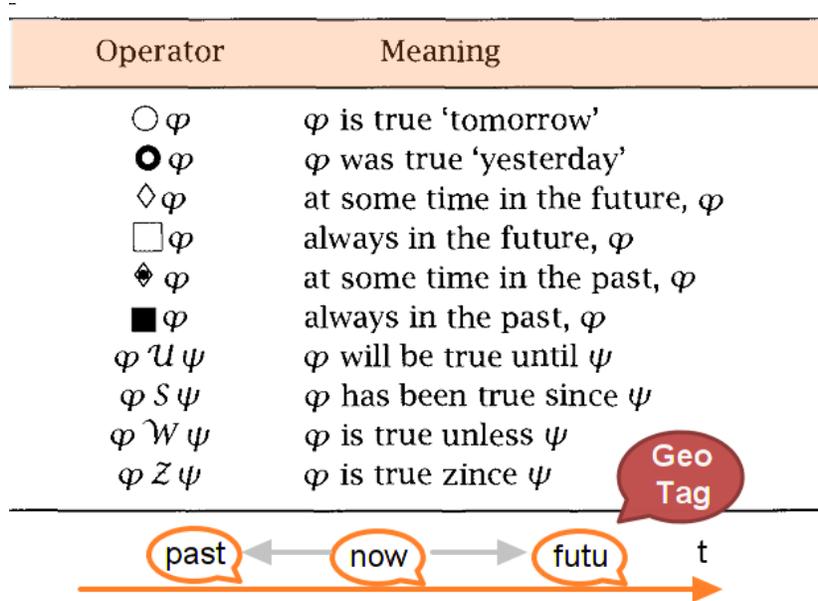


Figure 28. Zeitliche Zusammenhänge von Bedingungen und Aussagen werden durch Operatoren beschrieben [J]

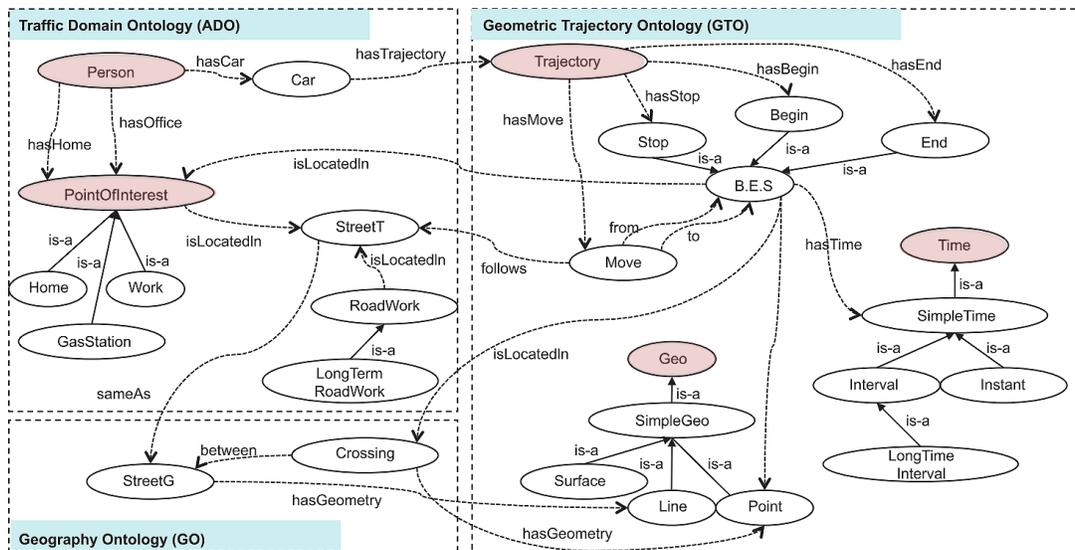
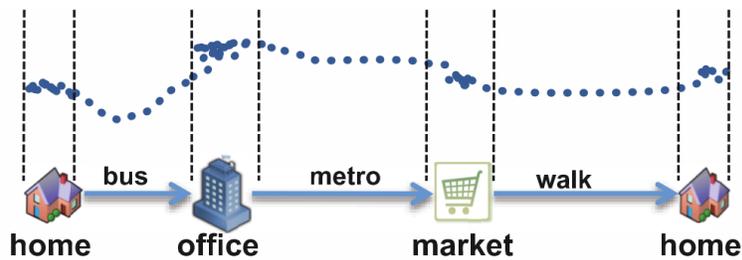


Figure 29. Ontologie eines Verkehrsmanagementsystems mit Crowdsensing [H]

## 5.9. Semantische Modellierung von Trajektorien

### Positionserfassung und Bestimmung

- Im Crowdsensing und Sourcing müssen geographische Positionen auf semantische Trajektorien abgebildet werden:



[H]

### Semantische Trajektorien

Verknüpfung geografischer Trajektorien (Pfade bestehend aus geometrischen Punkten) mit semantischen Attributen.

- Wir können eine Sequenz von GPS-Punkten eines sich bewegenden Objekts untersuchen, um auf die Menge der geografischen Attribute zu schließen, die das sich bewegende Objekt auf seinem Weg passiert.
- Die GPS-Sequenz (Pfad) kann als eine *Folge relevanter Stopps und Bewegungen* dargestellt werden, was aus der Geschwindigkeit des sich bewegenden Objekts abgeleitet werden kann.
- Darüber hinaus kann der Pfad mit geografischen Attributen assoziiert werden, z.B. am Morgen fuhr der Benutzer von zu Hause aus ins Büro, und während der Mittagspause ging der Benutzer zum Mittagessen in ein nahe gelegenes Restaurant.
- All dies sind Beispiele für eine Semantik aus GPS-Daten, die sogenannten "semantischen Trajektorien".

### Trajektorien

### Verarbeitung

Positionssensoren sind fehlerbehaftet (Signalverlust, Rauschen, Abweichungen, usw.). Die Datenverarbeitung muss diese Fehler berücksichtigen und ggfs. erfassen (Annotation).

- Bei GPS nimmt die Umgebung des aktuellen Aufenthaltsortes erheblichen Einfluss auf die Genauigkeit (Sichtbarkeit von Satelliten, Signalstörungen, Erfassungs- und Mittelungszeit)
- Die **Vorverarbeitung** und **Sensorfusion** sind wichtige Bestandteile der Trajektorienerfassung
- Mit zunehmender Erfassungszeit werden die Datenmengen groß! Datenkompression ist erforderlich.
- Durch Sensoren ermittelte Trajektorienpunkte können mit Karten abgeglichen werden (Data Cleaning)

### Aufgabe

1. Erfasse mit Deinem Smartphone die GPS Position (nehme einen zeitlichen Verlauf bis 5 Minuten auf)
2. Vergleiche die Werte mit Deinem Nachbarn
3. Wie lange dauert die erste Positionserfassung nach einem Kaltstart des GPS Empfängers?

---

### Management

Verwendung von Datenbanken für Roh- oder bereits vorverarbeitete Sensordaten ist nur ein Zwischenschritt der Datenreduktion. Besser ist die **Regression** von polynomiellen Funktionen durch die Trajektorienpunkte.

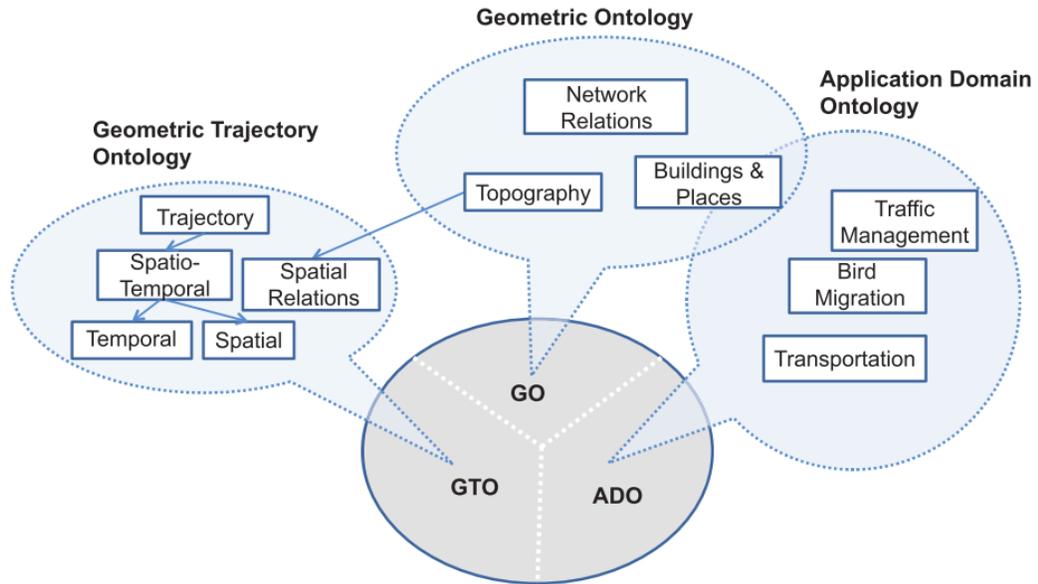
$$y = f(x) = a_0 + a_1x + a_2x^1 + ..$$

- Aber der zeitliche Verlauf ist wichtig, d.h.,  $f(time) \rightarrow Ort!$
- Wesentliche Aufgabe des Managements ist die Findung (query processing) von zeit- und raumassoziierten Trajektorienpunkten um **Zusammenhänge** zu erkennen

### Data Mining

**Maschinelles Lernen** kann verwendet um aus verschiedenen Trajektorienpunktsätzen auf Gemeinsamkeiten und Klassen schließen zu können, z.B. Erkennung von semantischen und sozialen Situationen

### Semantische Trajektorienontologie



**Figure 30.** Aufteilung in geometrische Trajektorien, geometrische Semantik, und Applikationssemantik [H]

- Die Erstellung von Semantischen Modellen aus Positionsdaten erfolgt auf drei Ebenen (Bottom-up):
  1. Datenmodell (Trajektorien, Positionspunkte)
  2. Konzeptuelles Modell (Sequenz von Episoden, Semantische Partitionierung)
  3. Semantisches Modell (Zuordnung von Semantik zu den Episoden)

#### Rohdatenmodell

Das Rohdatenmodell ist die erste *Abstraktionsebene* der Rohmobilitätsdaten. Die Rohdaten werden wie GPS-Aufzeichnungen normalerweise durch Positionssensoren erfasst, die kontinuierlich den Standort des sich bewegenden Objekts erfassen.

- Die rohen Mobilitätsdaten für ein sich bewegendes Objekt sind also im Wesentlichen eine lange Folge von räumlich-zeitlichen Tupeln (*Position, Zeitstempel*) die über einen bestimmten Zeitraum gesammelt werden.

- **GPS Tupel:** (*longitude, latitude, timestamp*)  $\Leftrightarrow (x; y; t)$

### **Konzeptmodell**

Der Begriff konzeptionelles Modell bezieht sich auf die *logische Unterteilung* einer räumlich-zeitlichen Trajektorie  $T_{spa}$  in eine Reihe nicht überlappender Episoden.

- Ein in Episoden unterteiltes  $T_{spa}$  wird als strukturierte Bahn ( $T_{str}$ ) bezeichnet.
- Konzeptionell abstrahiert eine Episode eine Teilsequenz von räumlich-zeitlichen Punkten in  $T_{spa}$ , die einen hohen Korrelationsgrad aufweisen, wie ein räumlich-zeitliches Merkmal (z. B. Geschwindigkeit, Bewegungswinkel, Dichte, Zeitintervall).

### **Semantisches Modell**

Im semantischen Modell ist eine semantische Trajektorie  $T_{sem}$  eine strukturierte Trajektorie, die durch semantische Annotationen ihrer Episoden erweitert wird.

- Semantische Trajektorien können durch **Integrieren von Daten aus geografischen Quellen** von Drittanbietern (z. B. geografische Datenbanken, die Landnutzung, Straßennetze oder Points of Interest beschreiben), soziale Netzwerke, die ortsbezogene Daten enthalten, und allgemeines Wissen über die reale Welt berechnet werden!

### **Beispiel**

- Ein Beispiel für eine semantische Trajektorie ist in der obersten Ebene folgender Abbildung 31 dargestellt.
  - Es zeigt die semantische Trajektorie eines bestimmten Mitarbeiters an einem bestimmten Tag: Er / sie geht morgens von zu Hause zur Arbeit; nachdem er die Arbeit am späten Nachmittag beendet hat, geht er zum Einkaufen auf den Markt und kommt schließlich abends nach Hause.

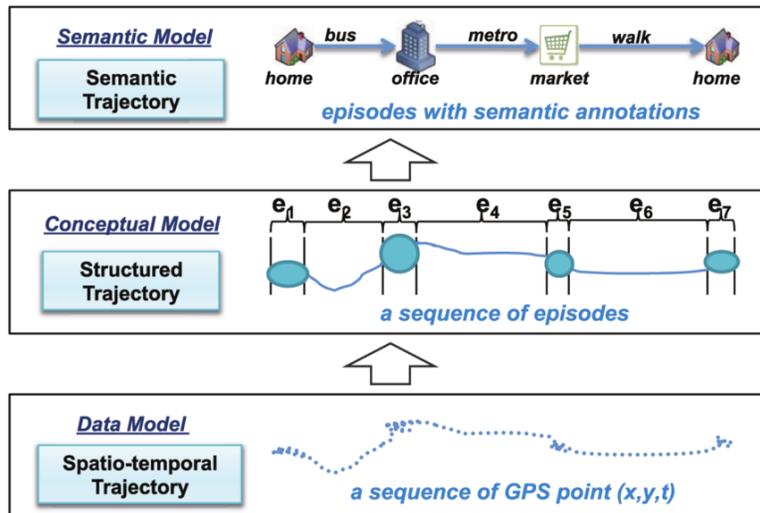


Figure 31. Das hybride raumsemantische Trajektorienschichtenmodell [H]

### Semantische Berechnungsplattform

Aus dem Schichtenmodell ergibt sich eine schichtenartige Verarbeitungsarchitektur.

- Die Datenvorverarbeitungsebene ist die unterste Ebene. Sie dient der Bereinigung der rohen GPS-Daten in Bezug auf vorbereitende Aufgaben wie das Entfernen von Ausreißern und das auf Regression beruhende Glätten.
- Das Ergebnis dieses Schrittes ist eine bereinigte Sequenz  $(x; y; t)$ .
- Die Trajektoridentifizierungsschicht ist die mittlere Schicht. Sie teilt die Sequenz von bereinigten  $(x; y; t)$  Punkten in mehrere sinnvolle Trajektorien (raumzeitliche Trajekturen  $T_{spa}$ ).
  - ❑ Dieser Schritt nutzt Lücken in der Sequenz aus und wendet wohldefinierte Richtlinien für zeitliche und räumliche Abgrenzungen an (z. B. tägliche Zeitintervalle, Stadtgebiete usw.).
- Die Bahnstrukturschicht ist die oberste Schicht und berechnet die in jeder raumzeitlichen Trajektorie vorhandenen Episoden und generiert die strukturierte Trajektorie  $T_{str}$ .
  - ❑ Sie enthält mehrere Algorithmen zum Berechnen von Korrelationen zwischen aufeinander folgenden GPS-Punkten  $\rightarrow$  Geschwindigkeit,

Dichte, Zeitserie.

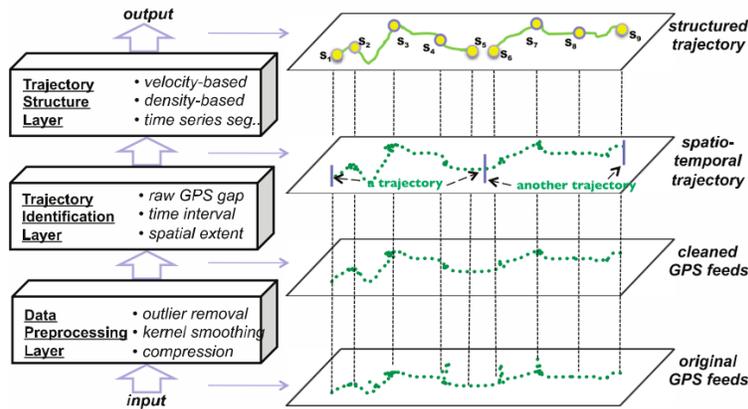


Figure 32. Berechnungsplattform für semantische Trajektorien [H]

## 5.10. Semantische Modellierung von Aktivitäten

### Semantische Aktivitäten

Ein Beispiel sei der Beschleunigungssensor. Die Messdaten des Beschleunigungssensors liefern in erster Linie die dreiachsigen Beschleunigungskomponenten des Geräts. Unter Verwendung geeigneter Algorithmen mit **maschinellen Lernen** können die Messwerte verwendet werden, um auf **Lokomotivzustände** eines Benutzers zuverlässig zu schließen.

- Beispiele für solche **Bewegungszustände** sind:

Gehen, Sitzen, Stehen, Springen, Laufen usw.

- Bei tragbaren Sensoren kann ein Beschleunigungsmesser, der an mehreren Körperstellen getragen wird, verwendet werden, um fein aufgelöste Bewegungen der Gliedmaßen abzuleiten.
- Wenn wir im Laufe der Zeit eine Abfolge solcher Ablesungen machen, kann eine **Kombination** solcher Bewegungen der Gliedmaßen wichtige Informationen liefern, um auf **komplexe Aktivitäten** schließen zu können, wie

Zubereiten von Kaffee, das Laufen auf einem Laufband, das Mittagessen usw.

- Diese komplexen Aktivitäten werden oft als **Aktivitätsroutinen** oder **semantische Aktivitäten** auf hoher Ebene bezeichnet, im Gegensatz zu den vorgenannten Lokomotionszuständen (auch als “Mikroaktivitäten” bezeichnet).

## 6. Introduction to Augmented Virtuality

*Fusion of Real and Virtual worlds creating Augmented Virtuality by using Mobile Agents!*

### 6.1. Simulation of Socio-Technical Systems

- Socio-technical systems are characterized by interactions of:
  - ❑ Human-Human (initiated by a human)
  - ❑ Human-Machine (initiated by a human)
  - ❑ Machine-Human (initiated by a machine, e.g., a chat bot)
  - ❑ Machine-Machine (initiated by a machine)
- The simulation of social ensemble behaviour requires simplification of interactions and individual behaviour
- Commonly simulations are performed with less than 1000 entities (humans, machines, ..) in a sandbox world
- Agent-based Modelling (ABM) is a suitable behaviour model for simulation

### 6.2. Augmented Virtuality

- Simulation worlds are commonly closed and rely on artificial sensory information generated by the simulator program or using data collected off-line (→ field studies).

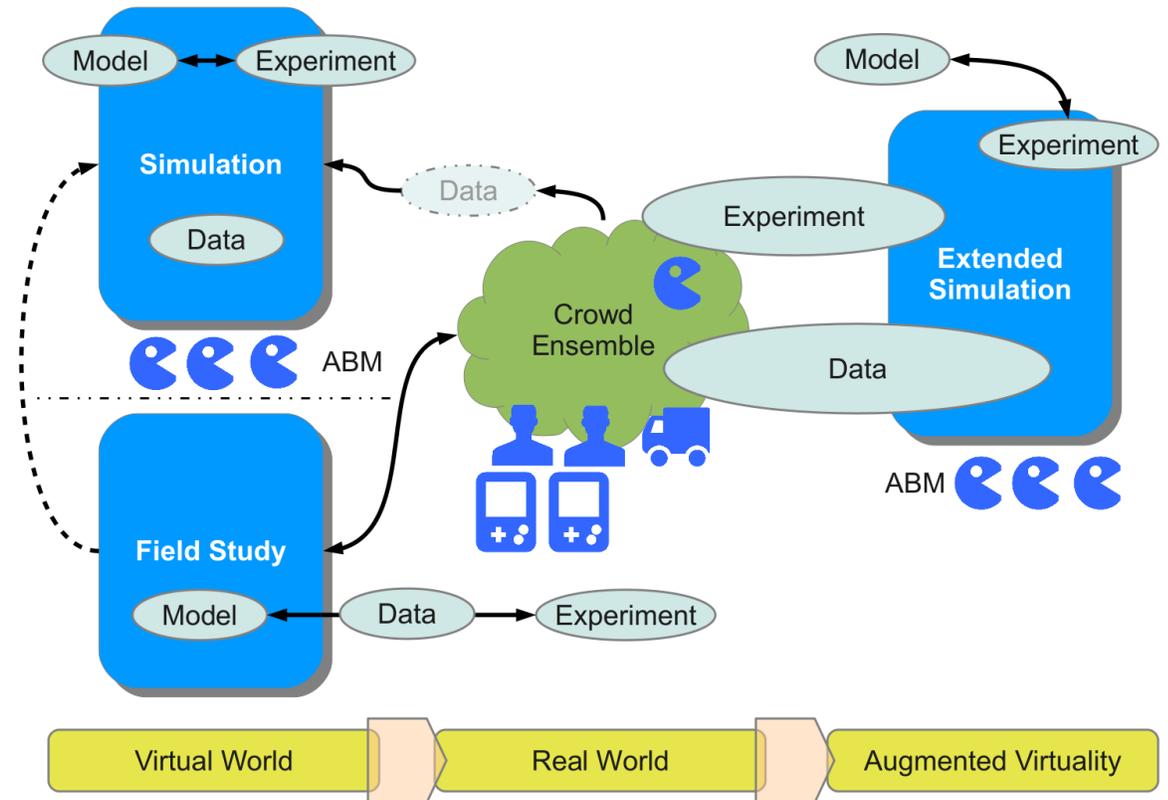
**A new Simulation paradigm providing augmented virtuality**

*Integration of Crowd Sensing and social Data Mining in simulation worlds*

- The simulation world interacts with real world environments, humans, machines, and other virtual worlds in real-time using ...

**Agent-based Modelling and Computation**

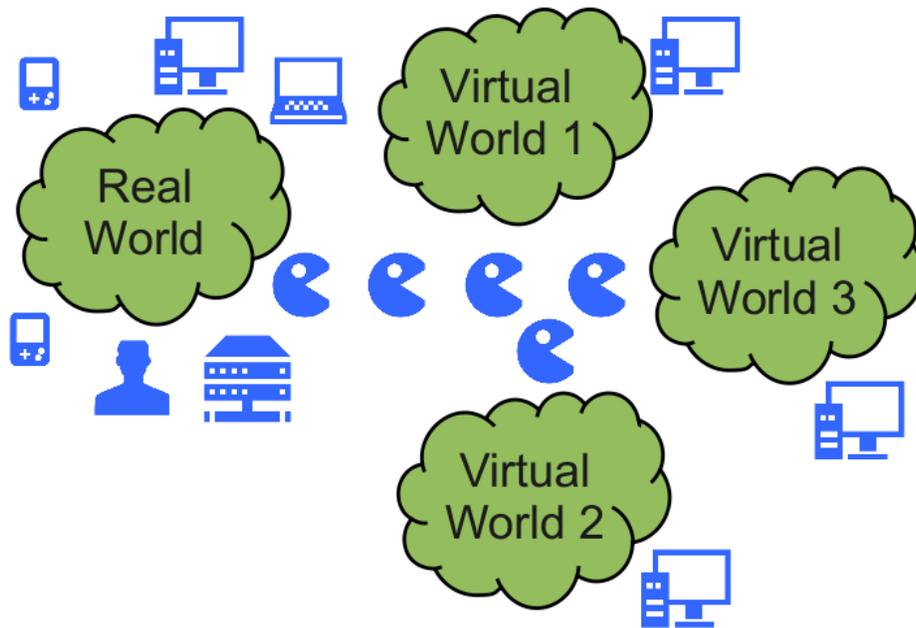
- **Agents** can represent artificial humans, bots, machines ..
- **Agents** can be used for distributed mobile computing → Crowd Sensing



**Figure 33.** Transition from field studies and simulations to extended simulations combining real and virtual worlds

### **Multi-Virtual Worlds**

- Multiple virtual worlds can be connected by the augmented virtuality approach.
- Each virtual world can be considered as a world and model part
- Multiple virtual worlds can be simulated independently and in **parallel**  
→ Speed-up!
- **Virtual and physical world fusion by mobile agents ...**



### **6.3. Challenges**

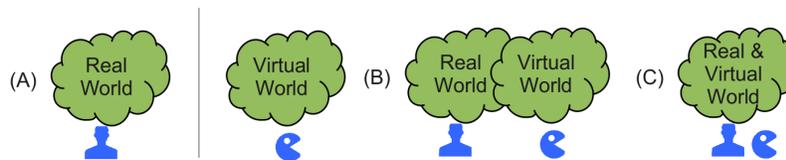
#### **Time**

- Different time scales in real- and virtual worlds
  - ❑ Simulation of distributed computing is much slower than Distributed Computing
  - ❑ Simulation is discrete (processed step-wise) with respect to the time scale

- ❑ Interaction (reactivity) is different in virtual and real worlds
- Short-term versa long-term:
  - ❑ A simulation is performed commonly on a short time interval (window with defined start and end point)
  - ❑ Real world environments with interaction between humans and machines do not pose a well defined starting or end point → long-term execution
- Big Data and Information Strength

### **Space**

- Different spatial scales in real and virtual worlds
- Three different spatial models:
  - A. Real and virtual worlds cover non-overlapping areas
  - B. Real and virtual worlds cover overlapping areas
  - C. Real world area is mapped on virtual world



## **7. Umfragen und Methoden**

*Das “Howto” und “Do not Do” bei der Durchführung von Umfragen - Statistik und Kochrezepte*

### **7.1. Ziele**

1. Entwurf einer Umfrage  $S$  (Survey) startet mit der **Definition der Zielsetzung von Zielen**

2. Häufig ist die Zielsetzung die Beantwortung einer oder mehrerer übergeordneter Fragen  $Q$  um eine Hypothese  $H$  zu testen

$$S = Q \rightarrow H \begin{cases} true \\ false \end{cases}$$
$$Q = \{q_1, q_2, \dots, q_k\}$$
$$H = \{h_1, h_2, \dots, h_l\}$$

### **Beispiele**

- Fühlen sich die Menschen auf der Straße sicher?
- Hat sich die Beschäftigungssituation im Land geändert?
- Nutzen Menschen das Internet während der Pandemie mehr und anders?

## **7.2. Umfragengestaltung**

3. Die Gestaltung der Umfrage unterteilt sich in weitere Schritte [K]:
  - Die genaue Definition der zu untersuchenden Population (die **Zielpopulation**)
  - Die Angabe, was gemessen werden muss (die Variablen  $\mathbf{X}$ ) und was geschätzt werden muss (die **Populationsmerkmale**).
  - Wo die Stichprobe ausgewählt wird (der **Stichprobenrahmen**)
  - Wie die Stichprobe ausgewählt wird (**Stichprobenentwurf** und **Stichprobengröße**).
  - Es ist wichtig die Umfrage sorgsam unter Beachtung obiger Schritte zu planen!

## **7.3. Die Zielpopulation**

*Die Definition der Zielgruppe der Umfrage ist einer der ersten Schritte in der Phase des Umfragedesigns. Die Zielpopulation ist die Population aus Entitäten, die untersucht werden soll. Es ist auch die Bevölkerung, auf die sich die Ergebnisse der Umfrage beziehen. Die Elemente der Zielgruppe sind häufig Menschen, Haushalte oder Unternehmen. Die Bevölkerung besteht also nicht unbedingt aus Personen.*

- Gerade im Mobilen Crowdsensing nicht genau eingrenzbar!

- Die Zielpopulation besteht aus der Menge:

$$U = \{e_1, e_2, \dots, e_N\},$$

mit  $N$  als Größe der Population und  $e$  als individuelle Entitäten.

#### 7.4. Die Ziel- und Hilfsvariablen

- Es werden zwei Arten von Variablen unterschieden: Zielvariablen und Hilfsvariablen.

##### **Zielvariablen**

- Diese Variablen messen die Eigenschaften der Elemente in der Zielpopulation.
- Jeder Teilnehmer aus der Zielpopulation nimmt an der Umfrage teil um eine oder mehrere Zielvariablen  $y$  zu beantworten:

$$Y = \{y^1, y^2, \dots, y^t\}$$
$$y^m = \{y_1^m, y_2^m, \dots, y_N^m\}$$

- Beispiel:  $q_i$ ="Wie hoch ist das Einkommen?"  $\rightarrow y_1$ =Antwort von Teilnehmer 1,  $y_2$ =Antwort Teilnehmer 2 usw.
- Hilfsvariablen messen häufig die Grundeigenschaften der Elemente.
  - Beispiele für eine Umfrage unter Personen könnten Geschlecht, Alter, Familienstand und Region sein.

