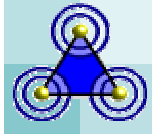
Herausforderungen bei der Softwareentwicklung für Sensor-Netzwerke

Jan Blumenthal, Dirk Timmermann
Universität Rostock

DFG Workshop “X-Layering”, SPP 1102 & SPP 1140
Frankfurt/Main, 24. November 2003



MD

Gliederung

- Projekt-Vorstellung
- Einführung zu Sensor-Netzwerken
- Aktuelle Arbeiten
- Zusammenfassung

Projekt-Informationen

Projekt:

Middleware für mobile spontan vernetzte Sensornetzwerke