##### **Hilfsvariablen**

- Solche Hilfsvariablen können nützlich sein, um die **Genauigkeit** von Schätzungen zu verbessern
- Sie spielen auch eine Rolle bei der **Korrektur** der negativen Auswirkungen von Nichtantworten
- Darüber hinaus bieten sie Möglichkeiten für eine detailliertere **Analyse** der Umfrageergebnisse

$$X = \{x^1, x^2, \dots, x^a\}$$
$$x^m = \{x_1^m, x_2^m, \dots, x_N^m\}$$

## 7.5. Populationsparameter

- Ein Populationsparameter ist ein numerischer Indikator, dessen Wert nur von den Werten  $y_1, y_2, \dots, y_N$  einer Zielvariablen  $y$  abhängt.

### Populationssumme

$$\tilde{y} = \sum_{k=1}^N y_k = y_1 + y_2 + \dots, y_N$$

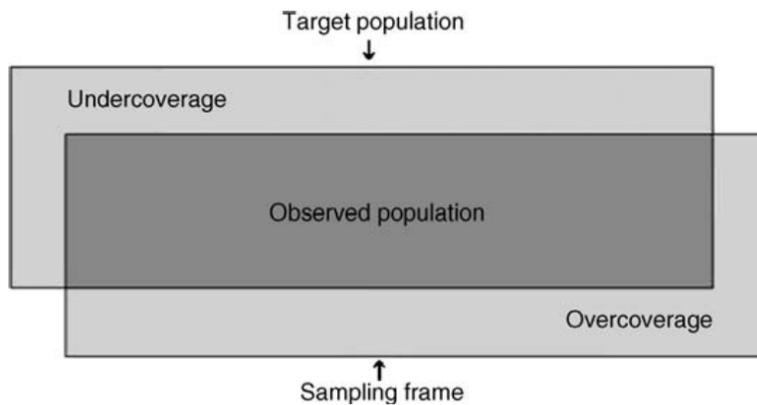
### Populationsmittelwert

$$\bar{y} = \frac{1}{N} \sum_{k=1}^N y_k = \frac{\tilde{y}}{N}$$

### Populationsvarianz

$$\sigma_N^2 = \frac{1}{N} \sum_{k=1}^N (y_k - \bar{y})^2, \sigma_{N-1}^2 = \frac{1}{N-1} \sum_{k=1}^N (y_k - \bar{y})^2,$$

## 7.6. Erhebungsfenster



**Figure 34.** Samplingframe: Problem der Unter- und Überabdeckung bei einer Umfrage [Bethlehem, 2009]

- Ein Stichprobenrahmen ist eine Liste aller Elemente in der Zielpopulation  $U$

- Für jedes Element in der Liste müssen Informationen zur Kontaktaufnahme mit diesem Element vorhanden sein → Erfassung von **Metadaten!**
- Solche Kontaktinformationen können Name und Adresse, Telefonnummer oder E-Mail-Adresse umfassen.
- Solche Listen können in Papierform (ein Kartenverzeichnis für die Mitglieder eines Clubs, ein Telefonverzeichnis) oder digital (eine Datenbank, die ein Register aller Unternehmen enthält) existieren
- Wenn solche Listen nicht verfügbar sind, werden manchmal detaillierte geografische Informationen und Karten verwendet → Mobiles Crowdsensing (GPS, IP)
- Beispiele für Erhebungsfenster können Register des Einwohnermeldeamts sein (bzw. die Liste der Entitäten de Population)
- Aber auch Mitgliederlisten in digitalen sozialen Netzen und Diskussionsforen

### **Unterabdeckung**

- Tritt auf wenn die Zielpopulation Elemente enthält, die im Stichprobenrahmen kein Gegenstück haben.
- Solche Elemente können in der Stichprobe niemals ausgewählt werden.

*Ein Beispiel für eine Unterdeckung ist die Umfrage, bei der die Stichprobe aus einem Bevölkerungsregister ausgewählt wird. Illegale Einwanderer sind Teil der Bevölkerung, werden jedoch im Stichprobenrahmen nie angetroffen.*

*Ein weiteres Beispiel ist eine Online-Umfrage, bei der die Befragten über das Internet ausgewählt werden. In diesem Fall kommt es zu einer Unterdeckung, da Personen keinen Internetzugang haben.*

- Unterabdeckung kann schwerwiegende Folgen haben: Wenn sich die Elemente außerhalb des Stichprobenrahmens systematisch von den Elementen im Stichprobenrahmen unterscheiden, können Schätzungen der Populationsparameter stark verzerrt sein.
- Ein komplizierender Faktor ist, dass es oft nicht sehr einfach ist, das Vorhandensein einer Unterdeckung festzustellen.

### **Überabdeckung**

- Situation, in der der Stichprobenrahmen Elemente enthält, die nicht zur Zielpopulation gehören.

- Wenn solche Elemente in der Stichprobe landen und ihre Daten in der Analyse verwendet werden, können Schätzungen der Populationsparameter beeinflusst (verfälscht) werden.
- Es sollte ziemlich einfach sein, eine Überdeckung im Feld zu erkennen. Dies sollte aus den Antworten auf die Fragen deutlich werden.

## 7.7. Störgrößen und Störfaktoren

- Probleme können auftreten, wenn sich die Einheiten im Stichprobenrahmen von denen in der Zielpopulation unterscheiden. Typisch ist der Fall, wenn eine aus Adressen und die andere aus Personen besteht.
- Weitere Störgrößen bei einer Umfrage können Differenzen zwischen der Zielpopulation und dem Ergebnungsfenster sein, z.B. durch unterschiedliche Erfassungsparameter

*Beispiel: Eine Umfrage, bei der Haushalte mit gleichen Wahrscheinlichkeiten ausgewählt werden sollen und der Stichprobenrahmen aus Personen besteht! Dies kann passieren, wenn die Stichprobe aus einem Bevölkerungsregister ausgewählt wird. Jetzt haben große Familien eine größere Auswahlwahrscheinlichkeit als kleinere Familien, da größere Familien mehr Personen im Stichprobenrahmen haben. Tatsächlich ist die Auswahlwahrscheinlichkeit einer Familie proportional zur Größe der Familie.*

## 7.8. Repräsentative Umfragen

- Eine Reihe von Fragen: Gibt es diese überhaupt? Ist Presse repräsentativ und ausgewogen? Wer oder was repräsentiert? Die Staatliche Institution als Repräsentanz der Bevölkerung? Ganz sicher nicht!

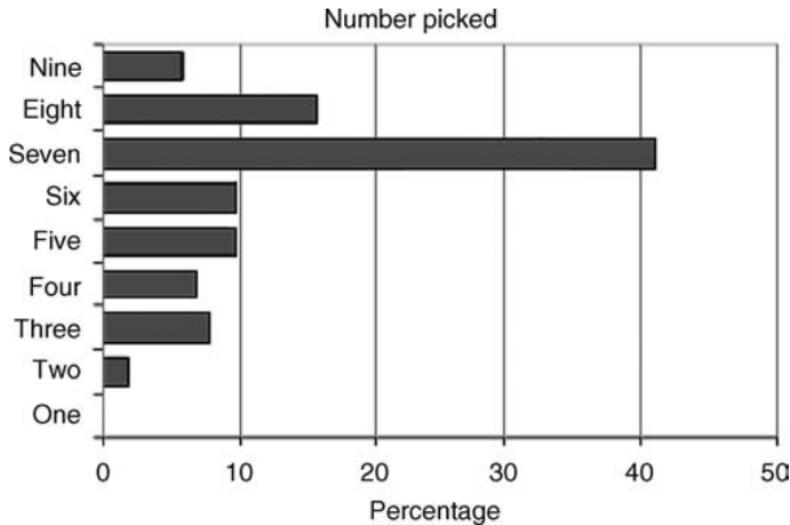
Nach Kruskal and Mosteller gilt folgender Kriterienkatalog:

1. **Allgemeine Anerkennung für Daten.** Es bedeutet nicht viel mehr als eine allgemeine Zusicherung ohne Beweise, dass die Daten in Ordnung sind. Diese Bedeutung von „repräsentativ“ wird normalerweise von den Medien verwendet, ohne zu erklären, was es genau bedeutet.
2. **Fehlen selektiver Einflussnahme.** Bei der Auswahl wurden weder bewusst noch unbewusst Elemente oder Gruppen von Elementen bevorzugt.
3. **Miniaturmodell** der Bevölkerung. Die Stichprobe kann als Maßstabmodell der Bevölkerung angesehen werden. Die Stichprobe hat die gleichen Merkmale wie die Population. Die Stichprobenanteile sind in jeder Hinsicht den Bevölkerungsanteilen ähnlich.

4. **Typische oder ideale Fälle.** Die Stichprobe besteht aus Elementen, die für die Bevölkerung „typisch“ sind. Dies sind „repräsentative Elemente“. (Idee des durchschnittlichen Manns - **Mittelwert**)
5. Erfassung der **Heterogenität** der Bevölkerung. **Variationen**, die in der Population existieren, müssen auch in der Stichprobe angetroffen werden. Die Stichprobe sollte also auch **atypische Elemente** enthalten.
6. **Bedeutung von Begriffen.** Anfangs wird der Begriff einfach verwendet, ohne zu beschreiben, was er ist. Später wird erklärt, was damit gemeint ist.
7. **Statistik.** Es wurde eine spezielle Probenahmemethode angewendet. Es muss eine Form der Wahrscheinlichkeitsstichprobe verwendet worden sein, die jedem Element in der Population gleiche Auswahlwahrscheinlichkeiten gibt.
8. **Gute Schätzung.** Alle Merkmale der Population und die Variabilität müssen wieder in der Stichprobe gefunden werden, damit zuverlässige Schätzungen der Populationsparameter berechnet werden können.
9. **Qualität.** Gut genug für einen bestimmten Zweck. Jede Probe, die zeigt, dass ein Phänomen, von dem angenommen wird, dass es sehr selten ist oder fehlt, mit einer gewissen Häufigkeit auftritt, reicht aus.

## 7.9. Randomisierung

- ▶ Randomisierung kann genutzt werden um eine möglichst gute statistische Verteilung bei einer Umfrage zu erzielen
- ▶ Zufällige Auswahl von Teilnehmern aus einem Register
- ▶ Aber: Menschen sind ganz schlechte Randomisierer



Picking a random number.

[K]

**Figure 35.** Menschen sollen eine Zufallszahl zwischen 1..9 nennen!

- ▶ Ein Randomisierer (Zufallszahlengenerator) ist eine Maschine (elektronisch oder mechanisch) mit folgenden Eigenschaften:
  - ❑ Es kann wiederholt verwendet werden
  - ❑ Es hat  $N$  mögliche Ergebnisse, die mit 1, 2, ...,  $N$  nummeriert sind, wobei  $N$  bekannt ist
  - ❑ Bei jeder Aktivierung wird eines der  $N$  möglichen Ergebnisse erzeugt
  - ❑ Bei jeder Aktivierung sind alle möglichen Ergebnisse gleich wahrscheinlich.
- ▶ Ein Randomizer ist ein theoretisches Konzept. Der perfekte Randomizer existiert nicht. Es gibt jedoch Geräte, die einem Randomizer nahe kommen.
- ▶ Und da ist dann auch schon bei digitalen (synthetischen) Zufallszahlengeneratoren Schluß!
- ▶ Ein digitaler (programmatischer/mathematischer) Zufallszahlengenerator erzeugt eine immer gleiche Sequenz von  $M$  aus  $N$  Zahlen die sich wiederholt!!!!



- Aber: Es gibt statistische Analysemethoden um die Qualität der Randomisierung und von randomisierten Sequenzen zu testen
  - ❑ Autokorrelationsanalyse um Sequenzlängen und Wiederholung zu testen
  - ❑ Einfache visuelle Analyse (Bilderzeugung aus random. Sequenzen) → Muster?
  - ❑ Test auf statistische Verteilung (Gauß, Poisson, usw.)

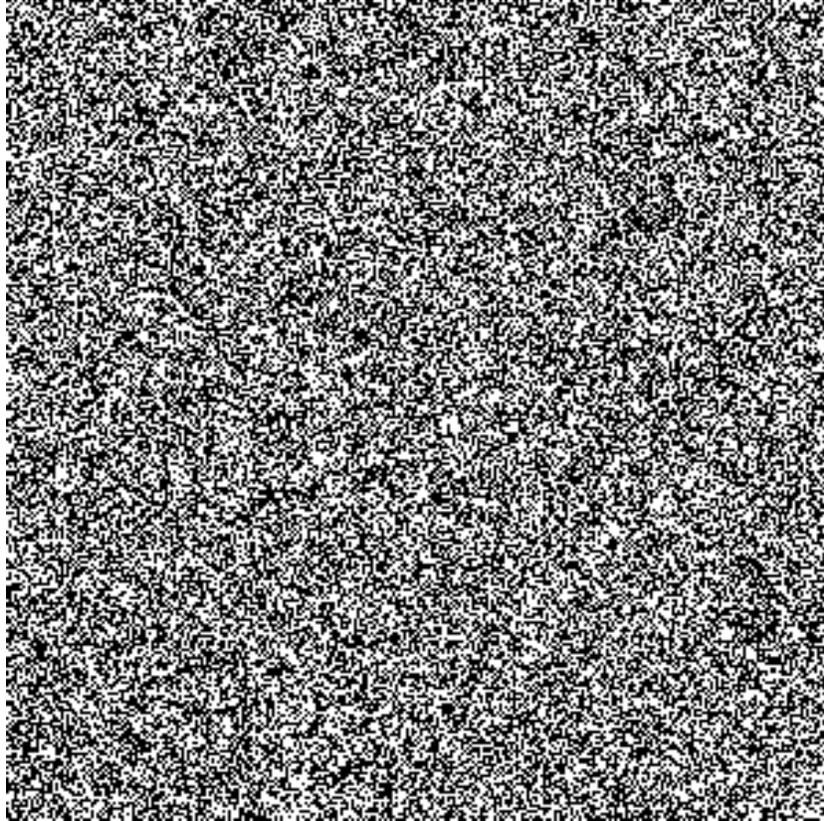
### ***random.org***

Charmaine und Foley empfehlen die folgende Liste aus der NIST suite die auf RANDOM.ORG verwendet wird:

- Frequency Test: Monobit
- Frequency Test: Block
- Runs Test
- Test for the Longest Runs of Ones in a Block
- Binary Matrix Rank Test
- Discrete Fourier Transform (Spectral Test)
- Non-Overlapping Template Matching Test
- Overlapping Template Matching Test
- Maurer's Universal Statistical Test
- Linear Complexity Test

- Serial Test
- Approximate Entropy Test
- Cumulative Sums Test
- Random Excursions Test
- Random Excursions Variant Test
- chi-square test
- test of runs above and below the median
- reverse arrangements test
- An overlapping sums test
- A binary rank test for  $32 \times 32$  matrices

**Beispiel**



---

**Figure 36.** Bitmap von random.org: Die Zufälligkeit kommt von atmosphärischem Rauschen, das für viele Zwecke besser ist als die Pseudozufallszahlalgorithmen, die typischerweise in Computerprogrammen verwendet werden (90000 Zufallswerte).

**Live Bild des Präsentationsbrowsers (*Math.random*)**

- Synthetische (Pseudo) Zufallsgeneratoren berechnen den nächsten “Zufallswert” aus dem vorherigen und es gibt eine (interne) Initialisierung (seed):

$$u_0 = f(\text{seed}), u_{n+1} = f(u_n)$$

- Der Beginn einer Pseudozufallsreihe wird von dem *seed* Wert bestimmt - gleicher *seed* Wert → gleiche Sequenz!

- Ein einfacher Pseudozufallsgenerator der die Samples aus dem Verlauf einer Sinusfunktion entnimmt:

```
var seed = 1; var STEP=10000;
function random2() {
    var x = Math.sin(seed++) * STEP;
    return x - Math.floor(x);
}
```

*Live Bild des Präsentationsbrowsers (my random STEP=10)*

*Live Bild des Präsentationsbrowsers (my random STEP=100000)*

## 7.10. Sequenzen

- Bei Sequenzen werden iterativ nach einander Pseudozufallszahlen erzeugt - die Reihenfolge ist wichtig
- Es sollte keine Gruppen (Teilsequenzen) in der Sequenz geben die sich wiederholen
- Die längste nicht wiederholende Sequenz sollte möglich lang sein (> 100000)

*Beispiel*

## 7.11. Stichproben

- Aus einer Zielpopulation  $U = 1, 2, 3, \dots, N$  werden einzelne Stichproben ausgewählt (direkt durch den Erheber oder indirekt durch den Rücklauf der Umfrage - nicht alle Teilnehmer antworten):
- Eine Stichprobe von einer Population  $U$  ist eine Menge von Indikatoren:

$$a = a_1, a_2, \dots, a_N$$

wobei jeder Indikator  $a_i$  die Anzahl der ausgewählten Entitäten aus  $U$  in der Stichprobe angibt.

- Beispiel: Lottozahlenziehung (Stichprobe, wiederholt):  $U=\{1,2,\dots,49\}$ ,  $a=\{5,2,2,\dots,0\}$
- Die gesamte Stichprobengröße ist also:

$$N = \sum_{k=1}^N a_k$$

- Und ebenso muss für die gesamte Stichprobenwahrscheinlichkeit natürlich gelten:

$$\sum_{k=1}^N p(a_k) = 1$$

so dass gilt:  $0 \leq p(a) \leq 1$ . D.h. der Entwurf einer Stichprobennahme (die Selektion) weist jeder Stichprobe  $a$  aus  $U$  eine Wahrscheinlichkeit  $p(a)$  zu.

- Das Stichprobenauswahlschema beschreibt einen praktischen Algorithmus zum Auswählen von Elementen aus der Grundgesamtheit  $U$  mit einem Zufallsgenerator.
- Die Wahrscheinlichkeit, dass eine Stichprobe ausgewählt wird, muss gleich der Wahrscheinlichkeit  $p(a)$  sein, wie sie im Stichprobenentwurf angegeben ist

### **Beispiele für Stichproben**

Population der Größe  $N=4$ :  
 $U=\{1,2,3,4\}$

Mögliche Stichproben der Größe  $M=2$ :  
(1, 1) (1, 2) (1, 3) (1, 4)  
(2, 1) (2, 2) (2, 3) (2, 4)  
(3, 1) (3, 2) (3, 3) (3, 4)  
(4, 1) (4, 2) (4, 3) (4, 4)

... noch mehr Statistik/Wahrscheinlichkeitsrechnung:

- Die Menge aller Stichprobenwerte  $a$  von  $U$ , die eine Wahrscheinlichkeit  $p(a)$  ungleich Null haben, unter dem Probennamewurf  $p$  ausgewählt zu werden, ist definiert durch

$$A = \{a | p(a) > 0\}$$

- Jedes Stichprobendesign kann durch eine Reihe von Einschlusserwartungen erster, zweiter und höherer Ordnung charakterisiert werden.
- Die Einschlusserwartungen erster Ordnung werden benötigt, um Schätzer für Populationsparameter zu erstellen.

- Die Einschlusserwartungen zweiter Ordnung werden benötigt, um die Genauigkeit von Schätzern zu berechnen.

### **Beispiele von Wahrscheinlichkeiten und Stichproben**

- Ein Stichprobe der Größe 2 wird aus einer Population der Größe 6 genommen.
- Es gibt insgesamt 36 mögliche Stichproben:  $A = \{ (1,1), (1,2), \dots, (6,6) \}$
- Jede der möglichen Stichproben hat die Wahrscheinlichkeit  $p(a) = 1/36$  gezogen zu werden

### **Auswahlwahrscheinlichkeit n-ter Ordnung**

- Die Wahrscheinlichkeit dass das bei der i-ten Auswahl (aus der Gesamtmenge) das Element  $k$  ausgewählt wird

$$p_k^{(i)}$$

- Im einem Beispiel mit der Ziehung einer Zahl aus der Menge  $\{1,2,3,4,5,6\}$  ist  $p^{(1)}$  (erster Ordnung) für jedes Element  $1/6$
- Die Auswahl eines Elements in einer zweiten Ziehung (nach der ersten) ist eine kombinierte Wahrscheinlichkeit, für das Beispiel ist dann:  $p^{(2)} = 1/6 \times 0 + 5/6 \times 1/5 = 1/6$
- Die Wahrscheinlichkeit zwei Element  $k$  und  $l$  hintereinander zu ziehen ist dann eine bedingte Wahrscheinlichkeit:

$$p_{kl}^{(i,j)} = \begin{cases} p_k p_l, & i \neq j \\ 0, & k \neq l \wedge i = j \\ p_k, & k = l \wedge i = j \end{cases}$$

- Im Beispiel die Zahlen 3 und 5:  $p_{35}^{(1,2)} = 1/6 \times 1/5 = 1/30$ .

## **7.12. Schätzung**

*Auswahlwahrscheinlichkeiten können ungleichmäßig verteilt und gestört werden (Bias). Die Auswahlwahrscheinlichkeiten beeinflussen die tatsächliche Aussagekraft der Zielvariablen  $Y$*

- Man kann daher nicht die wirklichen Zielvariablen  $Y$  bestimmen, sondern nur ein **Abschätzung** von  $Y \neq Est(Y)$ !

Die Schätzung eines Populationsparameters ist eine Statistik  $t$ , die zur Schätzung des Populationsparameters  $y$  verwendet wird. Angesichts der Grundgesamtheit  $U$  und des Stichprobenentwurfs  $p$  hängt der Wert der Statistik  $t$  nur von der Stichprobe  $a$  ab. Daher wird manchmal die Notation  $t=t(a)$  verwendet.

- Die Kombination eines Stichprobenentwurfs  $p$  und eines Schätzers  $t$  wird als Stichprobenstrategie  $(p, t)$  bezeichnet.
- Der Erwartungswert eines Schätzers  $t$  für einen Populationsparameter  $y$  ist unter der Stichprobenstrategie  $(p, t)$  gleich:

$$E(t) = \sum_{a \in A} t(a)p(a)$$

- Fazit: Ein Schätzer  $t$  für einen Populationsparameter  $y$  unter der Stichprobenstrategie  $(p, t)$  wird als **unverzerrter Schätzer** bezeichnet, wenn  $E(t)=y$ !

**Die Verzerrung und die Einflussgrößen spielen daher eine große Rolle bei der Erhebung (Planung und Durchführung)**

### **Vertiefung**

- J. BETHLEHEM, Applied Survey Methods A Statistical Perspective. John Wiley & Sons, 2009.

### **Erwartung einer Stichprobeneinbeziehung eines Elements erster Ordnung**

- Die Erwartung (Schätzung)  $\pi_k$  der Einbeziehung eines Elements  $k$  in eine Stichprobe ist gleich der erwartenden Häufigkeit des Elements in einer Stichprobe:

$$\pi_k = E(a_k) = \sum_{a \in A_p} a_k p(a)$$

**Erwartung einer Stichprobeneinbeziehung von Elementen zweiter Ordnung**

- Die Erwartung (Schätzung)  $\pi_{kl}$  der Einbeziehung der Elemente  $k$  und  $l$  in eine Stichprobe ist gleich dem Produkt der erwartenden Häufigkeit der Elemente in einer Stichprobe:

$$\pi_{kl} = E(a_k a_l) = \sum_{a \in A_p} a_k a_l p(a)$$

**Beispiele**

- Wieder das Beispiel einer Stichprobe der Größe 2 aus einer Menge  $U = \{1, 2, 3, 4, 5, 6\}$ .
- Es gibt 36 mögliche Stichproben, 10 davon enthalten eine bestimmte Zahl  $k$  aus  $U$  einmal, eine einzige enthält die Zahl  $k$  zweimal, 25 enthalten die Zahl  $k$  überhaupt nicht
- Es gilt dann  $\pi_k = (1 \times 2 + 10 \times 1 + 25 \times 0) / 36 = 0.333$ .
- Und das es nur zwei Stichproben  $(k, l)$  und  $(l, k)$  aus 36 möglichen gibt ist die Erwartung der Einbeziehung zweier Element  $\pi_{kl} = 2/36!$

## 8. Gestaltung von Umfragen - Kochrezepte

### 8.1. Die Umfrage

1. Der Umfrageprozess beginnt mit der **Formulierung der allgemeinen Forschungsfrage**.
2. **Ziel** der Umfrage ist es, diese Frage zu beantworten.
3. Zu diesem Zweck muss die Forschungsfrage in eine Reihe von **Variablen (sowohl Ziel- als auch Hilfsvariablen)** übersetzt werden, die in der Umfrage gemessen werden sollen.
4. Die Werte dieser Variablen werden verwendet, um eine Reihe relevanter **Populationsparameter** zu schätzen.
5. Zusammen sollten die Werte dieser Parameter einen ausreichenden Einblick bieten.
6. **Fragen** müssen definiert werden, um die Werte der Variablen zu erhalten.
7. Zusammen bilden diese Fragen den **Fragebogen**.

SieheBethlehem,2007,AppliedSurveyMethodsAStatisticalPerspective

## 8.2. Die Fragen

### *Tatsächliche Fragen (Sachfragen)*

- Sachfragen werden gestellt, um Informationen über Tatsachen und Verhalten zu erhalten.
- Es gibt immer einen individuellen wahren Wert.
- Dieser wahre Wert könnte zumindest theoretisch durch andere Mittel bestimmt werden, als dem Befragten eine Frage zu stellen.
- Beispiele für sachliche Fragen sind:
  - ❑ “Wie hoch ist Ihr regulärer Stundensatz für diesen Job?”
  - ❑ “Besitzen oder mieten Sie Ihren Wohnsitz?” Und
  - ❑ “Haben Sie eine Internetverbindung in Ihrem Haus?”
- Die Tatsache, die gemessen werden soll, muss genau definiert werden.
  - ❑ Ein kleiner Unterschied im Fragentext kann zu einer wesentlich anderen Antwort führen.
  - ❑ Beispielsweise kann eine Frage nach der Anzahl der Räume im Haushalt erhebliche Probleme verursachen, wenn nicht klar ist, was einen Raum ausmacht und was nicht. Sollten eine Küche, ein Badezimmer, ein Flur und ein Treppenabsatz enthalten sein?
- **Präzise Fragestellung** und **Vorabtests** durchführen (Sprache ist mehrdeutig)

### *Nicht tatsächliche Fragen (Nicht-sachlich)*

- Nicht-sachliche Fragen fragen nach Einstellungen und Meinungen.
- Eine Stellungnahme spiegelt normalerweise Ansichten zu einem bestimmten Thema wider, beispielsweise das Wahlverhalten bei den nächsten Wahlen.
- Eine Haltung ist ein allgemeineres Konzept, das Ansichten über ein umfassenderes, oft komplexeres Thema widerspiegelt.

- Mit Meinungen und Einstellungen gibt es keinen wahren Wert. Sie messen einen subjektiven Zustand des Befragten, der mit anderen Mitteln nicht beobachtet werden kann.
- Die Einstellung existiert nur im Kopf des Befragten.

*Mobiles Crowdsensing mit Chatbots die Diskussionsforen und Soziale Medien explorieren liefern häufig Antworten (Analysen) auf solche nicht-sachlichen Fragen!*

### 8.3. Antworten

#### *Meinungen*

- Es gibt Situationen, in denen Menschen sich keine Meinung zu einem bestimmten Thema gebildet haben!
- Sie beginnen erst, wenn sie mit der Frage konfrontiert werden, darüber nachzudenken.

*Nach dem gedächtnisbasierten Modell von Zaller (1992) sammeln Menschen alle Arten von Informationen aus den Medien und von ihren Kontakten zu anderen Menschen. Ein Großteil dieser Informationen wird im Gedächtnis gespeichert, ohne darauf zu achten.*

- **Gerade bei opportunistischen Umfragen im MCWS ist dieser Umstand zu beachten!**
- Wenn die Befragten eine Meinungsfrage beantworten müssen, können sie einige der relevanten Informationen abrufen, die im Speicher gespeichert sind.
  - ❑ Aufgrund der Einschränkungen des menschlichen Gedächtnisses wird nur ein Teil der Informationen verwendet.
  - ❑ Dies sind die Informationen, die sofort in den Sinn kommen, wenn die Frage gestellt wird.
  - ❑ Dies sind häufig Informationen, die erst kürzlich im Speicher gespeichert wurden.

## 8.4. Der Fragentext

### **Verständlichkeit**

- ▶ Die Fragen sollten **verständlich** mit der Zielgruppe angepassten **üblichen Vokabular** gestaltet werden!
  - ❑ Beispiel:  
Do you think that food prices are increasing at (1) the same rate as a year ago, (2)
  - ❑ Ökonomen verstehen diese Frage richtig (Aussage in der Frage: die Preise werden steigen!)
  - ❑ Nichtfachkundige verstehen aber eher darunter die Frage ob die Preise steigen oder fallen werden → Missverständnis und Fluch von Multiple-choice Fragen!
- ▶ **Unbestimmte Wörter** wie “normalerweise”, “regelmäßig”, “häufig”, “oft”, “kürzlich” und “selten” müssen vermieden werden, wenn kein zusätzlicher Text vorhanden ist, der erklärt, was sie bedeuten!

### **Eindeutige Fragen**

- ▶ Vermeide mehdeutige Fragen.
  - ❑ Wenn der Fragentext so ist, dass verschiedene Befragte die Frage möglicherweise unterschiedlich interpretieren, sind ihre Antworten **nicht vergleichbar**.
  - ❑ Wenn bei einer Frage beispielsweise nach dem Einkommen gefragt wird, muss klar sein, ob es sich um ein wöchentliches, monatliches oder jährliches Einkommen handelt.
  - ❑ Es muss auch klar sein, ob die Befragten ihr Einkommen vor oder nach Abzug der Steuern angeben sollten.
- ▶ Eine vage Formulierung kann auch zu Interpretationsproblemen führen.

### **Kurze Fragen**

- ▶ Vermeide lange Fragetexte.
  - ❑ Der Fragentext sollte so kurz wie möglich sein.

- ❑ in Befragter, der versucht, eine lange Frage zu verstehen, kann einen Teil des Textes weglassen und somit die Bedeutung der Frage ändern.
- ❑ Lange Texte können auch zu Abölenkung der Befragten führen, was zu einer verminderten Motivation führt, weiterzumachen.
- ❑ Aber: Natürlich sollte der Fragentext nicht so kurz sein, dass er mehrdeutig wird.

*Wenn ein Fragentext zu lang erscheint, kann er in zwei oder mehr kürzere Fragen aufgeteilt werden.*

- Einige Untersuchungen zeigen, dass längerer Fragetext manchmal zu besseren Antworten führt. Längerer Text möglicherweise besser für offene Fragen zu “bedrohlichen” Themen.

*Ein Indikator ist die Wortlänge einer Frage, die Anzahl der Silben in der Fragen und das Verhältnis Silben/Wörter!*

#### **Weitere Do not Do:**

- Vermeide **Leitfragen**. Eine Leitfrage ist eine Frage, die nicht neutral gestellt wird, sondern die Befragten in Richtung einer bestimmten Antwort führt.
  - ❑ Beispiel:  
Do you agree with the majority of people that the quality of the health care in the
  - ❑ Enthält einen Hinweis auf die “Mehrheit der Menschen”. Es deutet darauf hin, dass es sozial unerwünscht ist, nicht zuzustimmen.
  - ❑ Eine Frage kann auch leitend werden, indem die Meinung von Experten in den Fragentext aufgenommen wird.
- Vermeide Dinge zu fragen, die die Befragten **nicht wissen**.
  - ❑ Ein Fragetext kann sehr einfach und völlig eindeutig sein, aber es kann dennoch unmöglich sein, ihn zu beantworten.
  - ❑ Dies kann passieren, wenn die Befragten nach Fakten gefragt werden, die sie nicht (genau) kennen oder abschätzen können.

Beispiel:

How many hours did you listen to your local radio station in the past 6 months?

► Vermeide **sensible Fragen**.

- Sensible Fragen sollten so weit wie möglich vermieden werden.
- Sensible Fragen befassen sich mit Themen, die für die Befragten möglicherweise peinlich sind.
- Solche Fragen können zu ungenauen Antworten führen. Die Befragten können sich weigern, Informationen zu Themen wie Einkommen oder Gesundheit bereitzustellen.
- Die Befragten können auch vermeiden, eine Antwort zu geben, die sozial unerwünscht ist.
- Stattdessen können sie eine Antwort liefern, die sozial akzeptabler ist.

*Aber mittels opportunistischen Umfragen und Mobilten Crowdsensing sowie Anonymisierung könnten solche Fragen eher korrekt beantwortet werden!!*

► Vermeide verschachtelte **Mehrfachfragen**.

- Eine Frage muss jeweils eine Sache erfragen.
- Wenn in einer Frage mehr als eine Sache erfragt wird, ist unklar, was die Antwort bedeutet.
- Beispiel:  
Do you think that people should eat less and exercise more?

► Vermeide **negative Fragen**.

- Fragen dürfen nicht verneint werden, da dies für die Befragten schwieriger zu verstehen ist und einer Wertung enthält.
- Die Befragten können durch eine Frage wie verwirrt werden, wie z.B.:  
Are you against a ban on smoking?  
Would you rather not use a nonmedicated shampoo?
- Letzteres Beispiel zeigt doppelte Verneinung → noch schlechter!

► Vermeide **hypothetische Fragen**.

- ❑ Es ist schwierig für Menschen, Fragen zu imaginären Situationen zu beantworten, da sie sich auf Umstände beziehen, die sie noch nie erlebt haben.
- ❑ Im besten Fall ist die Antwort Rätselfragen und im schlimmsten Fall eine totale Lüge.
- ❑ Beispiel:  
If you were the president of the country, how would you stop crime?

## 8.5. Antworttypenklassen (Formularlayout)

### *Geschlossene Fragen*

- Eine geschlossene Frage wird verwendet, um qualitative Variablen zu messen.
1. Binäre Aussagen : { wahr, falsch }
    - Beispiel:  
Q: Ist die Erde eine Kugel  
A: {wahr, falsch}
  2. Mehrfachauswahl von kategorischen Symbolen (statisch) : {  $case_1, case_2, \dots, case_n, others$  }
    - Beispiel:  
Q: Welche Farbe hat der Himmel im Sommer?  
A: {rot,blau,grün,unbekannt}
    - Die Abfrage eines *others* Falles ist wichtig!
    - Es gibt mutual exclusive Mehrfachauswahl (nur einer Auswahl möglich) oder multiple Auswahlen im Bereich  $\&lbrack;i,k&rbrack;$
  3. Metrische und numerische Werte (i.A. innerhalb eines vorgegeben Intervalls und Genauigkeit):  $x=\&lbrack;a,b&rbrack;(step)$ 
    - Beispiel:  
Q: Wie alt sind Sie?  
A: [1,99] (1)

### **Offene Fragen**

4. Offene Fragen und Freiformantworten (Text) mit begrenzter Länge:  
Text(min/max length)

► Beispiel:

Q: Beschreiben sie kurz Ihren letzten Traum (20-500 Worte)  
(huh! böse, sensible Frage!)

A: "...."

## **8.6. Fragenkataloge**

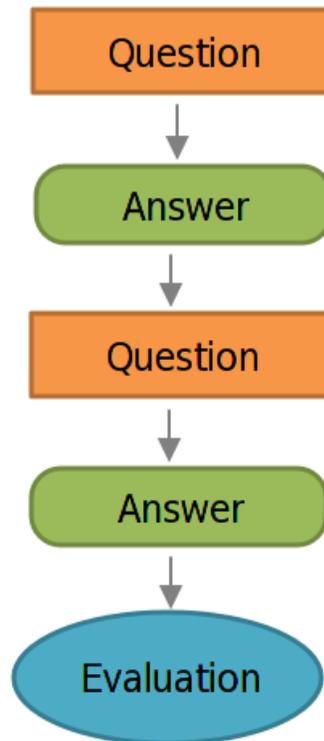
### **Statische Fragenkataloge**

- Die Reihenfolge und Auswahl (Sichtbarkeit) von Fragen ist statisch vorgegeben
  - D.h. jede Frage wird unabhängig von vorherigen Antworten gestellt

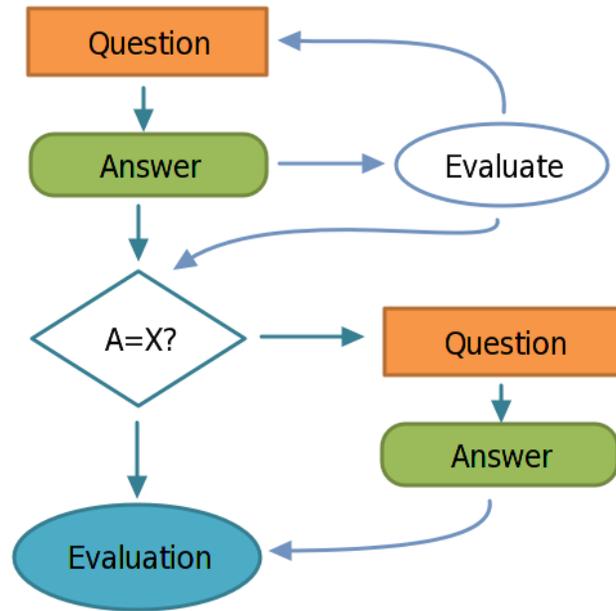
### **Dynamische Fragenkataloge**

- Die Reihenfolge und Sichtbarkeit von Fragen kann von vorherigen Antworten abhängen
- Nach einer Antwort kann ein dynamischer Überblick über die Antworten anderer Befragter gezeigt werden mit der Möglichkeit die eigene Antwort zu revidieren! (Backtracking)

### **Statisch**



Dynamisch



## 8.7. Auswahl der Stichprobe

```
1: Ingredients   Population size N
2:              Sample size n
3:              Random values u from the interval [0, 1)
4:
Step 1          Fill a vector v of length N with the numbers from
5:              1 to N: v[1]=1, v[2]=2, ..., v[N]=N
6: Step 2       Set the counter i to 1
7: Step 3       Compute a random value u from [0, 1)
8:
Step 4          Compute the sequence number  $k = [i + (N - i + 1) * u]$ .
9:
                The square brackets denote rounding down to the nearest integer
10:
Step 5          Exchange the values of elements v[i] and v[k]
11: Step 6      If i is smaller than n,
12:
                increase the value of i with 1 and go back to step 3
13:
Step 7          If the value of i is equal to n, sample selection is ready.
14:
                The first n elements v[1], v[2], .. ., v[n] of the vector v contain
15:
                the sequence numbers of the sampled elements
```

## 9. Crowdsensing Plattformen

### 9.1. Zentralistische Netzwerke

- Klassisch werden Klienten-Server Architekturen verwendet → Baum- oder Sterntopologie

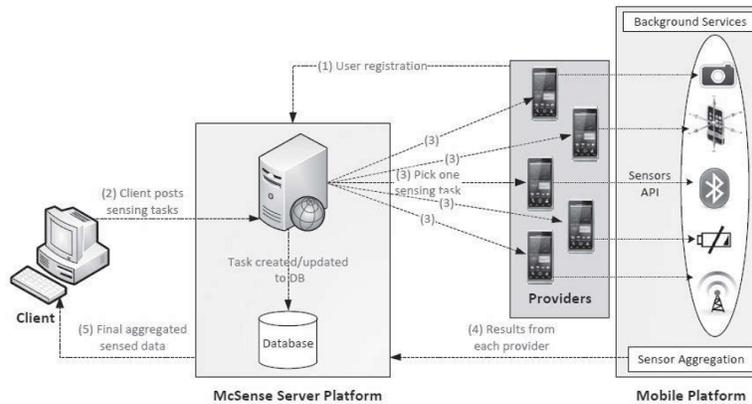


Figure 37. Beispiel: Die McSense Architektur mit zentraler Serverinstanz [A]

### Intelligenten Transportdienste

- Crowdsensing Plattformen als Software Middleware Schicht in Intelligenten Transportdiensten (ITS)
  - ❑ Ziele: Energie- und Schadstoffreduktion, Kostenreduktion, kürzere Transportzeiten usw.
- Wesentliche Komponenten:
  - ❑ Datenaggregation und Serviceschnittstellen (Informationsgewinnung)
  - ❑ DaFP: Data Forwarding Plane → Datentransport
  - ❑ SerMoN: Service Mediating and Monitoring System → Datenaggregation und Data Mining

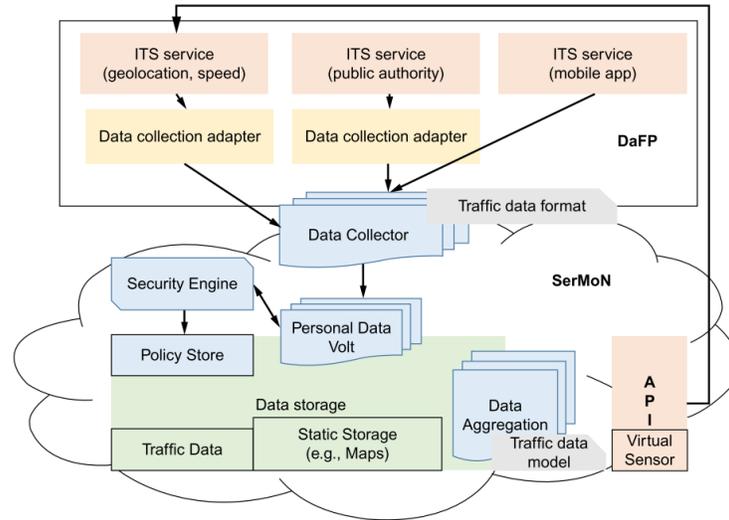


Figure 38. Beispiel: MobiWay Architektur [B]

**Katastrophenmanagement**

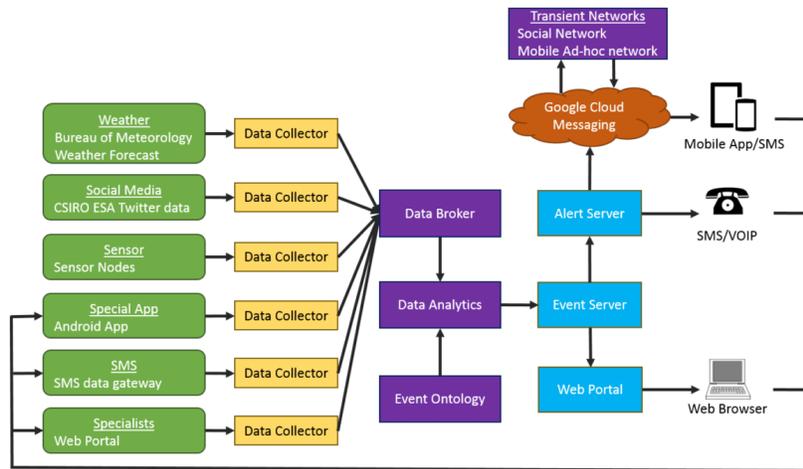


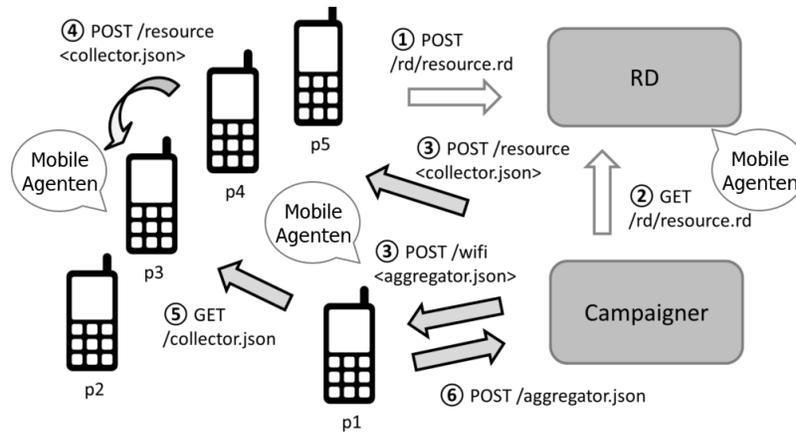
Figure 39. MobiWay Architektur [2]

### **Mobile Agenten**

- ▶ Verwendung von Mobilten Agenten (Multiagentensystem MAS) für die Datenaggregation [3]
- ▶ Komponenten und Services:
  - RD: Ressource Directory → Verzeichnisdienst → Organisation
  - Campaigner: Steuert die Sensorerhebungsaufgaben
- ▶ Es gibt eine Reihe von auf HTTP basierende Agenteninteraktionsprotokolle, um Operationen und Konversationen von mobilten Agenten im MAS zu ermöglichen.

### **Ablauf**

1. Die Geräte registrieren sich im RD .
2. Der Campaigner sucht Ressourcen für die Kampagne durch Suche von Ressourcen im RD
3. Er startet mobile Agenten, um die Kampagne auszuführen, und überwacht die Kampagnenausführung anhand der von den mobilten Agenten zurückgegebenen Ergebnisse (6).
4. Mobile Agenten migrieren automatisch im System, und
5. Die Agenten arbeiten zusammen, um ihre Aufgaben auszuführen.
6. Rückgabe der Ergebnisse



**Figure 40.** Crowdsensing mit Mobilten Agenten über eine HTTP Kommunikationsschnittstelle [3]

## 9.2. Energieeffizientes Mobiles Crowdsensing

- Plattformen bestimmen auch den Energiebedarf beim mobile Crowdsensing

Test	Power [mW]	Time (Energy) to get position [s] (J)				Time needed to shutdown [s]	Precision [m]	Battery life [hour]
		< 10s	1min	10min	1h			
<b>Positioning methods</b>								
1. GPS	~356	3 (1.07)	30 (10.7)	40 (14.2)	200 (71.2)	30	10	11.5
2. AGPS 3G	~973	11 (10.7)				30	10	11.5
3. AGPS 2G	~765	12 (9.18)				30	10	11.5
4. Net 2G	~663	3 (1.99)	6 (3.98)	6 (3.98)	6 (3.98)	6	100 - 1000	6.2
5. Net 3G	~1150	3 (3.45)	5 (5.75)	5 (5.75)	5 (5.75)	3	100 - 1000	3.7
<b>Sensors</b>								
6. Accelerometer	~124	NA				0	NA	28.1
7. WLAN	~800	0.85 (0.68)				0	100	38.7
8. Cell-ID	~0	0				0	1000	58 - 75
<b>Standby</b>								
9. Online 2G	~60							74.8
10. Online 3G	~77							58.3
11. Offline	~36							124.7

**Figure 41.** Energiebedarf eines Smartphones bei bestimmten Einsatzszenarien und Geräteverwendung [H]

## 9.3. Identifikation

- Eine wichtige Rolle beim Crowdsensing und Crowdsourcing spielt die Identifikation von Teilnehmern / Nutzern [H].

- In der Regel muss die Plattform geeignete Mittel und Operationalität zur Verfügung stellen, um
  - ❑ Authentifizierung → Identitätsfeststellung, und
  - ❑ Autorisierung → Verknüpfung Identität-Rollen/Befugnis zu ermöglichen.
- Statische versus dynamische Identität (Pseudonyme)
  - ❑ Geräte oder geräteidentifikationsgebundene Identitäten (automatisch, aber unsicher) → Mobile Geräte: IMEI Nummer (Gerät) oder Telefonnummer
- Schutz gegen Betrug!
- Schutz gegen Angriffe:
  - ❑ Denial-of-Service Angriffe
  - ❑ Man-in-the-middle Angriffe → Gerade bei Identifikationsverfahren bedeutend

Vor- und Nachteile von statischen und dynamischen Identitäten (Pseudonyme)

### **Statische (echte) Identität**

Der Vorteil der festen Identität ist ihre Sicherheit, solange der Authentifizierungsprozess garantiert, dass ein Benutzer wirklich derjenige ist, der er zu sein vorgibt.

Der Nachteil ist die Zentralisierung und ein möglicher Datenverlust. Wenn eine feste Identität erstellt wird, werden zahlreiche eindeutige Identitätsattribute an die Identität angehängt, um sicherzustellen, dass die Online-Identität gut mit der Offline-Identität des Benutzers verbunden ist. Daher hängt eine feste Online-Identität von anderen realen festen Identitäten ab. Mit der ständigen Bedrohung durch Hacker, die versuchen, Benutzerinformationen zu stehlen, werden Authentifizierungsprozesse für feste Identitäten immer komplexer.

### **Dynamische (pseudonymisierte) Identität**

Der Vorteil der Verwendung von Pseudonymen ist die Einfachheit. Daher erleichtert das Pseudonym den Benutzern den Zugriff auf Websites und ihre Dienste. Ein weiterer Vorteil eines Pseudonyms ist, dass es die tatsächliche Identität seines Besitzers schützt. Es ist vergleichbar mit dem Tragen einer Maske im Internet.

Menschen können mehrere Personas annehmen, die unterschiedlichen Bedürfnissen entsprechen. Zum Beispiel kann ein Benutzer ein Konto für

sein Geschäftsleben und ein anderes für das soziale Leben erstellen. Auch in Gesundheitsarchitekturen werden Pseudonyme verwendet und die Privatsphäre der Patienten besser geschützt.

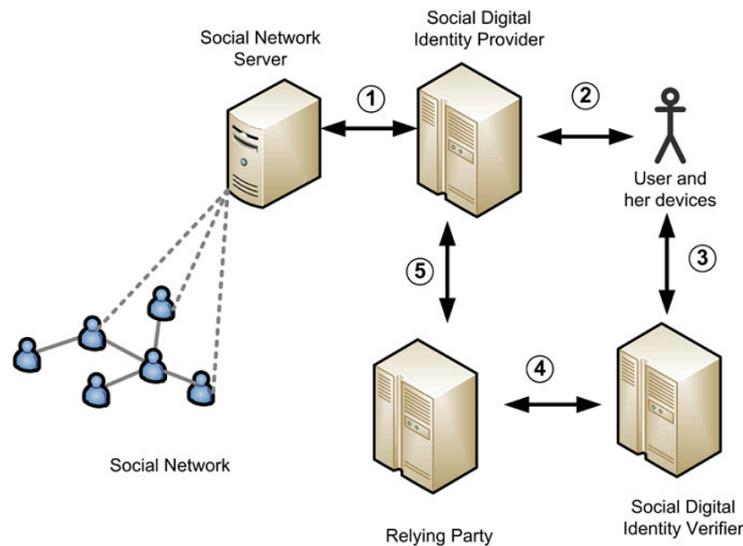
### Dynamische (pseudonymisierte) Identität

Die Verwendung von Pseudonymen hat den Nachteil, dass sie leicht missbraucht werden können. Es bietet böswilligen Benutzern die gleiche Sicherheit wie den legitimen Benutzern. Anders als rechtmäßige Benutzer ändern böswillige Benutzer immer ihre Konten, sodass Websites oder andere Benutzer ihre Aktivitäten kaum nachverfolgen können. Es können jedoch rechtmäßige Benutzer verfolgt werden, die bei einem oder wenigen Konten bleiben.

Weiterhin können Pseudonyme durch Data Mining wieder personalisiert werden! Die Verwendung von "human flesh search engines" macht dieses möglich. Häufig kann über Transaktion die pseudonymisierte Nutzer durchführen auf realen Identitäten (Sensorfusion) geschlossen werden.

### Konnektive und Soziale Identitäten

Die Nutzbarmachung der Konnektivität von Nutzern, z.B. in sozialen Netzwerken, kann zu einer Verbesserung der sicheren Identifikation von Nutzern führen (und z.B. auch Mehrfachidentitäten auffinden). Die **Konnektivitätsanalyse** ist ein wichtiges Hilfsmittel im Crowdsensing.



**Figure 42.** Beispiel einer Architektur für soziale Identifikation in Sozialen Netzwerken [H]

## 9.4. Implementierungsprinzipien

Die allgemeinen Prinzipien, die bei der Implementierung mobiler Crowd Sensing-Systeme beachtet werden sollten, sind:

1. Es sollte Anwendungsentwicklern ermöglichen, ihre Datenanforderungen der Erhebung in einer Hochsprache festzulegen.
2. Es sollte den allgemeinen Datenbedarf in allen Anwendungen ermitteln, um doppelte Erfassungs- und Verarbeitungsaktivitäten auf Geräten zu vermeiden.
3. Es sollte automatisch die Menge von Geräten identifizieren, die die gewünschten Daten bereitstellen können, und Anweisungen zur korrekten Konfiguration der Erfassungsaktivitäten auf den Geräten erzeugen.
4. Bei dynamischen Änderungen sollten die ausgewählten Geräte und Erfassungsanweisungen angepasst werden, um die gewünschte Datenqualität sicherzustellen.
5. Um zu vermeiden, dass verschiedene Versionen lokaler Analysen auf heterogenen Geräten implementiert werden, muss eine Schicht vorhanden sein, die die Unterschiede in den physischen Sensorzugriffs-APIs verbirgt und dieselbe API nach oben weitergibt.
  - ▶ Dadurch ist es möglich, dieselben lokalen Analysen auf verschiedenen Geräteplattformen wiederzuverwenden

## 9.5. Implementierungsrandbedingungen

Die Hauptherausforderungen für die Plattformarchitektur und Software sind:

- ▶ Verwaltung der Beteiligung der Benutzer am System (partizipatorisches vs. opportunistisches Wahrnehmen);
- ▶ Verwaltung des Vertrauens in die Benutzer, um das gesamte System nicht zu beeinträchtigen;
- ▶ Anpassung an die sich ändernde Ressourcenverfügbarkeit und den Stichprobenkontext eines Geräts;
- ▶ Umgang mit Ressourcenbeschränkungen für mobile Erfassungsgeräte;
- ▶ Bewältigung der Beweglichkeit von Erfassungsgeräten (z.B. wenn nicht genügend mobile Sensoren vorhanden sind oder sich der Sensor bewegt und die Probenahme für einen bestimmten Kontext gefährdet ist);
- ▶ Ermöglichen, dass Geräte sensorische Daten gemeinsam nutzen und gleichzeitig die Privatsphäre der Benutzer schützen;

- Zusammenstellung und Optimierung verschiedener Ressourcen und anwendungsrelevanter Metriken zur Definition von Methoden zur Datenerhebung und -verbreitung;
- Verwaltung großer Mengen generierter Daten;
- Definition eines Ansatzes zum Sammeln von Daten, mit dem die Genauigkeit von Algorithmen beurteilt werden kann, die Sensordaten interpretieren;
- robuste und genaue Aktivitätsklassifizierung in einer dynamischen Echtzeitumgebung durchführen;
- Bereitstellung nützlicher Informationen für die Benutzer;
- Skalierung des Sensorsystems von einer persönlichen bis zu einer Bevölkerungsskala;

## 10. Sensorische Architekturen

### 10.1. Sensoraggregation

In sensorischen Systemen werden Sensordaten in verschiedenen Ebenen verarbeitet:

#### ***Vertikale Ebenen***

##### **Perzeption**

Hier findet die Akquisition der rohen Sensordaten statt. Die Sensoren sind räumlich verteilt und werden lokal vorverarbeitet.

##### **Aggregation**

Einzelne Sensordaten werden zeitlich und räumlich zusammengeführt und gesammelt (Sensorfusion)

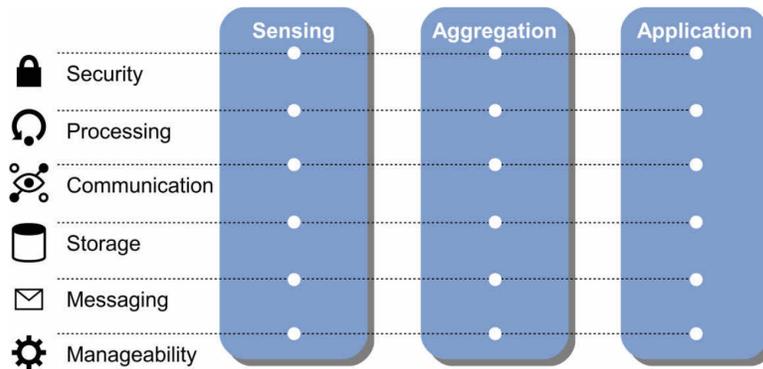
##### **Applikation**

Die gesammelten Daten werden nutzbar gemacht: Weitere Datenverarbeitung, Aufbereitung, Eigenschaftsselektion (Feature Selection), Informationsgewinnung (Feature Extraction), Visualisierung

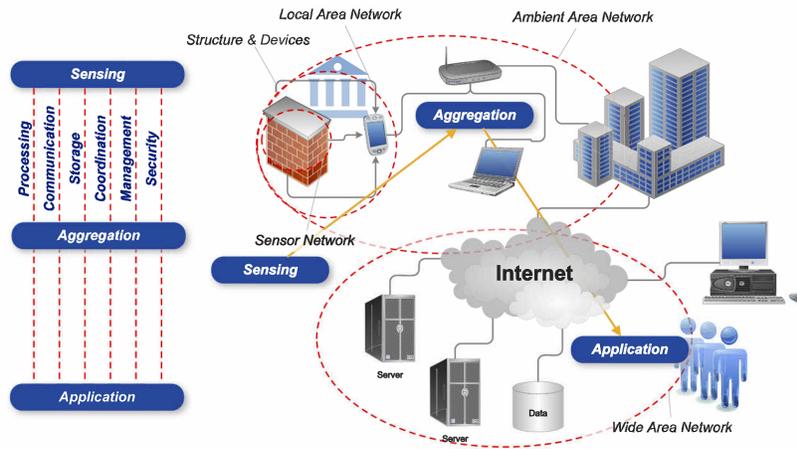
#### ***Horizontale Ebenen***

- Die horizontalen Ebenen durchziehen alle vertikalen Ebenen:

1. Sicherheit
2. Datenverarbeitung
3. Kommunikation
4. Datenspeicherung
5. Nachrichtenvermittlung
6. Management



**Figure 43.** Grundlegender Zusammenhang der horizontalen und vertikalen Ebenen in Sensorischen Systemen



**Figure 44.** Räumliche Abbildung der vertikalen Ebenen auf Cloud Computing

### **Mobile Agenten**

- Mobile Agenten können die Datenverarbeitung und Kommunikation auf allen drei vertikalen und allen sechs horizontalen Schichten übernehmen!
- Man unterscheidet zwei wesentliche Sensordatenverarbeitungsparadigma:
  - ❑ Online (Echtzeit) Verarbeitung
  - ❑ Offline Verarbeitung
- Agenten daher weiterhin eine Verbindung zwischen Online- und Offline Verarbeitung herstellen



**Figure 45.** Mobile Agenten in stark heterogenen Umgebungen

### **10.2. Lokalisation**

- Die Lokalisation von Nutzern wird häufig über die Erfassung der Netzwerkkonnektivität in Verbindung mit weiteren Sensordaten wie GPS bestimmt
  - ❑ Annahme: Personalisierte Geräte
  - ❑ Netzwerk lokalisierbar
- Aber: Mobiles Internet hat beliebige geografische Einwahlpunkte in das Internet → häufig unbrauchbar!

## Aufgabe

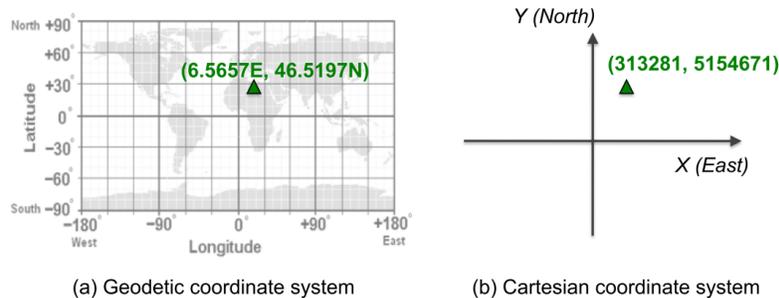
1. Mittels eigenen Smartphone IP Lokalisierung durchführen (kann auch mit der *JAMapp* durchgeführt werden)

➤ Wo seid ihr scheinbar?

- 
- Semantische Daten, z.B. von benutzten Suchmaschinen oder besuchte WEB Seiten können einen besseren Anhaltspunkt für den aktuellen Aufenthaltsort liefern. Aber wie kommt man an diese Informationen?

## GPS

- GPS Koordinaten werden in Längen- und Breitengraden angegeben (Kugel- und korrigiertes Ellipsoidmodell)
- Um auf lokale Nähe und Konnektivität schließen zu können ist i.A. eine Transformtion in kartesische Koordinaten erforderlich



**Figure 46.** Datentransformation von geodätischen GPS Daten in kartesisches Koordinatensystem:  $longitude, latitude \rightarrow x, y$

## 11. Skalierbarkeit und Robustheit

### 11.1. Skalierbarkeit

- Da es sich bei den mobilen Crowdsensinganwendungen um **verteilte Systeme** handelt spielt die Skalierbarkeit eine große Rolle
- Es gibt eine Lücke zwischen Test und realen Einsatz

- ❑ Testumgebung besteht z.B. aus 10 Geräten und 8 Nutzern → festgelegte statische Randbedingung.
- ❑ Reale Einsatzumgebung besteht variierend aus 100-100000 Geräten und bis zu 100000 Nutzern → nicht festgelegte dynamische Randbedingung
- Wenn es zentrale Instanzen gibt hat das Einfluss auf folgende Metriken:
  - ❑ Bearbeitungslatenz → Relativ da die “Nutzerlatenz” auch im Bereich von 100-1000 ms liegen kann und Verzögerungen in der Verarbeitung bei Umfragen nicht unbedingt auffallen
  - ❑ Datendurchsatz
  - ❑ Stabilität (Überlastung)
- Netzwerke stellen auch teilzentrale Instanzen dar!

## 11.2. Robustheit

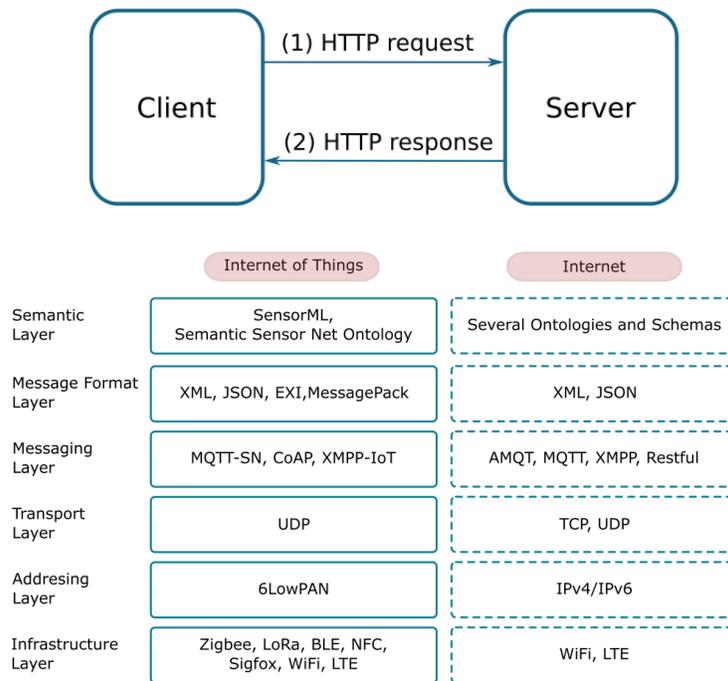
### *Single-point-of-Failure (SPoF)*

- Ein zentrale Instanz stellt immer ein SPoF dar und sollte unbedingt vermieden werden
  - ❑ Eine Cloudumgebung stellt sich für den Nutzer zwar wie eine Maschine dar, besteht aber tatsächlich aus einer Vielzahl von Rechnern und ggfs. auch Netzwerken
- Auch Netzwerkkomponenten inkl. lokales Mobilfunknetz können SPoF darstellen
- Werden Agenten verwendet kann der SPoF durch den selbstorganisierenden und selbstüberwachenden Einsatz eines Multiagentensystems eventuell kompensiert werden
- Beim Einsatz von mobilen Geräten in mobilen Netzwerken muss die Kommunikation und somit auch die Crowdsensinganwendung mit Kommunikationsverzögerungen und Fehlern als Normalfall umgehen können → Delay tolerant Network (DTN)

## 12. Agenten

### 12.1. Motivation

- Crowdsensing ist verteilte Datenverarbeitung
- **Verteilte Datenverarbeitung** bedeutet **verteilte Kommunikation zwischen Prozessen**
- Traditionelle verteilte Netzwerke wie das Internet der Dinge (IoT) müssen mit einer Vielzahl von Kommunikationsprotokollen und Netzwerkstrukturen umgehen können → **Stark heterogene Systeme**
- **Datenrepräsentation** ist eine weitere Hürde in solchen verteilten Systemen
- Häufig HTTP basierte Klienten-Server Kommunikation (zentrale Serverinstanz)



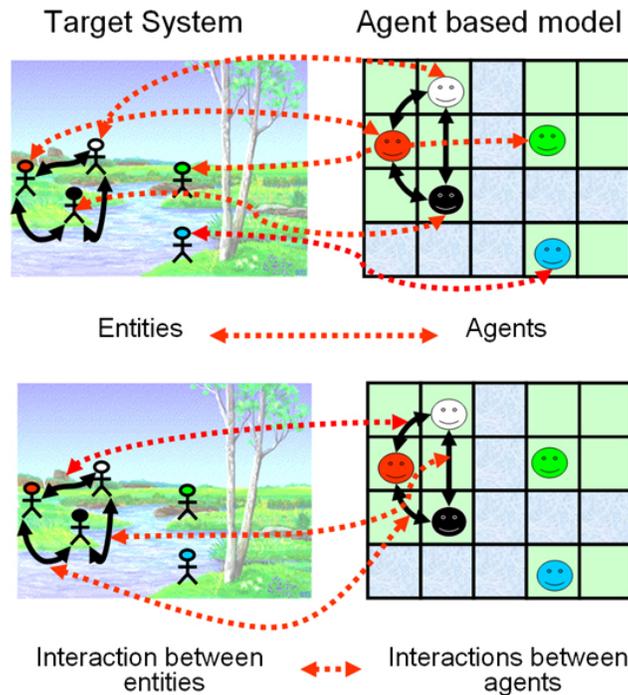
**Figure 47.** Große Diversität an IoT Protokollen und Datenrepräsentation für die Kommunikation → nicht einheitlich [4]

## 12.2. Agenten

Agenten besitzen eine Vielzahl von Fähigkeiten, die sie von klassischen Programmen unterscheiden - obwohl Agenten auch Programme sein können!

### *Merkmale*

- Fähigkeit zu eigenständiger Aktivität (**Nicht Nutzeraktiviert**)
- Autonomes, "selbstbestimmtes" Verhalten (**Nicht durch zentrale Instanz gesteuert**)
- Fähigkeit zum selbstständigen Schlussfolgern (**Umgang mit unsicheren Wissen**)
- Flexibles und rationales Verhalten (**Adaptivität an veränderliche Weltbedingungen**)
- Fähigkeit zu Kommunikation und Interaktion (**Synchronisation**)
- Kooperatives oder konkurrierendes Verhalten (**Lösung von Wettbewerbskonflikten**)
- Fähigkeit zur ziel- und aufgabenorientierten Koordination (**Kooperation**)
- Man unterscheidet zwischen realen und virtuellen Welten;
- Agenten können natürliche (reale) Welten abbilden oder in realen Welten agieren



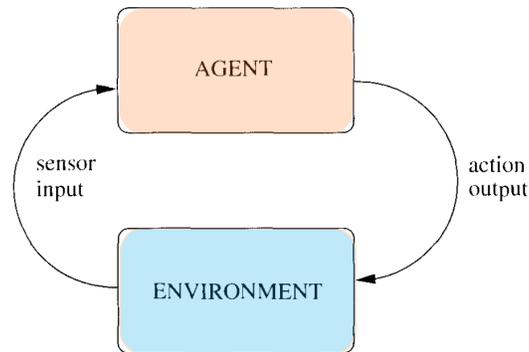
[Galan,2009]

**Figure 48.** Beziehung realer natürlicher Welt mit Lebewesen zu virtueller Welt mit Agenten

### **Agentenmodell**

- Ein Agent in seiner Umgebung nimmt sensorische Eingaben aus der Umgebung auf und produziert Ausgabe-Aktionen, die ihn *und* die Umgebung beeinflussen.
- Die Interaktion ist gewöhnlich fortlaufend und nicht-terminierend!

### **Einfachstes Agentenmodell**



[J]

### **Beispiel**

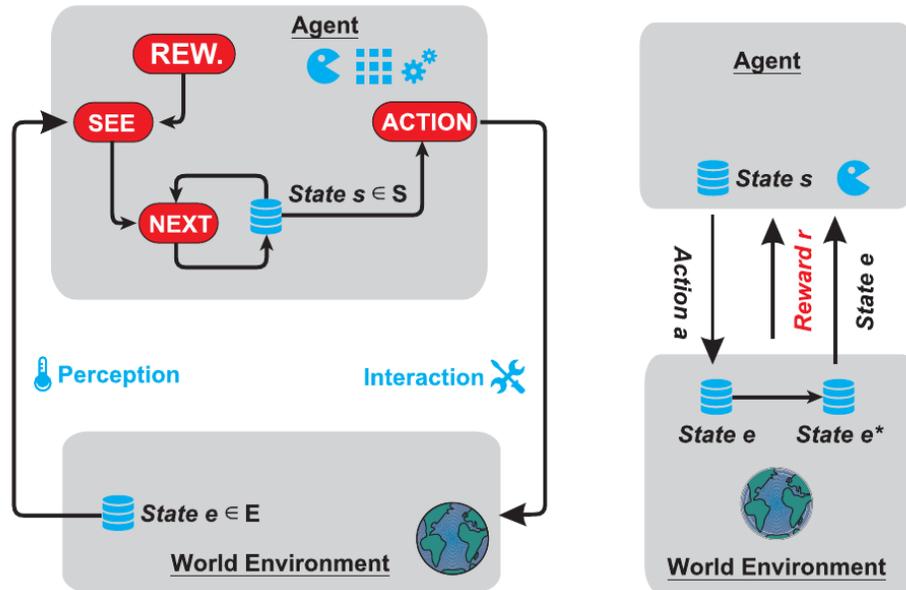
Umgebung: Raum in Gebäude  
Sensor: Temperatur T  
Aktor: Heizung

Verhalten:

- (1) Temperatur zu niedrig → Heizung an
- (2) Temperatur angenehm → Heizung aus

### **Reaktive und Zustandsbasierte Agenten**

- Reaktive und zustandsbasierte Agenten führen einen Zyklus durch:
  - ❑  $\text{Perzeption} \Rightarrow \text{Verarbeitung} \Rightarrow \text{Zustandsänderung} \Rightarrow \text{Entscheidung} \Rightarrow \text{Aktion}$



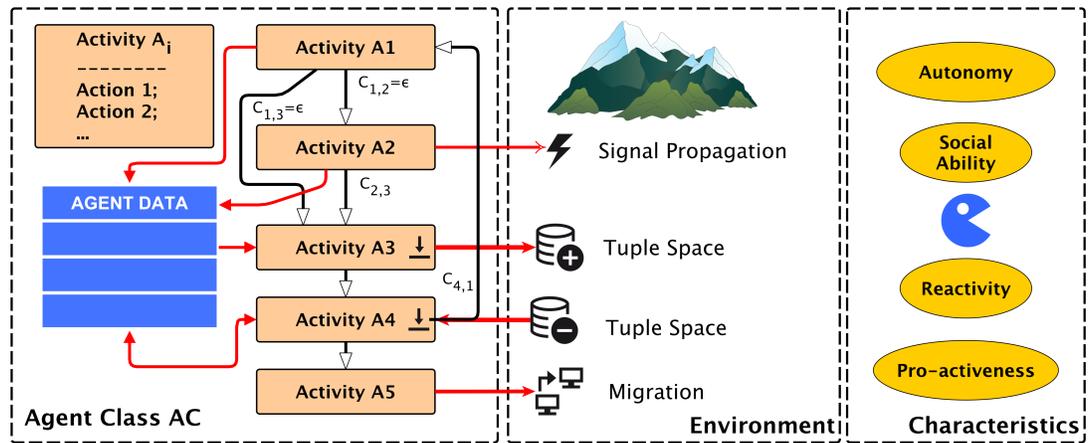
### 12.3. ATG Modell

#### *Zustandsbasierter Agent*

- Besteht aus: (1) Körpervariablen (2) Kontrollzustand und Verhalten

#### *Aktivität und Zustand*

- Das Verhalten eines aktivitätsbasierten Agenten ist durch einen Agentenzustand gekennzeichnet, der durch Aktivitäten verändert wird.
- Aktivitäten verarbeiten Wahrnehmungen, planen Aktionen und führen Aktionen aus, die den Steuerungs- und Datenzustand des Agenten ändern.
- Aktivitäten und Übergänge zwischen Aktivitäten werden durch einen Aktivitätsübergangsgraphen (Activity Transition Graph, ATG) dargestellt.
- Die Übergänge starten Aktivitäten in der Regel abhängig von der Auswertung von Agentendaten (Körpervariablen), die den Datenzustand des Agenten repräsentieren.



**Figure 49.** Links) Agentenverhalten, das von einem Aktivitätsübergangsgraphen vorgegeben ist, und die Interaktion mit der Umgebung, die durch Aktionen erfolgt, die in Aktivitäten ausgeführt werden (Rechts) Agentenmerkmale

- Ein Aktivitätsübergangsgraph, der mit Agentenklassen assoziiert ist, besteht aus einer Menge von Aktivitäten  $\mathbb{A} = \{A_1, A_2, ..\}$  und einer Menge von Übergängen  $\mathbb{T} = \{T_1 (C_1), T_2 (C_2), ..\}$ , die die Kanten des gerichteten Graphen darstellen.
- Die Ausführung einer Aktivität, die selbst aus einer Folge von Aktionen und Berechnungen besteht, ist mit dem Erreichen eines Unterzieles oder das Erfüllen einer Voraussetzung verknüpft, um ein bestimmtes Ziel zu erreichen, z. B. Sensordatenverarbeitung und Verteilung von Informationen.
- Normalerweise werden Agenten verwendet, um komplexe Aufgaben zu zerlegen basierend auf der Zerlegung durch MAS.
- Agenten können ihr Verhalten basierend auf Lern- und Umgebungsänderungen oder durch Ausführen einer bestimmten Unteraufgabe mit nur einer *Untermenge* des ursprünglichen Agentenverhaltens ändern → **Dynamische ATG**.
- Das ATG-Verhaltensmodell ist eng mit der Interaktion von Agenten mit deren Umgebung verbunden, hier hauptsächlich durch
  - ❑ Den Austausch von Daten unter Verwendung einer Tupelraum-Datenbank;
  - ❑ Durch Migration; und durch
  - ❑ Die Weitergabe von Nachrichten zwischen Agenten mittels Signalen.

- Replikation und Instantiierung von Agenten

## 12.4. DATG Modell

- Ein ATG beschreibt das vollständige Agentenverhalten.
- Jedes Unterdiagramm und jeder Teil des ATG kann einem Unterklasseverhalten eines Agenten zugeordnet werden.
- Daher führt das Modifizieren der Menge von Aktivitäten  $\mathbb{A}$  und Übergängen  $\mathbb{T}$  des ursprünglichen ATG zu mehreren Unter- und Oberverhalten, die Algorithmen implementieren, um verschiedene unterschiedliche Ziele zu erfüllen.
- Die Rekonfiguration der Aktivitäten führt zu einer Menge von Aktivitätsmengen  $\mathbb{A}^* = \{\mathbb{A}_i \subset \mathbb{A}, \mathbb{A}_j \subset \mathbb{A}, \mathbb{A}_k \supset \mathbb{A}, \dots\}$ , die von der ursprünglichen Menge  $\mathbb{A}$  abgeleitet sind, und die Modifikation oder Rekonfiguration von Übergängen  $\mathbb{T}^* = \{\mathbb{T}_1 \subset \mathbb{T}, \mathbb{T}_2 \subset \mathbb{T}, \dots\}$  ermöglicht die dynamische ATG-Zusammensetzung (Komposition) und Agentenunterklassifizierung zur Laufzeit,

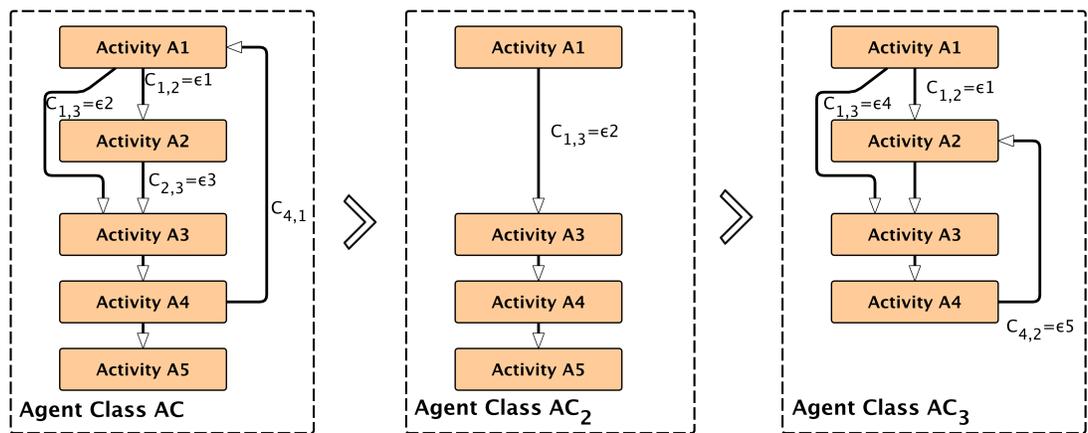
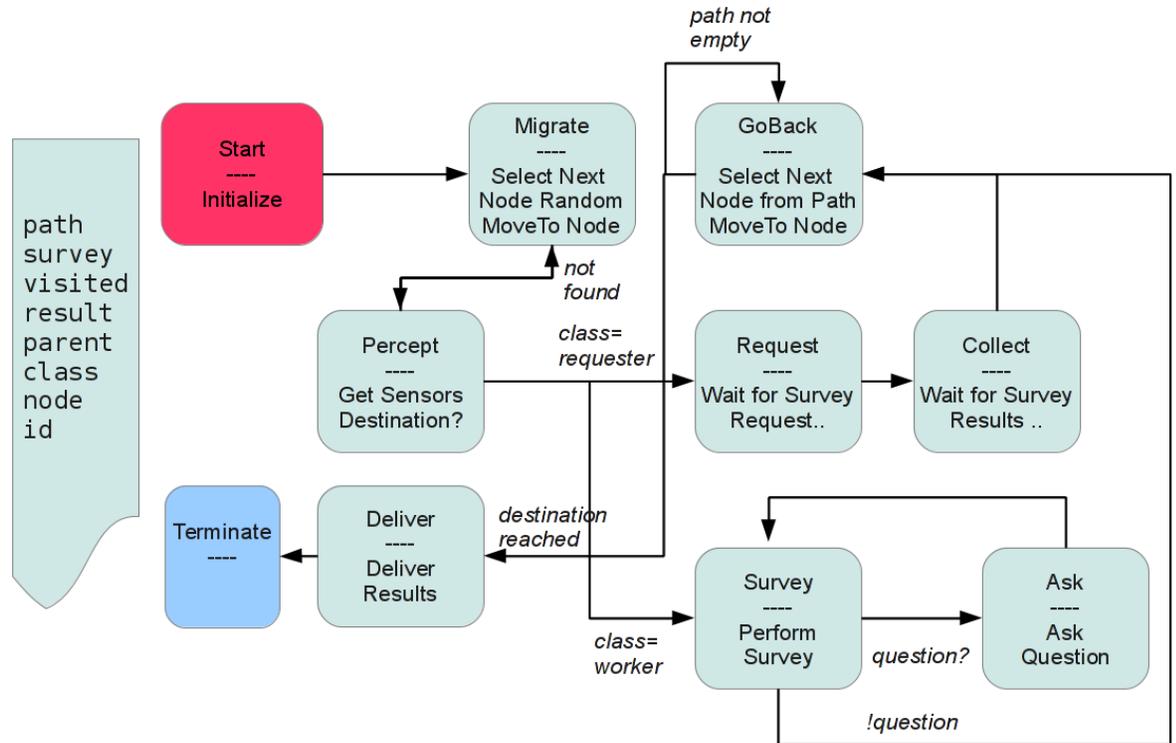


Figure 50. Dynamischer ATG: Transformation und Komposition

## 12.5. Beispiel eines Agenten

ATG eines einfachen Umfrageagenten mit Unterklassen (*master*, *requester*, *worker*)



## 12.6. Agentenklassen

Eine Agentenklasse beschreibt ein bestimmtes Verhalten eines Agenten mittels eines ATG und einer Menge von Körpervariablen.

- Von einer Klasse können zur Laufzeit Agenten instanziiert (erzeugt) werden.

### Verhalten

Eine bestimmte Agentenklasse  $AC_i$  bezieht sich auf das zuvor eingeführte ATG Modell, das das Laufzeitverhalten und die von Agenten ausgeführten Aktion definiert.

### Wahrnehmung

Ein Agent interagiert mit seiner Umgebung, indem er eine Datenübertragung unter Verwendung eines einheitlichen Tupelraums mit einer koordinierten datenbankähnlichen Schnittstelle durchführt.

Daten aus der Umgebung beeinflussen das folgende Verhalten und die Aktion eines Agenten. Daten, die an die Umgebung (z.B. die Datenbank) weitergegeben werden, beeinflussen das Verhalten anderer Agenten.

### Speicher

Zustandsbasierte Agenten führen Berechnungen durch, indem sie Daten ändern. Da Agenten als autonome Datenverarbeitungseinheiten betrachtet werden können, werden sie hauptsächlich private Daten modifizieren, und ein Berechnungsergebnis, das diese Daten verwendet, in die Umgebung übertragen.

Daher enthält jeder Agent und jede Agentenklasse eine Menge von Körpervariablen  $\mathbb{V} = \{v_1: typ_1, v_2: typ_2, ..\}$ , die durch Aktionen in Aktivitäten modifiziert und in Aktivitäten und Übergangsausdrücken gelesen werden.

### Parameter

Agenten können zur Laufzeit von einer bestimmten Agentenklasse instantiiert werden, die Agenten mit gleichen anfänglichen Steuerungs- und Datenzuständen erstellt.

Um einzelne Agenten zu unterscheiden (Individuen zu erzeugen), wird eine externe sichtbare Parametermenge  $\&Popf; = \{p_1: typ_1, p_2: typ_2, ..\}$  hinzugefügt, die die Erstellung verschiedener Agenten bezüglich des Datenzustands ermöglicht. Innerhalb einer Agentenklasse werden Parameter wie Variablen behandelt.

Eine Agentenklasse  $AC_i$  ist daher zunächst definiert durch das folgende Mengentupel:

$$\begin{aligned} AC_i &= \langle \mathbb{A}_i, \mathbb{T}_i, \mathbb{V}_i, \mathbb{P}_i \rangle \\ \mathbb{A} &= \{a_1, a_2, \dots, a_n\} \\ \mathbb{T} &= \{t_{ij} = t_{ij}(a_i, a_j, cond) \mid a_i \xrightarrow{cond} a_j; i, j \in \{1, 2, \dots, n\}\} \\ a_i &= \{i_1, i_2, \dots \mid i_u \in ST\} \\ \mathbb{V} &= \{v_1, v_2, \dots, v_m\} \\ \mathbb{P} &= \{p_1, p_2, \dots, p_i\} \end{aligned}$$

## Multiagentensysteme

### Definition 1.

Es gibt ein Multiagentensystem (MAS), das aus einer Reihe einzelner Agenten besteht ( $ag_1, ag_2, ..$ ). Es gibt verschiedene Verhaltensweisen für Agenten, die als Klassen  $\mathbf{AC} = \{AC_1, AC_2, ..\}$  bezeichnet werden. Ein Agent gehört zu einer dieser Klassen.

Jede Agentenklasse wird dann durch das erweiterte Tupel  $AC = \langle \mathbb{A}, \mathbb{T}, \mathbb{F}, \mathbb{S}, \mathbb{H}, \mathbb{V}, \&\text{Popf}; \rangle$  angegeben.

- $\mathbb{A}$  ist der Satz von Aktivitäten (Graphenknoten),  $\mathbb{T}$  ist der Satz von Übergängen, die Aktivitäten (Beziehungen, Graphenkanten) verbinden,
- $\mathbb{F}$  ist der Satz von Rechenfunktionen,
- $\mathbb{S}$  ist der Satz von Signalen,  $\mathbb{H}$  ist der Satz von Signalhandlern,
- $\mathbb{V}$  ist die Menge der Körpervariablen und  $\&\text{Popf};$  die Menge der Parameter, die von der Agentenklasse verwendet werden.

**Definition 2.**

In einer spezifischen Situation ist ein Agent  $ag_i$  an einen Netzwerkknoten  $N_{m,n,o,\dots}$  (z.B. Mikrochip, Computer, virtueller Simulationsknoten) an einem eindeutigen räumlichen Ort (m, n, o, ..) gebunden und wird dort verarbeitet.

Es gibt eine Menge verschiedener Knoten  $\mathbf{N} = \{N_1, N_2, \dots\}$ , die z.B. in einem maschenartigen Netzwerk mit einer Nachbarverbindung (z.B. vier in einem zweidimensionalen Gitter) angeordnet sind. Die Knotenverbindung kann dynamisch sein und sich im Laufe der Zeit ändern. Die Knotennachbarn sind unterscheidbar.

Jeder Knoten ist in der Lage, eine Anzahl von Agenten  $n_i(AC_i)$  zu verarbeiten, die zu einer Agentenverhaltensklasse  $AC_i$  gehören, und mindestens eine Teilmenge von  $\mathbf{AC}' \subset \mathbf{AC}$  zu unterstützen.

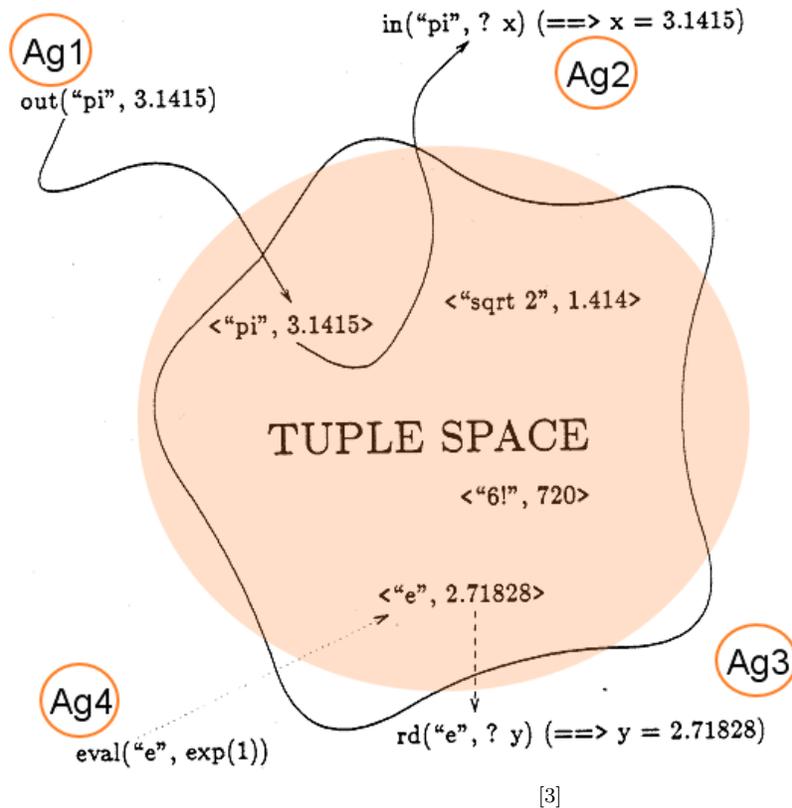
Ein Agent (oder zumindest sein Zustand) kann zu einem Nachbarknoten migrieren wo er weiter ausgeführt wird.

## 12.7. Tupelräume

- Tupel-Räume stellen ein **assoziertes Shared-Memory-Modell** dar, wobei die gemeinsam genutzten Daten als **Objekte** mit einer Reihe von **Operationen** betrachtet werden, die den Zugriff der Datenobjekte unterstützen
- Tupel sind in **Räumen** organisiert, die als abstrakte Berechnungsumgebungen betrachtet werden können.
- Ein Tupelraum verbindet verschiedene Programme, die **verteilt** werden können, wenn der Tupel-Space oder zumindest sein operativer Zugriff verteilt ist.

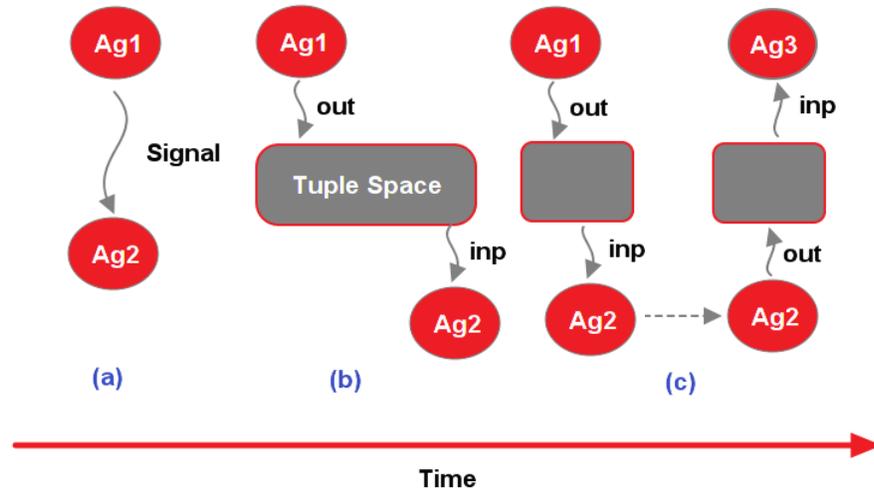
□ Oder: **Mobile Agenten** als Tupel Verteiler!

- Das Tupelraum Organisations- und Zugangsmodell bietet **generative Kommunikation**, d.h. Datenobjekte können in einem Raum durch Prozesse mit einer Lebensdauer über das Ende des Erzeugungsprozesses hinaus gespeichert werden.
- Ein bekanntes Tupelraum-Organisations- und Koordinationsparadigma ist **Linda** [GEL85].



**Figure 51.** Ein Schnappschuss eines Tupelraumes mit Tupeln und Tupeloperationen

- Kommunikation von Agenten über Tupelräume ist eine **Koordinations-sprache**.



**Figure 52.** Direkter Nachrichtenaustausch (a), z.B. durch Signale, im Vergleich zu generativer Kommunikation (b) und virtuelle verteilte Räume (c) durch mobile Prozesse (Agenten)

## 12.8. Tupelräume - Datenmodell

- Die Daten sind mit Tupeln organisiert.
- Ein Tupel ist eine lose gekoppelte Verbindung einer beliebigen Anzahl von Werten beliebiger Art /Typ/
- Ein Tupel ist ein Wert und sobald es in einem Tupelraum gespeichert ist, ist es persistent.
- Tupeltypen ähneln den Datenstrukturtypen, sie sind jedoch dynamisch und können zur Laufzeit ohne statische Beschränkungen erstellt werden.
- Auf die *Elemente von Tupeln* kann nicht direkt zugegriffen werden, was üblicherweise Mustererkennung und *musterbasierte Dekomposition* erfordert, im Gegensatz zu Datenstrukturtypen, die einen benannten Zugriff auf Feldelemente bieten, obwohl die Behandlung von Tupeln als Arrays oder Listen diese Beschränkung lösen kann.
- Ein Tupel mit  $n$  Feldern heißt  $n$ -stellig und wird in der Notation  $\langle v_1, v_2, \dots \rangle$  angegeben.

## 12.9. Tupelräume - Operationale Semantik

- Es gibt eine Reihe von Operationen, die von Prozessen angewendet werden können, bestehend aus
  - einer Reihe reiner Datenzugriffsoperationen, die Tupel als passive Datenobjekte behandeln,
  - und Operationen, die Tupel als eine Art von aktiven Rechenobjekten behandeln (genauer gesagt, zu berechnende Daten).
  - RPC-Semantik (Remote Procedure Call).

### out

`function (t:tuple)`

Die Ausführung der Ausgabeoperation fügt das Tupel  $t$  in den Tupelraum ein. Mehrere Kopien desselben Tupelwerts können eingefügt werden, indem die Ausgabeoperation iterativ angewendet wird. Die gleichen Tupel können nach dem Einfügen in den Tupelraum nicht unterschieden werden.

### inp

`function (p:pattern, callback: function (tuple|tuple[]|null),all?:boolean)`

Die Ausführung der Eingabeoperation entfernt ein Tupel  $t$  aus dem Tupelraum, der der Mustervorlage  $p$  entspricht. Wenn kein passendes Tupel gefunden wird führt das zu einer Blockierung des aufrufenden Prozesses bis ein passendes Tupel eingestellt wird.

### rd

`function (p:pattern, cb:function (tuple|tuple[]|null),all?:boolean)`

Die Ausführung der Leseoperation gibt eine Kopie eines Tupels  $t$  zurück, dass der Vorlage  $p$  entspricht, entfernt sie jedoch nicht. Wenn kein passendes Tupel gefunden wird führt das zu einer Blockierung des aufrufenden Prozesses bis ein passendes Tupel eingestellt wird.

### Beispiele

```
out(["Sensor",1,100]);
out(["Sensor",2,121]);
inp(["Sensor",1,_], function (t) { if (t) v=t[2]; });
inp(["Sensor",_,_], function (t) { if (t) n=t[1],v=t[2] });`
rd(["Sensor",_,_], function (t) { if (t) n=t[1],v=t[2] });`
```

`try_inp, try_rd`

```
function (0,p:pattern,callback?:function,all?:boolean) → tuple|tuple[]|null
```

Nichtblockierende Version von *inp/rd*. Wird kein passendes Tupel gefunden wird die Operation ergebnislos terminiert.

**try\_inp, try\_rd**

```
function (tmo:number,p:pattern,callback:function,all?:boolean) → tuple|tuple[]|null
```

Teilblockierende Version von *try\_inp/try\_rd*, Wird innerhalb einer Zeit von *tmo* kein passendes Tupel gefunden wird die Operation ergebnislos abgebrochen.

- Die Verwendung von zeitlich unbegrenzt blockierenden Operationen kann unter Betrachtung der Lebendigkeit von Agenten nachteilig sein. Daher sollte immer eine zeitliche Begrenzung und anschließende Abfrage des Operationsstatus erfolgen (abgebrochen?)

**test**

```
function (pattern) → boolean
```

Nicht blockierender Test eines Tupels.

**ts (testandset)**

```
function (pattern,function (tuple) → tuple) → tuple
```

Nicht blockierender Test eines Tupels und atomare Veränderung eines Tupels, dass der Vorlage *p* entspricht. Das zweite Argument ist eine Abbildungsfunktion. Das Ergebnistupel ersetzt das ursprüngliche.

**rm**

```
function (p:pattern,all?:boolean)
```

Entfernung eines oder aller passenden Tupel.

## **Markierungen**

- Tupel sind persistent und können für immer in einem Tupelraum verbleiben!
- Daher ist die Verwendung von *Markierungen* häufig sinnvoll.
- Eine Markierung ist ein Tupel mit einer Lebenszeit  $\tau$
- Nach Ablauf der Lebenszeit wird das Tupel - sofern es nicht entfernt wurde - durch einen Garbagecollector entfernt.

$$m = \langle \tau, \vec{d} \rangle, \text{ with } \vec{d} ::= d|d, \vec{d} \text{ and } d ::= v|\varepsilon|x, \tau : \text{timeout}$$

**mark**

```
function (tmo:number,t:tuple)
```

Ausgabe eines Tupels  $t$  mit einer Lebenszeit  $\tau$  (im Tupelraum).

**alt**

function (pattern [], function (tuple|tuple[]|null),all?:boolean)  
Gleichzeitige Mehrfachabfrage (mit *inp* Semantik) von einer Menge von Mustern. Das erste Tupel welches einem der Muster entspricht wird an die Rückruffunktion übergeben. Es gibt auch eine nicht- oder partiell blockierende Variante *try\_alt*.

## 12.10. Tupelräume - Produzenten und Konsumenten

**Produzent**

```
this.act = {
  percept : function () {
    out(["SENSORNODE"]);
    mark(1000,["SENSOR","CLOCK",time()])
    mark(1000,["SENSOR","GPS",
              {lati:x,long:y}])
  }
}
```

**Konsument**

```
this.act = {
  process : function () {
    if (test(["SENSORNODE"]))
      inp(["SENSOR","CLOCK",_], function (t) {
        if (t) log('It is time '+t[2]);
      })
  }
}
```

**Aufgabe**

Betrachte das zeitliche Verhalten im obigen Beispiel (Laufzeiteigenschaften)

1. Was könnte im Produzenten-Konsumenten System möglicherweise falsch laufen?

2. Wie müsste der Agentencode geändert werden um Laufzeitfehler zu vermeiden?

---

## 12.11. Signale

- *Tuplerraumkommunikation* ist
  - ❑ Anonym und
  - ❑ Generativ (d.h. die “Nachricht” kann eine längere Lebensdauer als der Absender besitzen)
- Eignet sich daher für die Kommunikation von lose gekoppelten Ensembles
- Neben diesen lose gekoppelten Gruppen gibt es durch Replikation von Kindagenten Gruppen mit starker Kopplung und Kenntnis der Identitäten
  - ❑ Elternagent erfährt die Identität eines Kindagenten beim Forking
  - ❑ Kindagent kann die Identität des Elternagenten durch die Operation `myParent()` erfahren.
- **Signale sind zielgerichtete Nachrichten mit identifizierbaren Absender und Empfänger** (wobei es bei Multi- und Broadcastnachrichten viele geben kann)
- Ein Signal `signal(arg)` besteht aus einem Signalnamen oder einer Signalnummer mit einem optionalen Argument.
- Signale müssen auf Empfängerseite mit einem Signalhandler im `this.on` Abschnitt des Agentenmodells eingerichtet werden.

### `send`

```
function (target:id,signal:string|number,arg?:*)
```

Sendet eine Signalnachricht `signal(arg)` an den Empfängeragenten `target`.

### `broadcast`

```
function (ac:string,r:number,signal:string|number,arg?:*)
```

Sendet eine Signalnachricht `signal(arg)` an alle Empfängeragenten der Klasse `ac` im Radius `r`.

### `this.on[“signal”]`

```
function (arg,from)
```

Ein Signalhandler für das Signal `signal`.

### **Produzent**

```
this.act = {  
  a1 : function () {  
    this.child = fork()  
  },  
  a2 : function () {  
    send(this.child, 'ALARM', 100)  
  }  
}
```

### **Konsument**

```
this.on = {  
  ALARM : function (arg, from) {  
    log('Got ALARM '+arg+' from '+from)  
  }  
}
```

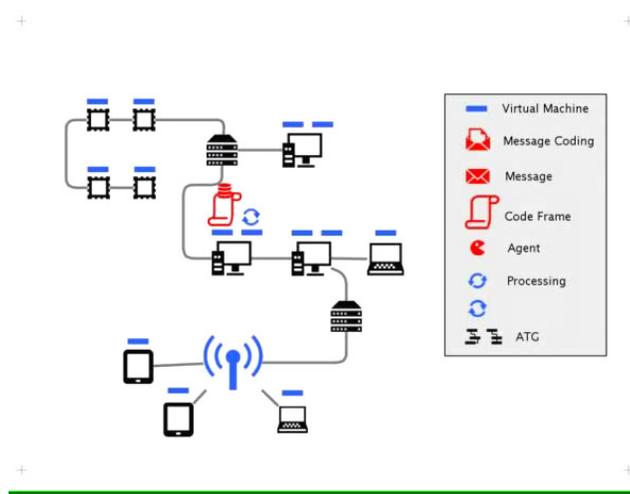
## **12.12. Mobile Agenten**

- Ein Agent in Ausführung wird durch einen Prozess repräsentiert.
- Der Prozesszustand setzt sich zusammen aus:
  - ❑ Körpervariablen (Datenzustand)
  - ❑ Nächste Aktivität und Mikroschedulingblöcke (Kontrollzustand)
  - ❑ Signale, Nachrichten
  - ❑ Pfad im Netzwerk (Plattformen auf denene der Agent bereits ausgeführt wurde)
  - ❑ Verhalten (D.h. ATG)!
- **Mobile Agenten bedeuten mobile Prozesse!**
- Bei der Migration muss ein Agentenprozess in einen
  1. **Prozessschnappschuss** (Container) eingefroren und serialisiert werden,

2. Der Schnappschuss in textueller Form (JSON+) von einer Plattform zu einer anderen übertragen werden, und
3. Der Prozessschnappschuss wieder deserialisiert werden, und der Prozess wird weiter ausgeführt.

### **Mobiler Code**

- Anstelle von mobilen Daten (Kommunikation) wird durch Agenten mobiler Code in Netzwerken transportiert



- Eine Agentenplattform repräsentiert einen physikalischen Knoten in einem Virtuellen Netzwerkgraphen
  - ❑ Es gibt virtuelle und physikalische Kommunikationsverbindungen zwischen den Netzwerkknoten
  - ❑ Beispiel: Internet Protokoll (IP/UDP/HTTP)
- *Ports* von Plattformen verbinden und stellen *Links* her
- Ein Agent kann auf einen anderen Knoten (Plattform) migrieren indem er die *moveto* Operation ausführt (innerhalb einer Aktivität):
  - ❑ Dabei muss der Agent als Argument das Ziel angeben
  - ❑ Ziel kann ein Knotenname (DIR.NODE(<nodename>)), eine IP Adresse (DIR.IP(<url>)), oder bei gerichteten Verbindungen die Richtung sein (z.B. DIR.NORTH)

- Alle aktuell mit einer Plattform verbundenen anderen Plattformen (Namen) können mit der *link* Funktion erfragt werden:

```
{  
  var connected=link(DIR.IP('%'));  
  if (connected.length) {  
    var next = random(connected);  
    moveto(DIR.NODE(next));  
  }  
}
```

## 13. Agentenplattformen

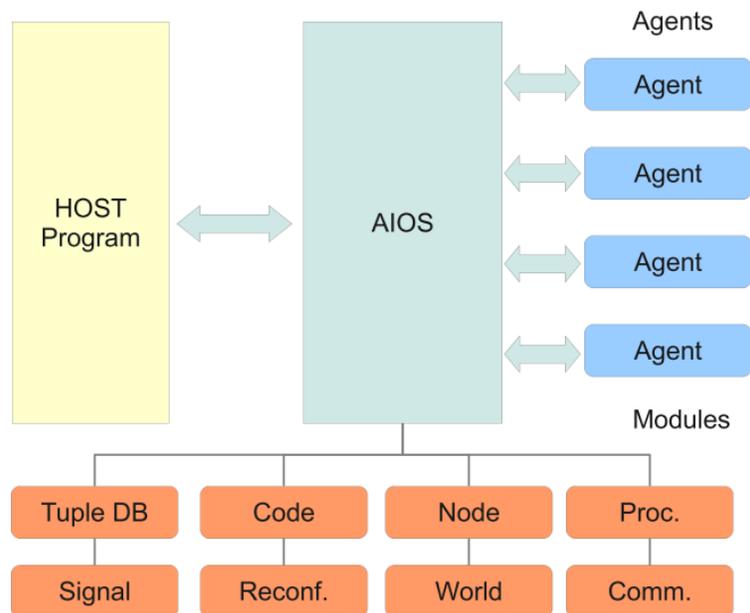
### 13.1. JavaScript Agent Machine

- Vollständig in JavaScript programmiert
- Portabel:
  - ❑ Web Browser
  - ❑ Cordova WebKit App
  - ❑ CLI mit nodejs/jxcore/jerryscript
  - ❑ Eingebettet in Java Programm
- Schlank: ca 1M Byte JS Code
- AgentJS Code kann direkt ausgeführt werden (nur Sandboxing erforderlich)!
- Eine physische JAM kann eine Vielzahl von virtuellen JAM Knoten ausführen (nur sequenzielles Scheduling)

### **AIOS**

- Das AIOS kapselt eine Vielzahl von Modulen und stellt eine API für Agenten zur Verfügung:
  - ❑ TUPLE: Tupleraum Datenbank
  - ❑ SIGNAL: Signalpropagation zwischen Agenten und JAM Knoten

- ❑ CODE: AgentJS Text  $\Leftrightarrow$  Code Transformation, Code Morphing, und Sandboxing
- ❑ NODE: Virtueller JAM Knoten
- ❑ WORLD: Bindung von virtuellen JAM Knoten in einem physischen Knoten (Virtualisierung)
- ❑ PROC: Agentenprozesses (Ausführungscontainer für Agenten)
- ❑ MOBI: Agentenmobilität, COMM: JAM Kommunikation
- ❑ SECU: JAM Capabilities und Sicherheit
- ❑ WATCHDOG: Faires Agentenscheduling durch Laufzeitüberwachung (Time slicing)
- ❑ ML: Machine Learning  $\rightarrow$  Fokus mobile Algorithmen und Modelle
- ❑ SAT: SAT Logic Solver  $\rightarrow$  Knowledge Base



**Figure 53.** Das Agent Input-Output System als Schnittstelle zwischen Agenten und JAM und einer Hostplattform (bzw. Applikation)

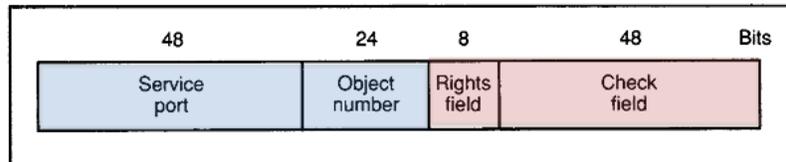
### **Agentenrollen**

- In der realen Welt ist die Anwendungssicherheit ein wichtiges Schlüsselmerkmal einer verteilten Agentenplattform.
- Die Ausführung von Agenten und der Zugriff auf Ressourcen müssen kontrolliert werden, um Denial-of-Service-Angriffe, Agent-Masquerading, Spionage oder anderen Missbrauch zu verhindern.
- Daher haben Agenten unterschiedliche **Zugriffsebenen (Rollen)**:
  0. Gast (nicht vertrauenswürdig, semi-mobil)
  1. Normal (vielleicht vertrauenswürdig, mobil)
  2. Privilegiert (vertrauenswürdig, mobil)
  3. System (sehr vertrauenswürdig, System relevant, lokal, nicht mobil)
- Die unterste Ebene (0) ermöglicht keine Agentenreplikation, Migration oder das Erstellen neuer Agenten.
- Die JAM-Plattform entscheidet über die Sicherheitsstufe für neue empfangene Agenten. Ein Agent kann keine Agenten mit einer höheren Sicherheitsstufe als die eigene erstellen.
- Die höchste Stufe (3) hat einen erweiterten AIOS mit Host-Plattform-Gerätezugriffsfähigkeiten.
- Agenten können Ressourcen (z.B. CPU-Zeit) aushandeln und ein Level-Raise erreichen, das mit einem Schlüssel gesichert ist, der die erlaubten Upgrades definiert.
  - ❑ Die Systemebene (3) kann nicht ausgehandelt werden.
- Der Schlüssel ist knotenspezifisch. Eine Gruppe von Knoten kann sich einen gemeinsamen Schlüssel teilen (durch einen Server-Port identifiziert der den JAM Knoten identifiziert).
- Ein Schlüssel besteht aus einem Server-Port, einem Rechtefeld und einem verschlüsselten Schutzfeld das das Rechtefeld enthält, das mit einem zufälligen Port generiert wird, der nur dem Server (Knoten) bekannt ist.

### **Schlüssel (Capabilities)**

- Ein Schlüssel kann verwendet werden um:
  - ❑ Einen Agenten von einer Plattform  $A$  nach  $B$  zu migrieren ( $B$  verlangt den Schlüssel um den Agenten auszuführen);

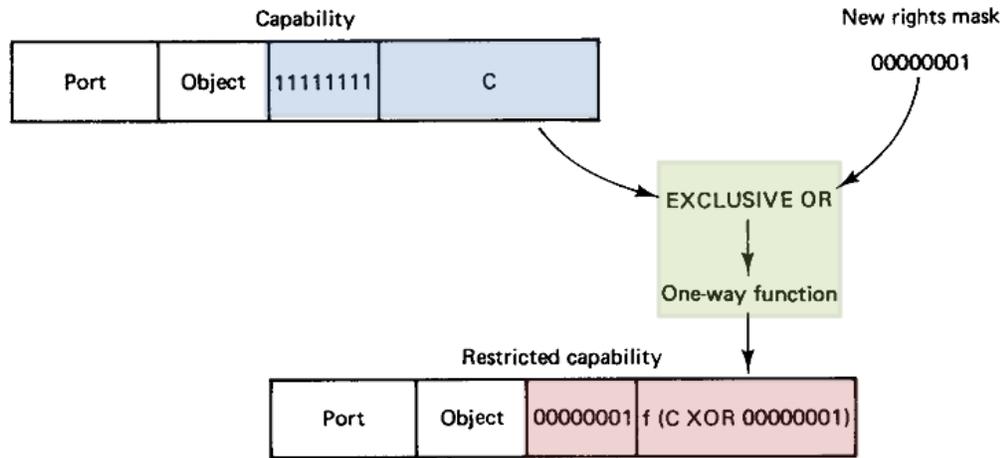
- Eine neue Agentenrolle (Stufe) auszuhandeln;
- In einer Agentenrolle neue Ressourcen auszuhandeln (CPU, MEM, TS, SCHED, ..);
- Um neue Agenten erzeugen und andere terminieren zu können



**Figure 54.** Aufbau einer Capability: Der Serverport bindet die Capability an einen Service, die Objektnummer ist optional und kann eine Unterklasse des Service oder der zu schützenden Ressource darstellen, das Rechtfeld kodiert die möglichen Operationen der Serviceklasse, und das Schutzfeld (enthält Rechtfeld nochmals verschlüsselt) schützt die Capability gegen Manipulation.

### **Schutz von Capabilities**

- Das Rechtfeld ist zentral da es die durch die Capability erlaubten Operationen angibt.
  - CPU Zeit erhöhen
  - Änderung des Privilegienlevels
  - Migration usw.
  - Zugriff auf Sensoren und persönliche Nutzerdaten
- Damit das Rechtfeld nicht manipuliert werden kann ohne dass die Capability ungültig wird (und ggf. die Objektnummer) wird ein Schutzfeld erstellt welches die Rechte mit einem privaten Schlüssel (Port) mittels einer One-way Funktion verschlüsselt.
  - Nur der Service / Agentenplattform kennt den privaten Schlüssel

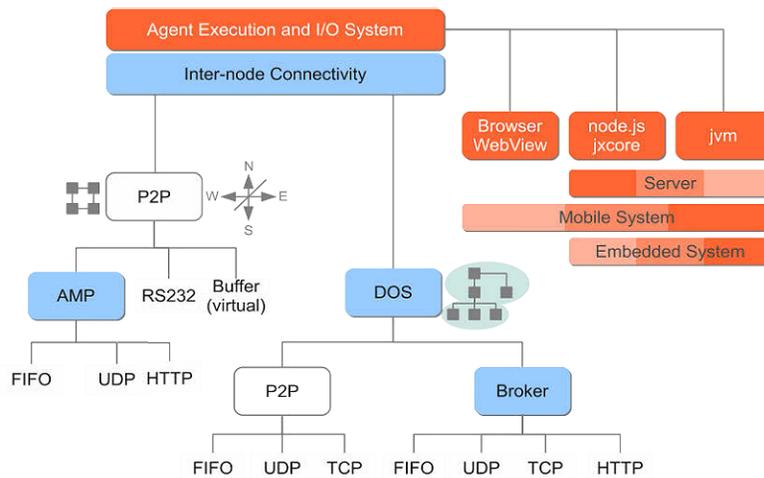


**Figure 55.** Erzeugung einer öffentlichen restriktiven Capability aus einer privaten nicht eingeschränkten (enthält privaten Schlüssel  $C$ ) mittels One-way Verschlüsselungsfunktion  $f(C \text{ xor } R)$

### Kommunikation und Netzwerke

- ▶ JAM Plattformen können in beliebigen Netzwerken miteinander verbunden werden.
- ▶ Eine Vielzahl von Kommunikationsprotokolle sind verwendbar:
  - ❑ RS232
  - ❑ UDP
  - ❑ TCP
  - ❑ HTTP
- ▶ Beliebige Netzwerktopologien können gebildet werden (physisch wie logisch):
  - ❑ Stern
  - ❑ Gitter (1D/2D/3D)
  - ❑ Bus
  - ❑ Intranet und Internet (allg. Graphen)

- AMP: Agent Management Port als gemeinsames Protokoll und Interface in heterogenen Systemen
- AMP definiert eine Menge von Nachrichten die dem Transport von
  - ❑ Agenten,
  - ❑ Signalen und Tupeln,
  - ❑ und Handshakes dienen.



**Figure 56.** JAM Konnektivität und eine breites Spektrum an unterstützten JavaScript Host Plattformen

### Netzwerkstrukturen

- JAM Plattformen können über eine oder mehrere virtuelle Kanäle (AMP) miteinander verbunden werden
  - ❑ Verwendung von Relaisstationen führt zu sternartigen Kommunikationsnetzwerken
  - ❑ Browser und mobile JAM Knoten können nur an im Internet sichtbare JAM Knoten über HTTP sich verbinden → erfordert Relais Knoten (ohne Nutzerinteraktion, kopfflos)

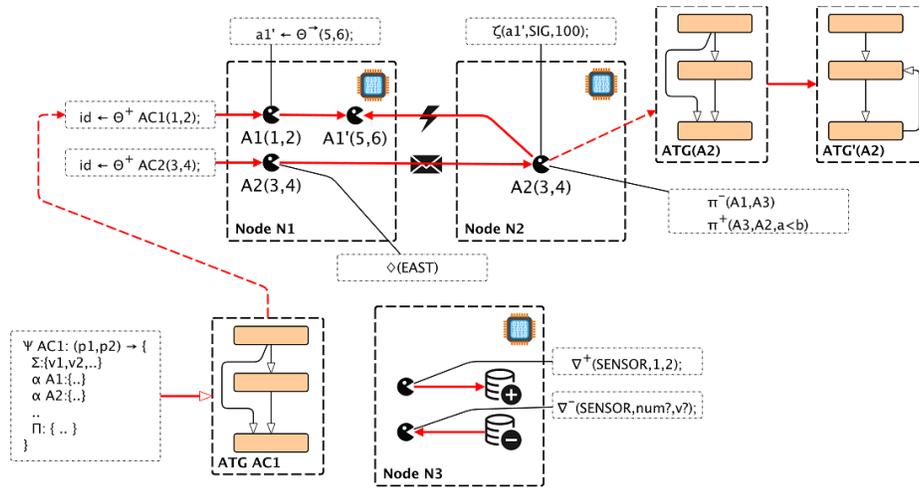


Figure 57. Effekt von verschiedenen AAPL/AgentJS Anweisungen zur Laufzeit auf Agenten und Plattformen

## 14. Programmierung

### 14.1. JavaScript :: Daten und Variablen

- Variablen werden mit dem Schlüsselwort `var` definiert → Erzeugung eines Datencontainers!
- Es gibt keine Typdeklaration in JS! Kerntypen:
 
$$\mathbf{T}_{\text{core}} = \{\text{number, boolean, object, array, string, function}\}$$
- Alle Variablen sind **polymorph** und können alle Werttypen aufnehmen.
- Bei der Variabledefinition kann ein Ausdruckwert zugewiesen werden

```
var v = ε, ...; v = ε;
```

### 14.2. JavaScript :: Funktionen

- Funktionen können mit einem Namen oder anonym definiert werden
- Funktionen sind Werte 1. Ordnung → Funktionen können Variablen oder Funktionsargumenten zugewiesen werden

- Eine Funktion kann einen Wert mit der **return** Anweisung zurückgeben. Ohne explizite Wertrückgabe → **undefined**
- Es wird nur Call-by-value Aufruf unterstützt - jedoch werden Objekte, Funktionen und Arrays als Referenz übergeben; Parameter  $p_i$  sind an Funktionsblock gebunden

```
function name (p1,p2,...) { statements ; return  $\epsilon$  } name( $\epsilon_1,\epsilon_2,\dots$ )
```

- Da in JavaScript Funktionen Werte erster Ordnung sind können
  - ❑ Funktionen an Funktionen übergeben werden und
  - ❑ Funktionen neue Funktionen zurückgeben (als Ergebnis mit **return**)
- Es können daher **anonyme** Funktionen **function (..) {..}** definiert werden die entweder einer Variablen als Wert oder als Funktionsargument übergeben werden.

```
var x = function (pi) {  $\epsilon(p_i)$  }  
array.forEach(function(elem,index) {  $\epsilon(p_i)$  }
```

### 14.3. JavaScript :: Datenstrukturen

*In JavaScript sind Objekte universelle Datenstrukturen (sowohl Datenstrukturen als auch Objekte) die mit Hashtabellen implementiert werden. Arrays werden in JavaScript ebenfalls als Hashtabelle implementiert!. D.h. Objekte == Datenstrukturen == Arrays == Hashtabellen.*

- Es gibt *kein* nutzerdefinierbares Typensystem in JavaScript.
- Eine Datenstruktur kann jederzeit definiert und verändert werden (d.h. Attribute hinzugefügt werden)

```
var dataobject = {  
  a: $\epsilon$ ,  
  b: $\epsilon$ , ..  
  f:function () { .. }  
}  
..  
dataobject.c =  $\epsilon$ 
```

- Dadurch dass Objekte und Arrays mit Hashtabellen implementiert (d.h. Elemente werden durch eine Textzeichenkette referenziert) werden gibt es verschiedene Möglichkeiten auf Datenstrukturen und Objektattribute zuzugreifen:

```
dataobject.attribute  
dataobject["attribute"]  
array[index]  
array["attribute"]
```

#### 14.4. JavaScript :: Objekte

- Objekte zeichnen sich in der objektorientierten Programmierung durch Methoden aus mit der ein Zugriff auf die privaten Daten (Variablen) eines Objekts möglich wird.
- In JavaScript kann auf Variablen eines Objekts (die Attribute) immer direkt zugegriffen werden.
- Attribute können Funktionen sein - jedoch können die Funktionen nicht wie Methoden direkt auf die Daten des Objektes zugreifen.
- Daher definiert man Methoden über Prototypenerweiterung in JavaScript.
- Die Methoden können über die `this` Variable direkt auf das zugehörige Objekt zugreifen (also auch auf die Variablen/Attribute)
- Es gibt eine Konstruktionsfunktion für solche Objekte mit Prototypendefinition der Methoden
- Objekte werden mit dem `new` Operator und der Konstruktionsfunktion erzeugt.

```
function constructor (pi) {  
    this.x=ε  
    ..  
}  
constructor.prototype.methodi = function (..) {  
    this.x=ε;  
    ..  
}  
...  
  
var obj = new constructor(..);
```

## 14.5. AgentJS

- ▶ AgentJS ist die JavaScript Implementierung von AAPL (Activity-based Agent Programming Language)
  - ❑ Die meisten AAPL Operationen und Anweisungen sind in AgentJS verfügbar
  - ❑ Aber: JavaScript unterstützt nicht das Konzept der Prozessblockierung
  - ❑ Daher anderes Scheduling und Ablaufmodell mit Scheduling Blocks

### *Agentenklasse*

- ▶ **Agentenklassen** werden in JS über **Konstruktorfunktionen** definiert.
- ▶ Eine Agentenklasse definiert:
  - ❑ Körpervariablen (nur mobile und werterhaltende)
  - ❑ Aktivitäten (als Funktionsobjekt)
  - ❑ Übergangsbedingungen (als Funktionsobjekt)
  - ❑ Optionale Signalhandler (als Funktionsobjekt)
  - ❑ Ein `next` Attribute initialisiert mit der Startaktivität
- ▶ Aber: Das `this` Objekt als Referenz auf eine Agenteninstanz ist auch in geschachtelten Funktionen gültig, d.h., in allen
  - ❑ Aktivitätsfunktionen,
  - ❑ Übergangsfunktionen,
  - ❑ Signalhandlerfunktionen, und in
  - ❑ Callback Funktionen erster Ordnung von eingebauten Funktionen (z.B. `iter(list,function () { this is agent! }`)
- ▶ Ein Agentenprozess wird in einem Sandkasten gekapselt ausgeführt. Daher:
  - ❑ Ein Agent darf **nur** auf Körpervariablen zugreifen und keine freien Variablen oder lokale Variablen verwenden (d.h. welche die außerhalb des Konstruktors definiert wurden oder welche die innerhalb des Konstruktorkörper definiert wurden):

```
function ac (p) {  
  var x;      // Wrong!!!  
  this.y = p; // Correct!!!  
  this.act = { ax: function () { var a; // is correct  
  }  
}
```

**Definition 3.** (*Konstruktorfunktion: Template einer Agentenklasse*)

```
function ac(p1,p2,..) {  
  // Body Variables  
  this.x=ε; ..  
  this.act = { // Activities  
    a1: function () { .. },  
    a2: function () { .. },  
    ..  
    an: function () { .. }  
  }  
  this.trans = { // Transitions  
    a1: function () { return cond?ai:aj },  
    a2: aj,  
    ..  
  }  
  this.on = { // Signal Handler  
    error: function (err) { .. },  
    signal1: function (arg) { .. }  
  }  
  this.next = a1;  
}
```

### **Agenteninstantiierung und Terminierung**

```
function create(class:string,arguments:{},level?:number) → id:string  
function fork(arguments?:{},level?:number) → id:string  
function kill(id:string|undefined)
```

- Bei der *create* Operation die einen neuen Agenten der Klasse *ac* instantiiert werden die Klassenparameter der Instanz mit konkreten Werten mit dem Parameterargument `arguments:{p1:v1,p2:v2,..}` initialisiert.
- Bei der *fork* Operation die eine Kopie des aufrufenden Agenten erzeugt

werden Klassenattribute (die `this.x` Variablen, nicht die Parameter `{p}!`) mit neuen Werten überschrieben, d.h. `arguments:{x:v1,y:v2,..}`

- Die `kill` Operation ohne Argument führt zur Terminierung des aufrufenden Agenten (`== kill(me())`)

### **Beispiele**

```
id = create('explorer',{dir:DIR.NORTH,radius:1});
child = fork({x:10,y:20});
kill(child);
kill(me());
```

### **Asynchrone Operationen**

- Da JavaScript Funktionen als Werte erster Ordnung behandelt werden häufig sog. Rückruffunktionen (callback) an Funktionen als Argumente übergeben, die von der aufrufenden Funktion dann aufgerufen werden
- Rein synchrone Operationen (z.B. Tupelraumoperation `out` oder jegliche Berechnung, auch mit Rückruffunktionen wie `iter`) können in einer Sequenz innerhalb einer Agentenaktivität ausgeführt werden
- Es gibt aber auch asynchrone Operationen ((z.B. Tupelraumoperation `rd`), die zwar aufgerufen werden, aber erst später (ggfs. nach Durchlauf einer Aktivität) ausgeführt werden.

### **Prozessblockierung**

- Eine Aktivität wird in einem Durchlauf ausgeführt und kann nicht blockieren, wie dies z.B. bei den Tupeloperationen der Fall sein könnte.
  - ❑ Daher darf sich in *AgentJS* maximal nur **ein** blockierende Operation (die tatsächlich dann die Aktivität blockiert, und nicht den Programmfluss) **am Ende** einer Aktivität befinden.
  - ❑ Bei einer Sequenz von blockierenden Operationen muss in der Aktivität ein Scheduling Block verwendet werden (Mikroaktivitäten).
  - ❑ Jede Mikroaktivität darf eine blockierende Operation enthalten!
- Es gibt drei verschiedene Blockkonstruktoren:

- ❑ B: Ein sequenzieller Scheduling Block
- ❑ L: Ein iterativer Schleifenblock
- ❑ I: Ein Iteratorblock für Arrays und Objekte (Strukturen)

**Definition 4.** (*Mikroaktivitäten und Blöcke*)

```
B([
    function () { .. },
    function () { .. },
    function () { .. },
    ..
])
L([
    function init () {..},
    function cond () {..},
    function next () {..},
    [ function () { .. },
      function () { .. }, ..
    ])
I(obj:[]|{ },
  function next (elem:*) { },
  [
    function () {..},
    function () {..}, ..
  ],
  function finalize () { .. }
)
```

### **Tupeloperationen**

```
function alt (pattern [],callback:function,all?:boolean,tmo?:number)
function collect (to:path,pattern) → number
function copyto (to:path,pattern) → number
function evaluate (pattern,callback:function (tuple)) → tuple
function inp (pattern,callback:function(tuple|tuple[]|none),all?:boolean,tmo?:number)
function listen (pattern,callback:function (pattern) → tuple)
function out (tuple)
function mark (tuple,tmo:number)
function rd (pattern,callback:function(tuple|tuple[]|none) ,all?:boolean,tmo?:number)
function rm (pattern,all?:boolean)
function store (to:path,tuple) → number
function test (pattern) → boolean
function ts (pattern,callback:function(tuple) → tuple)
function alt.try inp.try rd.try (tmo:number,..)
```

➤ *collect*, *copyto*, und *store* sind verteilte Tupelraumoperationen und wirken

auf entfernten Tupelräumen

### **Beispiele**

```
out(['MARKING1',1]);
out(['SENSORA',100,true]);
inp(['SENSORA',_],function (tuple) {
  if (tuple) this.s =tuple[1];
});
rm(['SENSORA',_],true);
rd.try(0,['SENSORA',_],function (tuple) { .. });
// oder bei Timeout 0 alternativ ohne callback =>
var t = rd.try(0,['SENSORA',_]);
ts(['MARKING',_],function (t) { t[1]++ });
alt([
  ['SENSORA',_],
  ['SENORB',_],
  ['EVENT'],
],function (tuple) {
  if (tuple && tuple[0]=='EVENT') {...}
  else ..
});
```

### **Signale und Handler**

```
function send (to:aid,sig:string|number,arg?:*)
function broadcast (class:string,range,@sig,@arg?)
function sendto (to:dir,sig:string|number,arg?:*)
function sleep (milli:number)
function timer.add (milli:number,sig:string,arg:*,repeat:boolean) → string
function timer.delete (sig:string)
this.on = { SIGNAL : function (arg,from) { .. } }
```

### **Typen und Muster**

```
type aid = string
type range = hops:number|region:{dx:number,dy:number,..}
enum DIR = {NORTH , SOUTH , WEST , EAST , ..
            PATH (path:string) , NODE(id:string),
            IP (ip:string) ,
            CAP {cap:string}
} : dir
```

- Signalhandler werden außerhalb von Aktivitäten in der `this.on` Sektion ausgeführt. Der Objektattributname ist der Signalname.

### **Code Muster**

```
this.child=none;
this.act = {
  a1: function () {
    this.child=fork({child:none});
    timer.add(500,'QUERY',true);
  }
  a2: function () {
    // Raising of signal
    if (this.child) send(this.child,'PARENT',me());
  }
}
// Installation of Signal Handler
this.on : {
  PARENT : function (arg) {
    log('Got signal from my parent '+arg);
  },
  QUERY: function () { .. }
}
```

## Mobilität von Agenten

```
enum DIR = {NORTH , SOUTH , WEST , EAST , ..
            PATH (path:string) , NODE{node:string),
            IP (ip:string) ,
            CAP (cap:string)
} : dir
function moveto (to:dir)
function opposite (dir) → dir
function link (dir) → boolean|string|[]
```

- Für die IP Richtung existiert eine Typkonstruktionsfunktion `DIR.IP(ip:string|number)` mit der Angabe einer IP Adresse und einer IP Portnummer im Format "IP:PORT", oder bei Verbindungen von JAM Knoten auf den gleichem Hostrechner nur die IP Portnummer.
- Die *opposite* Funktion liefert die entgegengesetzte Richtung, z.B. *NORTH* → *SOUTH*. Bei IP Ports i.A. nicht definiert oder bei bereits migrierten Agenten die IP Adresse des letzten Knotens.
- Als Ziel kann auch ein JAM Knotenname angegeben werden (`DIR.NODE("name")`).

### moveto

```
function moveto (to:dir)
```

Der Agent wird eingefroren, zu der neuen Plattform gesendet, und dort weiter ausgeführt. Die Richtung kann einen geometrischen Bezug haben (z.B. `DIR.NORTH`), oder eine Plattformreferenz beschreiben (z.B. `DIR.NODE('hanaqude')`). Eine geometrische Richtung macht nur in Maschennetzwerken it P2P Verbindungen Sinn.

- Wenn keine Verbindung in die angegebene Richtung existiert wird der Agent terminiert!

### link

```
function link (dir) → boolean|string|[]
```

Liefert Informationen über aktuell erreichbare Nachbarknoten. Die *link* Funktion testet ob ein Port mit einer anderen Seite verbunden ist (nicht zuverlässig). Bei Multicast IP Ports (Standard) gibt die Funktion alle derzeit angebundenen anderen Knotenadressen (Routes) zurück (Aufruf mit Argument `DIR.IP('*')`).

- Die von der aktuellen Plattform (JAM Knoten) erreichbaren JAM Knotennamen können z.B. mit der `link(DIR.IP("%"))` Operation als Liste aus-

gegeben werden.

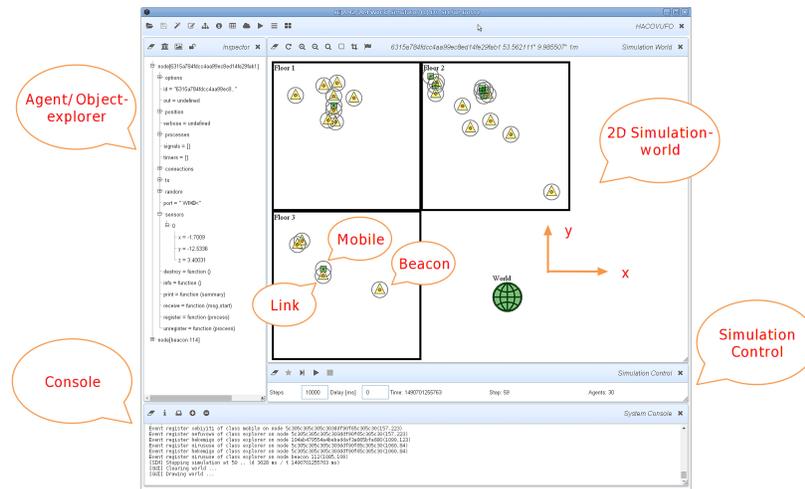
- Nach einer Migration wird die nächste folgende Aktivität (definiert durch die Übergangsbedingungen) ausgeführt. Daher muss die *moveto* Anweisung die letzte in einer Aktivität oder eingebettet in einen Scheduling Block sein!

## 14.6. Simulation

- Neben der “realen” Ausführung von mobilen Agenten in Computernetzwerken stellt die Simulation von verteilten Netzwerken aus Agentenplattformen und “virtuellen” mobilen Agenten ein wichtiges experimentelles Werkzeug dar um auch komplexe Szenarien raum- und zeitaufgelöst zu untersuchen.

### **SEJAM**

- SEJAM ist das *Simulation Environment for JAM* und baut auf einer “physischen” JAM Plattform auf
  - ❑ JAM wird um eine Bedieneroberfläche und Visualisierung ergänzt
  - ❑ Eine virtuelle Welt besteht aus einer Vielzahl virtueller (logische) JAM Knoten und einer geometrischen zweidimensionalen Welt
  - ❑ Agenten können in der virtuellen Welt zwischen virtuellen Knoten migrieren
- Die virtuellen Knoten sind in der geometrischen Welt mobil
- Die virtuellen Knoten verfügen über virtuelle Kommunikationsverbindungen (z.B. radial wie Mobilfunk)
- Die JAM Plattform in SEJAM kann mit realen Netzwerken gekoppelt werden!
- Mobile Agenten können nicht nur zwischen virtuellen Knoten migrieren, sondern auch die auf externe Plattformen in der realen Welt migrieren



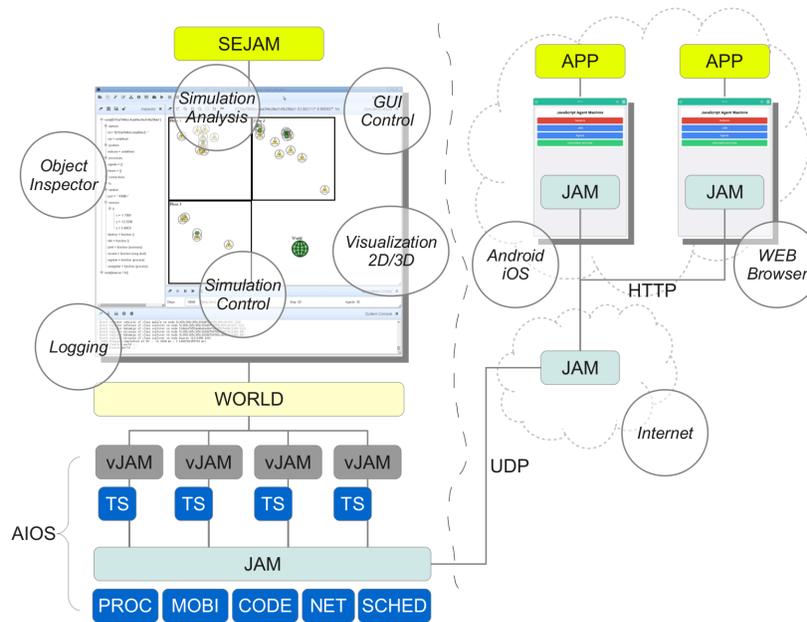
**Figure 58.** SEJAM2 mit zweidimensionaler Plattformwelt (stationäre und mobile Knoten) und Agenten; Kreise um Knoten symbolisieren Funkkommunikation

### Augmented Virtuality

- Die Fusion von virtuellen Simulationen mit realen Netzwerken ermöglicht die Kopplung von virtuellen Welten mit realen Welten und Interaktion mit Menschen!
- Chat bots können sowohl in virtuellen als auch realen (menschbezogenen) Welten agieren!
- Das ermöglicht das Studium komplexer sozio-technischer Systeme

**Human-in-the-loop simulation for Augmented Virtuality** besteht daher aus:

- Virtueller Simulationen und Framework SEJAM2 mit JAM Plattform und virtuellen JAM Netzwerken die mit dem Internet verbunden sind
- Mobilen Geräten mit JAM Plattform verbunden mit dem Internet oder via Baken (Bluetooth z.B.)
- Stationäre Geräte (Sensoren, IoT, und Baken) die über das Internet mit der virtuellen Welt verbunden sind



**Figure 59.** Augmented Virtuality: Human-in-the-loop Simulation mit mobilen Agenten

## 15. Demonstrator

### 15.1. Smart City: Self-Organizing Light Control

#### Goal

Simulation and investigation of Crowd Interaction with Smart Cities

- Self-organized control of ambient light conditions (e.g., in streets or buildings) using Crowd Sensing

#### Virtual World

- World consists of streets and buildings
  - ❑ Beacons placed in buildings and beside streets
  - ❑ Smart Light Devices illuminating streets and buildings

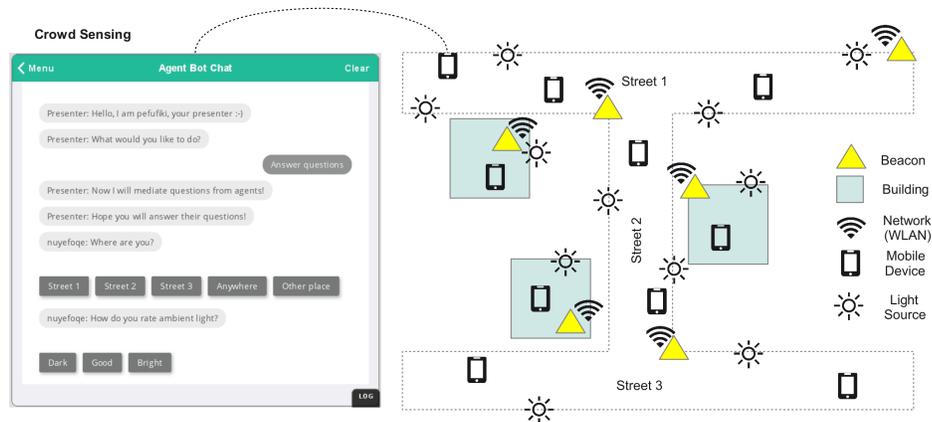
- **Agents** (can) represent:
  - ❑ Technical devices → Light Control
  - ❑ Network Infrastructure and Communication
  - ❑ Computational units and Smart Controllers
  - ❑ Chat Bots performing Crowd Sensing
  - ❑ Robots
  - ❑ Artificial Humans (Artificial Crowd)

### ***Real World***

- Mobile Devices using a WEB App with Chat Bot dialog
  - ❑ Users interact with remote agents via a chat dialog;
  - ❑ Remote agents investigate user location and an assessment of satisfaction of ambient light situation.
- Additionally, the device sensor data is collected (light, position, ..) if available

### ***Self-organizing Light Control***

- Beacons sent out mobile *explorer agents* performing question-answer dialogs on mobile devices (random walk)
- Based on collected *crowd data*:
  - ❑ The light conditions in buildings and streets should be adapted (darker or brighter);
  - ❑ Addressing 1. Crowd demands 2. Energy Saving.
- If action is required, mobile *notification agents* are sent out to neighbouring nodes to change light intensity based on:
  - ❑ Directed diffusion
  - ❑ Random walk
  - ❑ Divide-and-Conquer



**Figure 60.** Simulation world connected to the Internet performing Crowd Sensing → Real world is mapped on Virtual World!

## 15.2. Conclusions

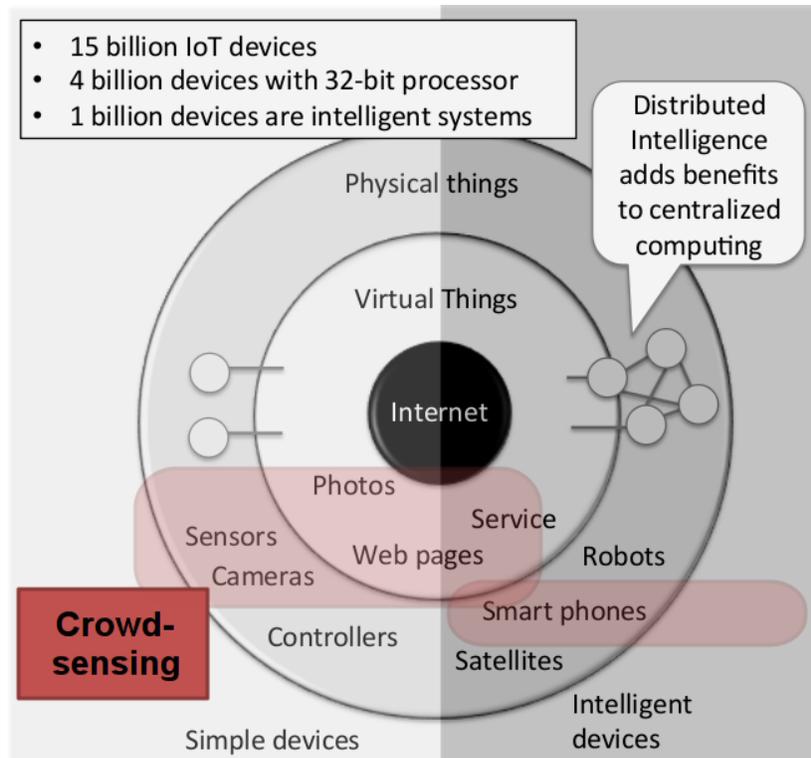
1. Fusion of real and virtual worlds can contribute to investigate complex socio-technical systems with ensemble sizes beyond Millions of entities (users, devices, machines)
2. Using agent-based modelling and simulation with mobile agents create an unified model and representation for:
  - Artificial humans
  - Chat Bots
  - Devices and Machines
  - Distributed Computation
3. Augmented Virtuality means the integration and tight coupling of ABM simulation with Crowd Sensing and technical devices interaction
4. Issues:
  - Time scales in real and virtual worlds
  - Spatial scales and spaces in real and virtual worlds
  - Mapping of real on virtual worlds

## 16. Sensoren

### 16.1. Das IoT und Crowdsensing

IoT: Internet-of-Things

- Das IoT wächst und bietet immer mehr sensorische Daten von Geräten die auch im Zusammenhang mit dem Crowdsensing und mobilen Geräten eingesetzt werden können
  - Fusion von heterogenen Daten: Geräte + Sensoren + Nutzer



**Figure 61.** Das IoT und Fusionsbereiche für das Crowdsensing [I]

## 16.2. IoT Sensorknoten

### Architektur

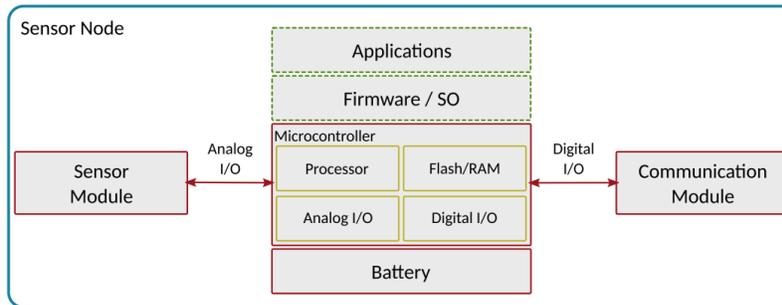


Figure 62. Allgemeine Architektur eines (nicht)mobilen Sensorknotens [4]

### Hardware

Board	CPU Speed	Memory/Storage	Power Comp.	Ports A/D	Size and Weight	USB Ports/Wireless Interfaces	Price	Operating Systems
Raspberry Pi Model A	700 MHz	256 MB/SD (4 GB)	0.8 W	0/30	6.5 × 5.5 cm/100 g	1 USB port/-	\$25	Raspbian
Raspberry Pi 2	900 MHz	1 GB/SD (4 GB)	1.5 W	0/30	8.4 × 5.5 cm/136 g	4 USB ports/-	\$35	Raspbian Ubuntu Mate Windows 10 IoT
Raspberry Pi 3	1.2 GHz	1 GB/SD (4 GB)	1.8 W	0/30	8.4 × 5.5 cm/136 g	4 USB ports/Wi-Fi and Bluetooth	\$40	Raspbian Ubuntu Mate Windows 10 IoT
Beagle Bone	720 MHz	512 MB/SD (4 GB)	1.0 W	14/6	8.4 × 5.5 cm/100 g	1 USB port/-	\$75	BeagleBone Linux
Intel Edison	500 MHz	1 GB/SD (4 GB)	0.6 W	14/6	5.9 × 2.8 cm/82 g	-/Wi-Fi and Bluetooth	\$90	Yocto Project
Pycorn	32 MHz	1 kB/32 kB	0.2 W	14/6	6.8 × 5.4 cm/100 g	Wi-Fi, Bluetooth and LoRa	\$45	MicroPython
Arduino Uno	32 MHz	1 kB/32 kB	0.2 W	14/6	6.8 × 5.4 cm/100 g	-/-	\$25	Processing-based
Arduino Nano	16 MHz	512 B/16 kB	0.2 W	14/6	4.2 × 1.8 cm/20 g	-/-	\$10	Processing-based
Smartphone	1GHz	1GB/16GB	5W	16	10 × 6cm/ 150g	Wi-Fi, Bluetooth, USB, NF, RFID Cellular/Mobile	\$100	Android ...

Figure 63. Vergleich verschiedener verfügbarer Hardwareplattformen für (nicht)mobile Sensorknoten [4]

## 16.3. Intelligent Large Scale Sensing

Neben Menschen liefern auch Geräte (allg. Sensoren) aus der Umgebung, technische Geräten, und mobile Geräte sensorische Daten die mit Methoden der KI fusioniert und aufbereitet werden können → Internet of Intelligent Things (IoIT) mit Crowdsensing!

### Personenzentrisches Erfassungssystem

Bereits eingeführt: Die Wahrnehmung von/durch Personen stellt eine Anwendungsdomäne dar, die den traditionellen Sensornetzbereich

**der Umwelt- und Infrastrukturüberwachung erweitert.**

- Die Menschen werden zu Trägern von Messgeräten und sowohl zu Produzenten als auch zu Konsumenten von Ereignissen.
- Um diese Netze von mobilen, auf Menschen ausgerichteten Erfassungsvorrichtungen zu nutzen, können aufgabenorientierte mobile Erfassungssysteme genutzt werden, die zu einer hohen Abdeckung fähig sind.
- Neue Art von Infrastruktur, bei der **mobile Sensoren das Potenzial haben, logisch zu mehr als einem Netzwerk zu gehören, während sie physisch mit ihren Netzbetreibern verbunden sind.**
- Durch die Nutzung dieser Systeme wird es möglich sein, enorme Datenmengen von einer großen Anzahl von Benutzern zu ermitteln und auszuführen.
- Ein personenzentriertes Erfassungssystem bringt die Individuen in eine symbiotische Beziehung mit sich selbst und technischen Geräten (Sensoren).
- Die traditionelle Sichtweise auf Sensornetzwerke verbindet sich mit der Ansicht, dass Personen, die Sensoren tragen, eine opportunistische Flächenabdeckung in die Realität übertragen.
- **Diese mobilen Sensoren können in räumliche und kontextuelle Bereiche gelangen, die mit statischen Sensoren nicht erreicht werden können, und sind besonders für Anwendungen hilfreich, die gelegentlich eine Erfassung erfordern.**
- Der Einsatz dieser Systeme kann **kontextabhängiges Rechnen** ermöglichen!

***Ubiquitäre Wahrnehmung***

- Verteilte Systeme wurden als Plattform verwendet, um die Interaktion zwischen Personengruppen und einer Reihe von Geräten zu ermöglichen.
- Mit dem fortschreitenden technologischen Fortschritt bei der Erfassung, Berechnung, Speicherung und Kommunikation werden allgegenwärtige Erfassungsgeräte Teil globaler verteilter Erfassungssysteme.
- Pervasive Computing wurde hauptsächlich zum Erstellen von Systemen verwendet, die eine kleine Anzahl von Geräten umfassen, die mit einzelnen Benutzern oder kleinen Gruppen interagieren, von tragbaren Computing-Plattformen bis hin zu allgegenwärtigen Computing-Infrastrukturen.
- Da die Technologie einfach in das tägliche Leben integrierbar ist, können kostengünstige Sensorsysteme auf der Grundlage verschiedener Sensorvor-

richtungen, z.B. Mobiltelefone, die von einer großen Anzahl von Personen getragen werden, aufgebaut und einfach eingesetzt werden.

- Tatsächlich werden mobile Geräte zunehmend als persönlicher Zugang zu einer großen Anzahl von Sensor- und Aktuator-Netzwerken verwendet, um Benutzer- und Community-Daten zu sammeln.
- Diese großen Datenmengen können entweder zentral (auf dedizierten Servern oder in der Cloud) verarbeitet oder über viele mobile und stationäre Computergeräte verteilt werden.

### **Standortbezogene Wahrnehmung**

- Mobile Erfassungsgeräte sind benutzerzentriert und daher geeignet, ortsabhängige Benutzeraktivitäten und -interaktionen zu erfassen.
- Für Anwendungen kann es erforderlich sein, dass ausgewählte Daten einem bestimmten Bereich und Zeitfenster zugeordnet werden. Der Standort kann über WLAN ermittelt werden oder auf der Triangulation der Signalstärkemessungen von zellularen Basisstationen basieren.
- Bei der ersten Methode ist jedoch eine **Internet-Konnektivität erforderlich, und bei der letzteren variiert die Genauigkeit stark, je nach den innerhalb der Reichweite befindlichen Basisstationen.**
- Die ortsbezogene Aktivitätserkennung profitiert daher vom breiten Einsatz von GPS-Sensoren, da sie Standort-schätzungen ermöglicht.
- GPS-basierte Positionsinformationen sind auf mehrere Meter genau, wenn gute Sicht auf GPS-Satelliten besteht, d.h. unter freiem Himmel. Ungenauigkeiten können durch Anwenden von Filtern basierend auf Qualitätskennzahlen und Ablehnungsmetriken relativ zu den Schätzungen früherer und nachfolgender Proben ausgeglichen werden.
- Andere Sensoren, nämlich der Kompass und das Gyroskop, sorgen für eine verbesserte Positionserkennung, um ortsbezogene Anwendungen zu verbessern. Die Standortklassifizierung kann auf Bindungen zwischen einem physischen Standort und einer generischen Klasse des Standorttyps basieren.
- Bindungen können aus Datenbanken des geografischen Informationssystems (GIS) bezogen werden.
- Soziale Netzwerke können diesen Prozess verbessern, indem sie andere Metriken bereitstellen, beispielsweise bedeutende/nachgefragte Orte.
- Die Gewährleistung der Vertraulichkeit in diesem Prozess ist eine Herausforderung, die Routing und Ressourcen für die Verwaltung erfordert, da ein mobiler Sensor nicht auf einen privaten sicheren Ort beschränkt ist,

sondern mehrere Standorte passieren kann. Darüber hinaus kann ein mobiler Sensor im Laufe der Zeit von verschiedenen Anwendungen verwendet werden.

### **Opportunistische und partizipative Wahrnehmung**

Bereits eingeführt: Zielt darauf ab, eine großflächige Erfassung zu realisieren, indem die zunehmenden Wahrnehmungsfähigkeiten von personalisierten Geräten wie Mobiltelefonen genutzt werden.

- Architektonische Herausforderungen umfassen Methoden zur Datenerfassung, Analyse und gemeinsamen Nutzung von Daten sowie den Schutz der Privatsphäre der beteiligten Personen.
- Es sollte jedoch eine wichtige Anforderung in Betracht gezogen werden: Nutzer von Sensorik-Geräten werden von Erhebungsaktivitäten, die tatsächlich recht häufig sein können, progressiv ablehnend oder desinteressiert reagieren.
- Systeme, die eine ständige Beteiligung von Menschen erfordern, um Sensing-Aktivitäten zu unterstützen, erreichen möglicherweise nicht den gewünschten Akzeptanzfaktor.

*Es ist daher notwendig, die Rolle des Einzelnen im Erfassungssystem zu definieren. Es werden zwei Modalitäten dabei unterschieden: partizipative und opportunistische Wahrnehmung.*

- Bei der **partizipativen Wahrnehmung** werden Personen in den Entscheidungsprozess über die erfassten Daten einbezogen.
  - Sie können entscheiden, welche Daten gemeinsam genutzt werden sollen, und sie haben die Kontrolle über Datenschutzfragen. Bei diesem Ansatz ist das Ziel auf eine Gruppe von Benutzern beschränkt, die an dem System teilnehmen möchten.
- Infolgedessen sollte eine teilnehmende Sensoranwendung von der Community angesprochen werden. Andererseits kann der Besitzer der sensorischen Vorrichtung (z.B. Smartphone) mit opportunistischem Erfassen unabhängig von jeder Sensoranwendung bleiben.
- Beim **opportunistischen Erfassen** nutzt ein System automatisch die Ressourcen eines Geräts, wenn sein Status (z.B. Standort oder Benutzeraktivität) den Kontextanforderungen der Anwendung entspricht.

- Trotzdem muss jede Erkennungsaktivität die Privatsphäre (z.B. der Standort des Eigentümers) und die Transparenz des Besitzers des Geräts (z.B. Vermeidung der Auswirkungen auf die Erfahrung des Benutzers) berücksichtigen.
  - ❑ Ein wesentliches Merkmal eines opportunistischen Sensorsystems besteht daher darin, zu prüfen, wann der Gerätezustand sowie die Anforderungen an den Datenschutz und die Transparenz erfüllt sind.
- Opportunistische Wahrnehmung wird möglich, wenn das System seinen Zustand als Reaktion auf eine dynamische Umgebung ändern kann.
  - ❑ Die Probenahme erfolgt nur, wenn die Anforderungen erfüllt sind und sie vollständig automatisiert ist, ohne dass Einzelpersonen am Datenerfassungsprozess beteiligt sind.
- Dies hat zur Folge, dass die Entscheidungslast von den Benutzern weggeschoben und in das System verlagert wird, was dazu führt, dass mehr Ressourcen in diesem Entscheidungsprozess benötigt werden.
- Dieser höhere Ressourcenbedarf sollte die normale Nutzungserfahrung der Erfassungsgeräte nicht spürbar beeinträchtigen.
  - ❑ Dieses Problem kann angegangen werden, wenn die opportunistische Erfassung als sekundäre Operation mit niedriger Priorität an den Erfassungsvorrichtungen betrachtet wird.
- Da Geräte die Erfassungsanforderungen jedoch möglicherweise nur für kurze und intermittierende Perioden erfüllen können, sollte ein Kompromiss zwischen Verfügbarkeit und Ressourcenmanagement in Betracht gezogen werden.

## 16.4. Der Nutzer als Sensor

### Das heißt also:

- Heutzutage hat die Popularität von Smartphones rapide zugenommen und Menschen generieren verschiedene Arten von Daten, wie Sensordaten, SNS-Posts und durch mobile Anwendungen (Apps):
  - ❑ Zugangsdaten
  - ❑ Operationale Protokollierung (Server logs ...)
- Solche vom Benutzer generierten mobilen Daten sind für viele Big-Data-Anwendungen sehr wichtig.

- Für die Datensammlung gibt es verschiedene Möglichkeiten [B]:
  1. Flussanalysen von einzelnen Personen und Personenmengen
  2. Soziale Sensoren
  3. Abstrakte Kontextanalysen
- Nutzer von mobilen Geräten können kontextbezogene Daten hoher Qualität liefern wenn diese gezielt abgefragt werden
- Zu unterscheiden sind **personifizierte Abfragen** (vertrauenswürdige Nutzer) und **anonyme Abfragen** (nicht vertrauenswürdige Nutzer und Daten)

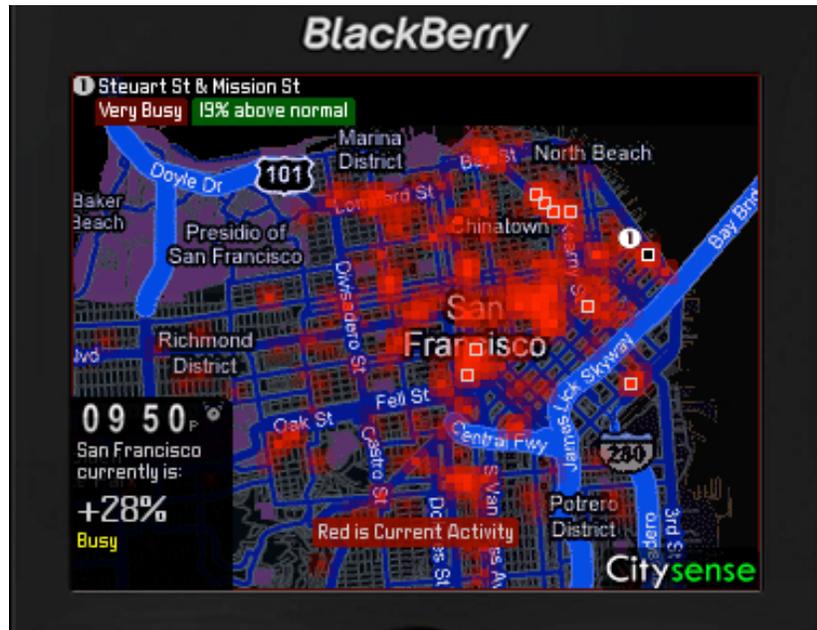
### **Flussanalyse**

*Flussanalyse ist eine typische Anwendung für mobile Big Data-Analysen.*

- **Beispiel Citysense** von Sense Networks, Inc. als ein kommerzielles Produkt, das Dienste auf Basis von Flussanalyse von Personen für eine große Anzahl von GPS-Spuren anbietet, eine Art mobiles Großgerät.
- Insbesondere bietet Citysense Dienste an, um
  - i. Die Existenz und Bewegung von Menschen in Echtzeit zu zeigen;
  - ii. Die zukünftige Bewegung von Menschen vorherzusagen.
- Dies wird erreicht, indem die Bewegungen von Personen aus einer großen Anzahl (einige Milliarden) von GPS-Datensätzen modelliert werden und dann die zukünftige Bewegung anhand des konstruierten Modells und Echtzeit-GPS-Datensätzen (einige Zehntausende von GPS) **vorhergesagt** wird.

*Wenn GPS-Daten und andere mobile Daten wie Umweltsensordaten und Web- / SNS-Daten integriert werden (oder miteinander verschmolzen werden), können neue Anwendungen damit erstellt werden, die zuvor nicht möglich waren.*

- Zum Beispiel präsentiert und prognostiziert eine Anwendung nicht nur die Bewegungen von Personen, sondern auch Gründe, warum sich die Menschen so bewegen und wie sie sich zu dieser Zeit bewegt haben.
  - Diese Anwendung ist beispielsweise für die zivile Planung und das Marketing sehr nützlich.



**Figure 64.** Citysense von Sens Network: Der Nutzer ist gleichzeitig Produzent von Daten als auch Konsument von aggregierten Informationen

### **Soziale Sensoren**

- ▶ SNS-Analyseergebnisse können als soziale Sensordaten betrachtet werden (d. h. die analytischen Verfahren können als soziale Sensoren verstanden werden).
- ▶ Wenn die sozialen Sensordaten mit Zeit und Ort (und anderen Metadaten wie Unsicherheit / Genauigkeit) verknüpft sind (gelabelte Daten), sind diese Daten für vorhandene Sensoranalyseverfahren zum Entwickeln vieler Anwendungen anwendbar.
  - z.B. zur Erkennung von Ereignissen / Ausreißern, zur Modellierung und zur Vorhersage.

### **Abstrakte Kontextbezogene Daten**

- ▶ Bei mobilen Diensten wie der mobilen Suche und ortsbezogenen Diensten ist es allgemein bekannt, dass die Berücksichtigung des Benutzerkontexts nützlich ist, um geeignete Dienste wie POI (Point of Interest, POI) (z.B.

Geschäfte und Restaurants) und Nutzerempfehlungen bereitzustellen (benutzerorientierte Schnittstellen in Apps).

- In den meisten traditionellen kontextsensitiven Diensten werden jedoch nur Kontexte auf niedriger Ebene wie Ort, Zeit und Aktivität (z. B. Gehen, Fahren und Pausieren) verwendet.
- Wenn übergeordnete Kontextinformationen wie z. B. Tätigkeiten, Allgemeinzustand, Begleitpersonen und Stimmung / Gefühl von Benutzern gesammelt werden, können basierend auf den detaillierten Kontexten der Benutzer bessere mobile Dienste erstellt werden.

### ***Nichttechnische Herausforderungen***

#### **Offene Daten (Open Data)**

Seit vielen Jahren ermutigen sowohl nationale als auch lokale Regierungen in vielen Ländern Daten offen zulegen. Verschiedene Arten von Daten sollen für die Öffentlichkeit zugänglich gemacht und für die Analyse von Big Data verwendet zu werden. Zu diesem Zweck müssen nicht nur technische Fragen, sondern auch soziale, politische und wirtschaftliche Fragen adressiert werden.

- Insbesondere Richtlinien und Gesetze, die Menschen und Unternehmen mit mobilen Daten dazu ermutigen, ihre eigenen Daten der Öffentlichkeit zugänglich zu machen.
- Außerdem sind neue Crowdsourcing-Dienste (Crowdsensing) erforderlich, um Menschen allg. oder Arbeiter zu ermutigen, mobile Daten einschließlich Sensor- und Standortdaten bereitzustellen.
- Um solche Dienstleistungen zu realisieren, sind Anreizmechanismen von großer Bedeutung.

#### **Privatssphäre**

Eines der wichtigsten Probleme ist die Privatssphäre als nächste Schritt nach der breiten Verfügbarkeit mobiler Daten besteht (d.h. mobile Daten sind offen verfügbar).

- Auch wenn die Privatssphäre jeder Person, die mit einem Datensatz verbunden ist, gewahrt bleibt (d.h. persönliche Informationen werden entfernt), können mobile Datenfusion und Big Data-Analyse verschiedene Datensätze miteinander verbinden und die privaten Informationen und Personenbezug extrahieren.
- Um dieses Problem zu lösen, sind nicht nur technische Lösungen, sondern auch einige Richtlinien für erforderlich, um die eigenen mobilen Daten

ordnungsgemäß für die Öffentlichkeit zugänglich zu machen.

### **Eigentumsrechte an Daten**

Wenn eine mobile Datenfusion durchgeführt wird, ist nicht immer bekannt wer der Eigentümer der durch die Fusion neu generierten Daten ist. Die Eigentumsrechte sind besonders wichtig, wenn die generierten Daten der Öffentlichkeit zugänglich gemacht werden.

- Daher können unberechtigte und minderwertige Daten ohne Rechtseigentum weit verbreitet sein, was für die Analyse großer Datenmengen hinderlich ist. Daher sind einige Richtlinien, Gesetze und Richtlinien zur Bestimmung des Eigentums an Daten, die durch mobile Datenfusion generiert werden, erforderlich.

### **Verfolgbarkeit der Daten**

Es ist schwierig die Historie von Daten zu verfolgen, gerade im Zusammenhang mit dem Dateneigentümerproblem, und da die mobile Datenfusion unter Verwendung mehrerer Datensätze auf hierarchische Weise durchgeführt wird (d.h Daten, die durch eine mobile Datenfusion erzeugt werden, können eine Quelle der nächsten Fusion sein usw.).

- Rückverfolgbarkeit ist wichtig, da dies, ähnlich wie bei der Urheberschaft von Daten, eine Garantie für die Richtigkeit der erzeugten Fusionsdaten sein kann. Daher sind neben den Techniken zur Rückverfolgbarkeit auch einige Richtlinien, Gesetze und Richtlinien erforderlich, um die Rückverfolgbarkeit der mobilen Datenfusion sicherzustellen.

### **Technische Herausforderungen**

#### **Verteilung und Fusion von heterogenen Daten**

Selbst wenn eine Vielzahl mobiler Daten verfügbar wird, muss es **Schnittstellen** (API) für den Austausch solcher heterogenen mobilen Daten geben.

- Um die Entwicklung von Systemen / Anwendungen zu ermöglichen, ist es außerdem notwendig, Mechanismen zur Unterstützung der Fusion heterogener mobiler Daten zu nutzen, wie zum Beispiel das Verbergen der Heterogenität physikalischer Sensoren und die Unterstützung der Entwicklung von Fusionsalgorithmen (Transparenz).

### **Ontologien**

Die notwendigen Programmierschnittstellen sind eng mit der Frage der Dateninterpretation und Semantik verknüpft. Es muss daher Ontologien geben, die Daten einen semantischen Kontext und Verknüpfungen zuordnen!

### **Verteilung von sozialen Sensordaten**

Benutzergenerierte SNS-Nachrichten können zum Erstellen von sozialen Sensordaten verwendet werden. Dies geschieht durch Anwendung bestimmter Analysemethoden. Obwohl eine große Anzahl von Studien durchgeführt wurde, um verschiedene Arten von Informationen aus SNS-Daten (den Sozialsensordaten) zu extrahieren, werden i.A. die analytischen Ergebnisse nicht geteilt können eines der wichtigsten Probleme die Privatsphäre.

- Wenn die Analyseergebnisse als (virtuelle) soziale Sensordaten veröffentlicht werden können, ist dies für Entwickler verschiedener Anwendungen von großem Nutzen.

### **Kontext**

Abstrakte Kontextinformationen der Benutzer sind nützliche mobile Daten, die zur Verbesserung verschiedener Anwendungen wie der mobilen Suche verwendet werden können. Es ist jedoch schwierig, einen solchen abstrakten Kontext zu erfassen, da es oft schwierig ist, Kontext- (Situations-) Informationen zu melden und zu charakterisieren.

### **Wahrung der Privatsphäre**

Um die Privatsphäre der Benutzer zu schützen wenn mobile Datenfusion durchgeführt wird, sind einige technische Ansätze erforderlich, um automatisch zu überprüfen, ob die mobile Datenfusion die privaten Informationen der Benutzer nicht preisgibt.

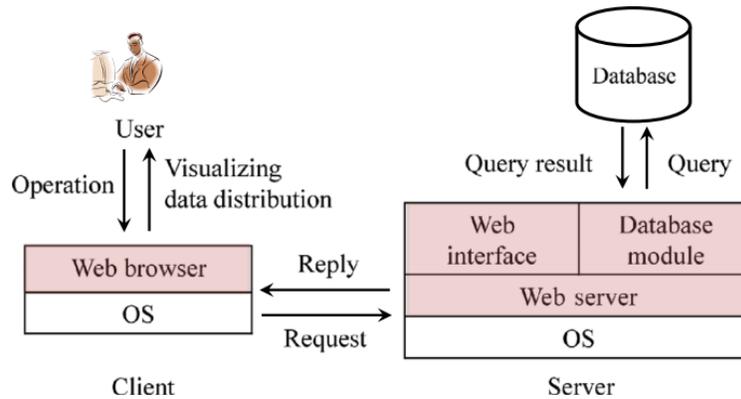
### **Tracking**

Um die Rückverfolgbarkeit auch dann zu erreichen, wenn die mobile Datenfusion aufeinanderfolgend / hierarchisch durchgeführt wird, sind einige Mechanismen erforderlich, um den Verlauf der Fusionsprozesse zu verfolgen und die nachverfolgten Informationen für Entwickler zugänglich zu machen → Tagged Data.

## **16.5. WEB basierte Dienste**

- WEB basierte Dienste sind bisher die häufigsten Mensch-Sensor-Maschine-Erheber Schnittstellen!
- Vorteil: Überall verfügbar und benutzbar bei Internetkonnektivität

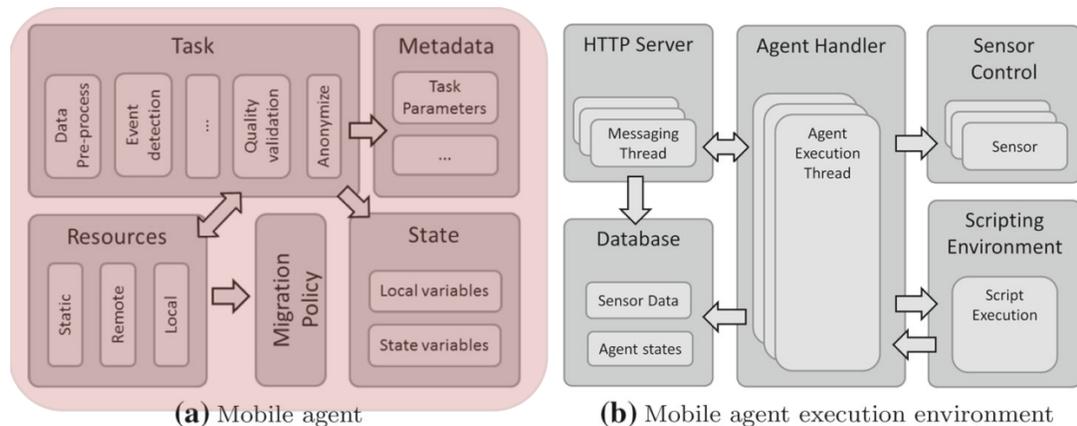
- Kann auch mittels WEB Browser realisiert werden.
- Aber: Nur Nutzerinteraktion und ggfs. Positionserfassung möglich - (fast) kein Zugriff auf physische Sensoren des Gerätes



**Figure 65.** Klassische Crowdsensingarchitektur mit WEB Dienst (SeRAVi Plattform) [B]

### 16.6. Agentenbasierter Dienst

- Verwendung mobiler Agenten über eine HTTP WEB Schnittstelle als einfachste Kommunikations- und Interaktionsarchitektur



[3]

**Figure 66.** WEB basierte Crowdsensingarchitektur (b) und Komponenten des Agenten (a)

## **Ressourcen**

Folgende Ressourcen gibt es:

1. Teilnehmergeräte für Erkundungsaufgaben und als Agentenausführungsplattformen;
  2. Integrierte physische Sensoren, einschließlich GPS-Empfänger, WiFi-Radio, Kamera und Mikrofon;
  3. Rohe und aufbereitete Sensordaten; und
  4. Mobile Agenten durch ihre (Daten)Zustände.
- Jede Ressource ist eindeutig identifiziert und kann mit einer URL angesprochen werden.
  - HTTP wird als standardisiertes universelles Kommunikationsprotokoll verwendet.
  - Daher muss jede Systemkomponente einen HTTP-Server ausführen.
  - Die HTTP-Methoden und ihre entsprechenden Antwortcodes werden als generische Grundelemente in der Kommunikation innerhalb des Systems verwendet, einschließlich der Kommunikation zwischen Agenten und zwischen Agenten.
  - Dadurch werden externe Web-Services nahtlos als Datenquellen für die Erhebung integriert.
  - Darüber hinaus umfasst HTTP die Aushandlung von Inhalt und Sicherheitsmechanismen.

## **Aufgaben**

Folgende wichtige Aufgaben werden von den Agenten wahrgenommen:

1. **Datenvorverarbeitung und Fusion**
2. **Ereigniserkennung und Verarbeitung**
3. **Validierung der Daten und Qualitätssicherung**
4. **Anonymisierung und Verschlüsselung**

### **Agentenarchitektur**

- Die interne Architektur des mobilen Agenten entspricht dem ATG Modell wo:
  - ❑ Das Verhalten aufgabenorientiert ist (in Tasks unterteilt);
  - ❑ Pfeile Abhängigkeiten zwischen den Komponenten zeigen;
  - ❑ Der Zustand z.B. von einer Aufgabe (Task) aktualisiert wird; und
  - ❑ Die Aufgaben sind in verschiedene Funktionen unterteilt, die aufgabenspezifische Datenverarbeitungsalgorithmen und ihre Interaktionen implementieren.
  
- Aufgaben können Erfassungsparameter wie die Abtastrate steuern und den physischen Sensor aktivieren.
- Aufgabenfunktionalitäten sind in der Agentenzusammensetzung gekennzeichnet und können in jeder Programmiersprache implementiert werden, sofern sie mit sprachspezifischen Bezeichnern versehen sind → *AgentJS*.
- Dies erleichtert die **Trennung von Verhalten und Aufgaben**, sowie die Änderung der Agentenfunktionalität zur Laufzeit und die Erstellung von **übergeordneten Plänen** für MAS.

### **Dialoge**

- Mobile Agenten können über Dialoge und Sensorschnittstellen Nutzerdaten und Kontext erlangen
- Chat bots sind bekannte Programme die mit Nutzern in einen Dialog treten können
- Ein Dialog besteht aus Nachrichten, Fragen, und Antworten
- Man unterscheidet statische und dynamische Dialoge, die sich aufgrund von Antworten dynamisch, z.B. mit einer Baumstruktur, entwickeln

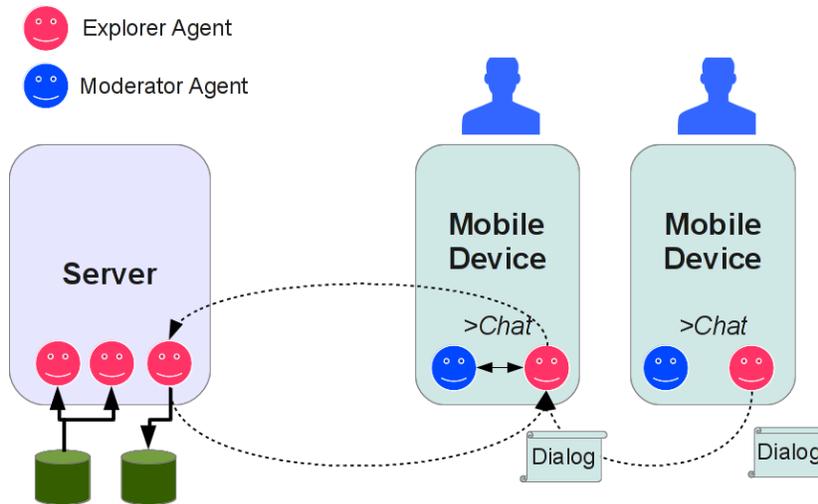


Figure 67. Mobile Agenten übernehmen die Erhebung sozialer mobiler Daten mittels Agent-Agent Kommunikation und Verhandlung

## 16.7. Dialoge

### *Chat Dialog*

- Ein Dialog besteht aus einer Sequenz von
  - ❑ Aussagen oder Informationen (Nachrichten)
  - ❑ Fragen
  - ❑ Antworten

### *Moderatoragent*

- Werden mobile Agenten verwendet um soziale, sozio-technische, und personenbezogene Daten auf mobile Geräten zu erheben ist eine Moderationsinstanz auf dem jeweiligen Geräte erforderlich → **Moderatoragent**
- Der Moderatoragent wird Nachrichten und Frage-Antwortdialoge zur Verarbeitung annehmen und folgende Kriterien bewerten:
  - ❑ Authentifizierung des Fragestellers
  - ❑ Autorisierung der Anfrage (z.B. geschlossene peer-to-peer Umfrage)
  - ❑ Inhaltliche Bewertung: Spam? Sinn? Kontext?

- Wiederholung (Frage schon - von anderen Agenten - gestellt?)
- Reihenfolge und Rückmeldung des Nutzers

### **Chat Dialog in der JAMapp**

- Agenten können dem lokalen Moderatoragenten via **Tupelraum** verschiedene Nachrichten und Fragen zur Bearbeitung übergeben

#### **Nachricht**

`['MESSAGE', id:string, message:string, capability?]`

Eine Nachricht für den Nutzer ausgeben. Dabei wird die Nachricht mit der angegebenen Identität ausgegeben. Ein optional Capability kann zur Autorisierung verwendet werden.

#### **Frage**

`['QUESTION', id:string, question:string, {choices:[]}|{}, capability?]`

Ein Frage wird gestellt. Dabei wird die Nachricht mit der angegebenen Identität ausgegeben. Eine optionale Capability kann zur Autorisierung verwendet werden.

#### **Antwort**

`['ANSWER', id:string, question:string, _, capability?]`

Warten auf eine Antwort zu der gestellten Frage

```
var i=0,dialog = [
  {text:'Hello, I am a friend!'},
  {question:'How are you?' },
  {question:'Where are you?', choices:['Street 1','Street 2'],'Street 1'},
  {question:'How do you rate ambient light?', choices:['Dark','Good'],'Good'},
  {question:'Make a guess of room temperature:', value:30, default:30}
]
out(['MESSAGE',id,dialog[0].text])
L(function () { i=1 },
  function () { return i < dialog.length },
  function () { i++ },
  function () {
    var options={
      choices:dialog[i].choices,
      default:dialog[i].default,
      value: dialog[i].value };
    out(['QUESTION',id,dialog[i].question,options])
    inp.try(timeout,['ANSWER',id,dialog[i].question,_], function (t) {
      if (t) dialog[1].answer=t[3];
    })
  })
})
```

Beispiele von Dialogen

## 16.8. Physische Sensoren

*Smartphones*

*IoT*

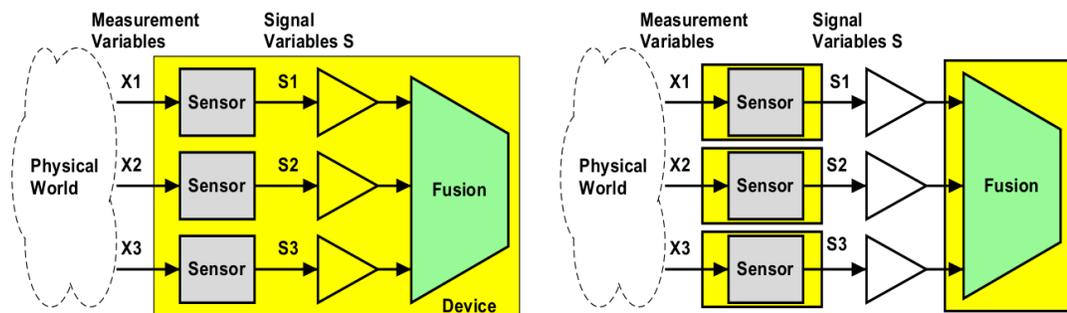
**Gassensoren**

- Umweltüberwachung → außerhalb und innerhalb von Gebäuden
- Lebensmittelüberwachung

## 16.9. Virtuelle Sensoren

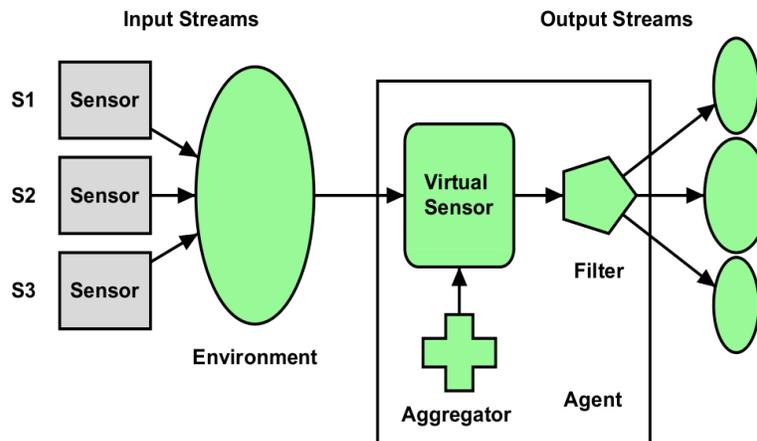
- Physikalische Sensoren sind je nach Geräteklasse unzuverlässig und verarscht:
- Die Fusion mehrerer Sensoren kann die Genauigkeit und Sicherheit erhöhen, indem Sie Folgendes verwenden:
  - ❑ Sensoren, die dieselbe physikalische Größe messen (Mittelwertfilter)

- ❑ Sensoren, die verschiedene physikalische Größen messen (z.B. Dehnung + Temperatur)
- ❑ Verteilte korrelierte Sensoren in einer Region
- ❑ Verteilte unkorrelierte, aber markierte Sensoren (räumlich / benutzerbezogen)



**Figure 68.** (Links) Integrierte Fusion (Rechts) Externe Fusion

- Die Umgebung eines Sensors besteht aus einer Reihe von Eingabeströmen von Daten, die von physischen oder virtuellen Sensoren generiert werden.
- Die Umgebung definiert den Kontext innerhalb des virtuellen Sensors.
- Der Aggregator verarbeitet die Eingabeströme → Fusion
- Der Filter erzeugt eine Reihe von Ausgabeströmen
- **Agenten können virtuelle Sensoren darstellen und implementieren!**



**Figure 69.** (Links) Virtuelle Sensoren operieren auf Eingabeströmen von Daten und erzeugen Ausgabeströme (Rechts) Agenten implementieren virtuelle Sensoren

- Agenten implementieren den Sensordaten-Aggregator und die Filterfunktion
- Agenten sind Fusions-, Speicher- und Kommunikations- / Transporteinheiten
- Die Agentenverarbeitung wird von einer vereinheitlichten Agentenverarbeitungsplattform gekapselt

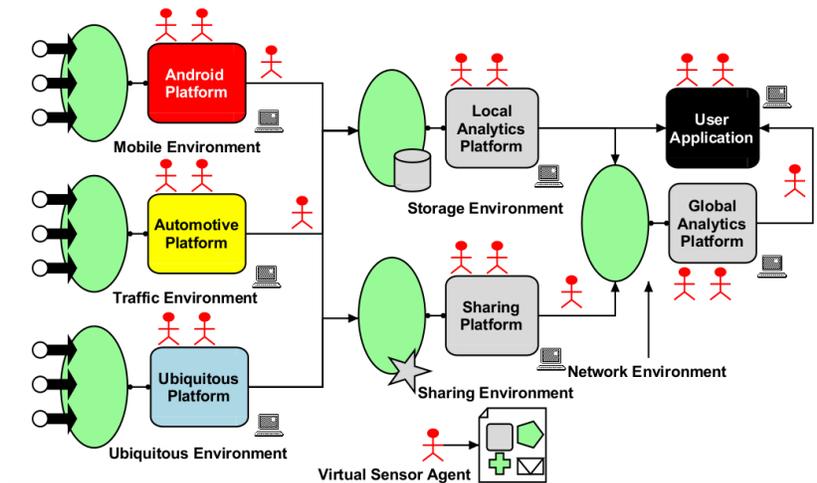


Figure 70. Hierarchische Architektur als Netzwerk aus virtuellen Sensoren: Agenten sind virtuelle Sensoren

## 16.10. Aggregation und Fusion

Um Crowdsensing Anwendungen zu entwickeln, ist es wichtig, verschiedene sensorische Datentypen und Quellen (d.h. heterogene mobile Daten) vollständig zu nutzen.

- Zu diesem Zweck müssen diese unterschiedlichen mobilen Daten miteinander kombiniert werden.
  - ❑ Daten aus Sozialen Netzwerken (Social Network Services, SNS)
  - ❑ Gerätedaten ..
- Dies ist jedoch aus mehreren Gründen nicht einfach [B, Kap. 3]:
  1. Das Volumen und die Vielfalt mobiler Daten sind extrem gross.
    - Daher sind traditionelle Techniken und Plattformen in vielen Situationen nicht geeignet oder ausreichend.
  2. Verschiedene Arten von Kontextinformation (z.B. Beschäftigung, Gesundheitszustand, mentaler Zustand, Begleitpersonen, Gefühle und Meinung), die direkt von Benutzern (also gewöhnlichen Personen oder Arbeitern) erlangt werden sollten, sind im Gegensatz zu herkömmlichen Sensordaten **schwierig automatisch zu erfassen**.
- Weiterhin spielt die Datenteilung eine zentrale Rolle:

3. Obwohl viele analytische SNS-Methoden existieren um verschiedene Informationen wie Trend, Ereignisse und das “Befinden” der Benutzer anzuleiten, werden die Analyseergebnisse i.A. nicht mit anderen Benutzern und Anwendungen geteilt (→ isolierte Anwendungen).
  - D.h. die SNS-Analyseergebnisse sind zwar gesellschaftsrelevante soziale Sensordaten, werden aber über keine allgemein zugängliche Plattform zum Austausch verfügbar sein, was Sensorfusion erschwert.

### **Soziale Sensoren**

#### Zusammenfassung

1. Ableitung von epidemeiologischen und soziomedizinischen Informationen aus physikalischen Sensoren (Smartphones usw.)
2. Psycho-physiologische Informationen → Z.B. Stresssensoren mit Smart Watches
3. Aus physikalischer Nähe auf soziale Nähe schließen → Gruppen in sozialen Medien
4. Sozialer Kontext
5. Soziologische Umfragen mit spezifischen Fragestellungen → Modellprüfung
6. Meinungen → Meinungsforschung (Gesellschaft, Politik)
7. Nutzbarmachung von Expertenwissen und Expertenmeinung
8. TBC

### **Nähe und Kontext**

- MAS und Crowdsensing können zur Bestimmung von Nähe und Kontext verwendet werden, beruhend auf Sensorfusion.
- Man unterscheidet:
  - ❑ Physikalische und räumliche Nähe von Geräten und Personen
  - ❑ Soziale Nähe
  - ❑ Nähe durch Kontext (wie z.B. Meinungen, Vorlieben, ..)

- Fusion von:
  - ❑ Lokalisation mit GPS Daten
  - ❑ Verhaltensdaten (mit lokalem Bezug, z.B. abgeleitet aus Daten von Bewegungssensoren)
  - ❑ IP Lokalisation (Providerstandortinformation)
  - ❑ WLAN (IP, Accesspoint und Gerätescans) und Verwendung von Standortdaten von WLAN Accesspoints
  - ❑ Bluetooth Scan

### **Kontextuelle Wahrnehmung**

[J]5.5

## **16.11. Aggregation und Maschinelles Lernen**

- Neben der klassischen Fusion als Aggregation von Daten können maschinelle Lernverfahren verwendet werden um wesentliche Eigenschaften aus den Daten zu extrahieren:
  - ❑ Feature Selection: Auswahl von Datenvariablen mit hohem Informationsgehalt
  - ❑ Feature Extraction: Ableitung von neuen reduzierten Informationen aus selektierten Datenvariablen → Klassifikation und Prädiktion, Clustering

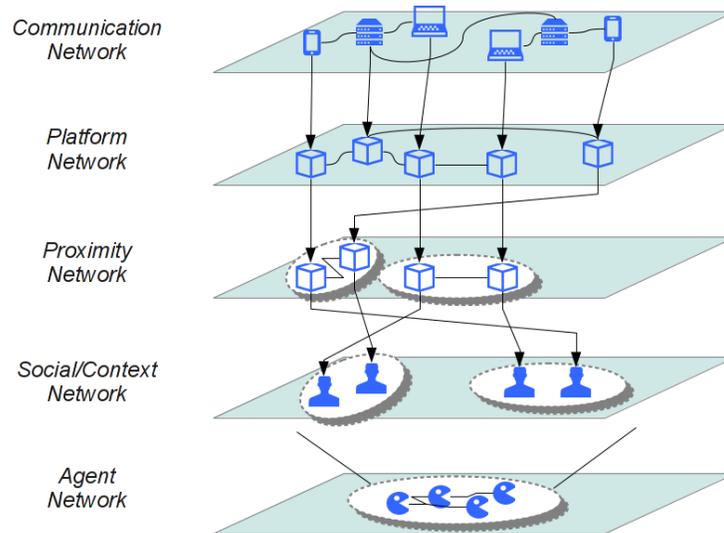
## **17. Organisationsstrukturen**

### **17.1. Netzwerke**

Man unterscheidet verschiedene Netzwerkebenen die mit gerichteten oder ungerichteten Graphen repräsentiert werden können:

- Physische Netzwerke → Verbindungen stellen räumlichen Kontext dar, der mit Situationen, örtlichen Gegebenheiten und weiteren geographischen Eigenschaften verknüpft sein kann;
- Logische Netzwerke → Verbindungen stellen Kommunikationsstrukturen (IP, Internet, ..) und logischen Kontext dar

- Organisatorische Netzwerke → Verbindungen stellen Organisation zwischen Plattformen und Agenten dar
- Soziale Netzwerke → Verbindungen stellen soziale Verbindungen dar
- Die Graphen auf den unterschiedlichen Ebenen (horizontale Konnektivität) können miteinander verbunden sein (vertikale Konnektivität)



**Figure 71.** Verschiedene Netzwerkebenen und deren vertikale Verbindungen

## 17.2. Räumliche Nähe

- Sensordaten haben häufig einen räumlichen Kontext den man als räumliche Nähe bezeichnet
- So können Bewegungsprofile verschiedener Geräte, die sich in einem räumlichen Kontext befinden, genutzt werden um auf Gruppensituationen zu schließen
- Aber auch Meinungsbildung kann einen räumlichen Kontext haben (situationabhängig). Dabei gibt es kurzzeitige (opportunistische) und langfristige Meinungsbildung (auch politische Meinung usw.)
- Ein Raum kann definiert sein durch:
  - ❑ Geographische Metriken

- ❑ Soziale Bindungen und Kontakte
- ❑ Konstrukte (wie Gebäude)
- ❑ Verbindungen wie Straßen oder in Gebäuden
- ❑ Kommunikationsnetzwerke und Kommunikationsplattformen (soziale Medien)

## 18. Sicherheit

### 18.1. Ebenen der Sicherheit

1. Plattform und Ausführungsebene
2. Kommunikationsebene
  - Gerät-Gerät
  - Agent-Gerät
  - Agent-Mensch
  - Agent-Agent
3. Nutzerebene
4. Datenebene

### 18.2. Sicherheitsrisiken

#### Denial-of-Service (DOS)

Mutwillig herbeigeführte Einschränkung eines Services (z.B. der einer Agentenplattform) durch Überlastung mit einer größeren Anzahl Anfragen, als der Service diese verarbeiten kann, woraufhin reguläre Anfragen nicht oder nur sehr langsam beantwortet werden. Wenn möglich, ist es jedoch wesentlich effizienter, Programmfehler auszunutzen, um eine Fehlerfunktion (wie einen Absturz) der Serversoftware auszulösen, worauf diese auf Anfragen ebenfalls nicht mehr reagiert.

#### Man-in-the-middle-Attack (MitmA)

Der Angreifer steht entweder physisch oder logisch zwischen zwei Kommunikationspartnern, hat dabei mit seinem System vollständige Kontrolle über den Datenverkehr zwischen zwei oder mehreren Netzwerkteilnehmern und kann die Informationen nach Belieben einsehen und sogar manipulieren.

Auch (einfache) Verschlüsselung ist unwirksam, solange sie nicht auf Rechnerzertifikaten beruht, da es nun zwei getrennte Punktverbindungen mit Verschlüsselung und Schlüsseltasuch gibt.

#### **Datenmissbrauch, Spionage**

xxx

#### **Datenmanipulation**

xxx

### **18.3. Multiagentsysteme und Plattformen**

#### **Aufgabe**

Betrachte JAM Agenten und die JAM Plattform.

1. Welche Sicherheitslücken könnte es geben?
2. Wo und wie könnten Denial-of-Service Angriffe ansetzen?
3. Wo und wie könnten Man-in-the-Middle Angriffe ansetzen?
4. Werden vertrauliche Daten geschützt?
5. Was könnte unter Beachtung einer einfachen und kompakten Plattform/Kommunikation verbessert werden? Welche Rolle könnten die bereits unterstützten Capability Konzepte spielen?

---

#### **Quotas**

1. Ausführungszeit
  - Scheduling von Agenten (round-robinson, Fairness)
  - Gesamte Ausführungszeit als belegbare Ressource des Agenten auf eine Plattform → verhandelbar mit Capabilities
2. Sandkasten/Container → Virtualisierung der Ausführung von Agenten
  - Zugriff auf Ressourcen (Tupelräume, Speicher, usw.)
  - Programmierschnittstelle und privilegierte Rollen
  - Erzeugung von Agenten (als belegbare Ressource)

### 3. Ensemblelimitierungen

- ▶ Begrenzung der pro Zeitintervall ausgeführten Agenten
- ▶ Ausschluss bestimmter Agenten oder Agentenklassen
- ▶ Hemmung der Mobilität und Replikation

## 18.4. Schutz privater Daten

- ▶ Viele Crowdsensing-Anwendungen sammeln Daten von Teilnehmern, die dann zur Identifizierung dieser Personen verwendet werden können.
- ▶ Solche Daten können zum Beispiel unterschiedliche Vorlieben und Interessen eines bestimmten Benutzers sowie Standortdaten umfassen.
- ▶ Opportunistisches Crowdsensing stellt ein größeres Problem hinsichtlich des Datenschutzes als teilnehmendes Crowdsensing dar.
  - ❑ Dies liegt daran, dass die Benutzer von Anwendungen, die opportunistisches Crowdsensing durchführen, keine direkte Kontrolle über die von ihnen übermittelten Daten haben.
  - ❑ Zum Beispiel assoziieren die automatisch auf Smartphones gesammelten GPS-Daten den Standort eines Benutzers einfach mit anderen Sensordaten.
  - ❑ Wenn diese Standortverfolgung jedoch auf der Grundlage der historischen Daten zeitlich zusammengefasst wird, können andere Benutzerinformationen abgeleitet werden.

GPS-Sensorerfassung kann verwendet werden, um private Informationen über einzelne Benutzer abzuleiten, z.B. die Routen, die sie während ihrer täglichen Arbeitswege nehmen, sowie ihren Wohn- und Arbeitsort.

- ▶ Die täglichen GPS-Sensormessungen für den Verkehr können jedoch innerhalb eines größeren Gebietes geteilt werden, um die Verkehrsteuerung in einer bestimmten Stadt zu verbessern. Daher ist es wichtig, die Privatsphäre von Einzelpersonen zu wahren und gleichzeitig mobile Crowdsensing-Anwendungen zu ermöglichen.
- ▶ Folgende Methoden können angewendet werden um private und personalisierte Daten zu schützen:

### **Anonymisierung**

Ein Ansatz zum Schutz der Privatsphäre der Daten ist die Anonymisierung. Anonymisierung bedeutet die identifizierenden Informationen zu entfernen, die von Crowdsensing-Anwendungen erfasst werden.

- Es gibt jedoch zwei Probleme.
  - ❑ Erstens kann die bloße Entfernung identifizierender Informationen wie Namen und Adressen aus den Daten keine Anonymität garantieren. Wenn beispielsweise Crowdsensing-Anwendungen Standortdaten sammeln, kann die Anonymisierung die Identifizierung von Personen nicht verhindern. Der Grund ist, dass anonymisierte Standortdaten immer noch die häufig besuchten Standorte einer Person anzeigen, was wiederum zur Identifizierung dieser Person führen kann.
  - ❑ Zweitens sind im Zusammenhang mit der Anonymisierung von Daten Datennutzen und Datenschutz widersprüchliche Ziele. Infolgedessen verbessert die Anonymisierung der Daten den Schutz der Privatsphäre, verringert jedoch den Nutzen der Daten.

### **Verschlüsselung**

Die Vertraulichkeit der persönlichen Daten der Benutzer kann durch die Verwendung von Verschlüsselungstechniken geschützt werden. Durch die Verschlüsselung der von den Benutzern übermittelten Daten können unbefugte Dritte keine personenbezogenen Daten verwenden, selbst wenn sie Zugang zu den verschlüsselten Daten erhalten. Solche kryptographischen Techniken können jedoch rechenintensiv sein, was zu einem erhöhten Energieverbrauch führt, und sind möglicherweise nicht skalierbar, da sie die Erzeugung und Pflege mehrerer Schlüssel erfordern.

### **Störung der Daten**

Datenstörung bezieht sich auf das Hinzufügen von Rauschen zu Sensordaten, bevor diese von den Einzelpersonen eingereicht werden. Infolge der Störung sind die von Einzelpersonen übermittelten Daten nicht identifizierbar. Trotzdem würden solche Daten einen guten Betrieb von Crowdsensing-Anwendungen ermöglichen.

- Eine beliebte Form der Datenstörung ist die Mikroaggregation. Der Begriff Mikroaggregation bezieht sich auf das Ersetzen eines bestimmten Feldes durch einen aggregierten oder allgemeineren Wert.
  - ❑ Die Ersetzung einer Postleitzahl durch den Namen einer Provinz oder eines Staates ist ein typisches Beispiel für eine Mikroaggregation.
- Die Mikroaggregation kann operativ in zwei Schritten definiert werden, nämlich Partitionierung und Aggregation. Partition bezieht sich auf die Partitionierung des Datensatzes in mehrere Teile (Gruppen, Cluster).
  - ❑ Die Aggregation bezieht sich auf das Ersetzen jedes Datensatzes durch einen Durchschnittswert.

### **Lösungen basierend auf K-Anonymität**

- ▶ K-Anonymität ist eine wichtige Maßnahme zum Schutz der Privatsphäre, um die Offenlegung personenbezogener Daten zu verhindern. Eine Tabelle erfüllt die K-Anonymität, wenn jeder Datensatz in der Tabelle in Bezug auf jeden Satz von Quasi-Identifizier-Attributen von mindestens K-1 anderen Datensätzen **nicht unterscheidbar** ist.
- ▶ Im Kontext von Standortprivatheit kann das Ortsattribut als Quasi-Bezeichner betrachtet werden. Der anonyme Datenschutz von anonymen Standorten bedeutet daher, dass der Standort des Benutzers nicht von mindestens K-1 anderen Benutzern unterschieden werden kann.
- ▶ Um einen anonymen Standortschutz für K-Standorte zu erreichen, besteht ein gängiger Ansatz darin, einen als Anonymisierer bezeichneten Vertrauensserver einzubauen, der dafür verantwortlich ist, die Benutzer-ID zu entfernen und einen anonymisierenden räumlichen Bereich (ASR) auszuwählen, der den Benutzer und mindestens K-1 Benutzer enthält die Umgebung

### **Weiterführende Literatur**

- ▶ [A] C. Borcea, M. Talasila, and R. Curtmola, *Mobile Crowdsensing*, Kapitel *Privacy Concerns and Solutions*

## **18.5. Explorationsstrategien mit MAS**

- ▶ Es gibt verschiedene Ansätze Crowdsensing im Sinne einer Umfrage durchzuführen:

### **Zentrale Instanz**

Es gibt eine zentrale Instanz (Rechner/Plattform/Datenbank/Cloud) die Fragen und Dialoge formuliert, die Nutzer gezielt für eine Umfrage einlädt, und Agenten gerichtet zu den Nutzern aus einer bekannten Menge zwecks Umfrage sendet.

### **Multiple Koordinierte Instanzen**

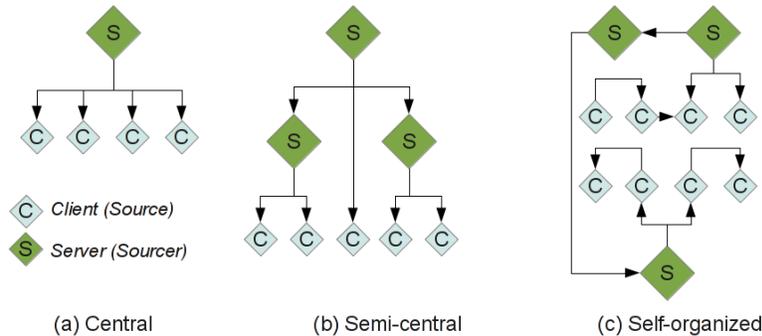
Es gibt mehrere Instanzen, die semiverteilt die Umfrage durchführen. Aber weiterhin Koordination (Konsens, Aufgabenverteilung, usw.), gezielte Einladung, bekannte Menge von Nutzern.

### **Selbstorganisierendes System**

Es gibt keine ausgewiesene zentrale Instanz. Die Umfrage ist eher ad-hoc und mit dynamischer Menge von Nutzern. Mobile Agenten können selbst entscheiden auf welchen Geräten und mit welchen Nutzern ein Dialog geführt

wird. Emergenz des MAS ist eine globale Fragestellung oder Aufgabe die zu lösen ist.

- Beim selbstorganisierenden Ansatz können auch Nutzer /Agenten/ Mikroumfragen (in einem räumlichen Kontext) anstoßen!



**Figure 72.** Verschiedene Crowdsensingstrategien mit mobile Agenten

## 19. Literatur und Referenzen

### 19.1. Bücher

- C. Borcea, M. Talasila, and R. Curtmola, *Mobile Crowdsensing*. CRC Press, 2017.
- M. Migliardi, A. Merlo, and S. A.-H. Baddar, Eds., *Adaptive Mobile Computing*. Elsevier, 2017.
- D. C. Brabham, *Crowdsourcing*. MIT Press, 2013.
- J.-F. Lebraty and K. Lobre-Lebraty, *Crowdsourcing - One Step Beyond*. JohnWiley & Sons, Inc., 2013.
- J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. 2012.
- I. Foster, R. Ghani, C. Bureau, F. Kreuter, and J. Lane, Eds., *Big data and social science : a practical guide to methods and tools*. Taylor & Francis, 2017.
- B. Furht and A. Escalante, *Handbook of Cloud Computing*. Boston, MA: Springer US, 2010.

- H. Z. Yan and D. Chakraborty, *Semantics in Mobile Sensing*. MORGAN&CLAYPOOL, 2014.
- I. D. Schall, *Service-Oriented Crowdsourcing - Architecture, Protocols and Algorithms*. Springer, 2012.
- J. M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, Ltd, 2008/2011.
- K. A. R. Bar and M. Maheswaran, *Confidentiality and Integrity in Crowdsourcing Systems*. Springer Cham Heidelberg NewYork Dordrecht London, 2014.
- L. F. Xhafa, N. Bessis, *Inter-cooperative Collective Intelligence: Techniques and Applications*, Springer, 2014
- M. A. M. Arsénio and A. M. Arsénio, *Internet of Intelligent Things: Bringing Artificial Intelligence into Things and Communication Networks*, in *Inter-cooperative Collective Intelligence: Techniques and Applications*, Springer, 2014.
- N. J. BETHLEHEM, *Applied Survey Methods A Statistical Perspective*. John Wiley & Sons, 2009.
- O. F. Hou, Y. Pei, and J. Sun, *Mobile Crowd Sensing: Incentive Mechanism Design*. Springer, 2018.

## 19.2. Publikationen

1. D. Yang, G. Xue, X. Fang, and J. Tang, *Incentive Mechanisms for Crowdsensing: Crowdsourcing with Smartphones*, IEEE/ACM Transactions on Networking, vol. 99, 2015.
2. R. Gaire et al., *Crowdsensing and privacy in smart city applications*, in *Smart Cities, Cybersecurity and Privacy*, 2018.
3. T. Leppänen, J. Á. Lacasia, Y. Tobe, K. Sezaki, and J. Riekkki, *Mobile crowdsensing with mobile agents*, Proc. of th 15th International Conference on Autonomous Agents and Multiagent System (AAMAS), 2016.
4. O. Alvear, C. T. Calafate, J.-C. Cano, and P. Manzoni, *Crowdsensing in Smart Cities: Overview, Platforms, and Environment Sensing Issues*, Sensors, vol. 18, no. 460, 2018.

### **19.3. Folien**

- a. [courses.cs.washington.edu](http://courses.cs.washington.edu), Map and Reduce - Theory and Implementation, 2007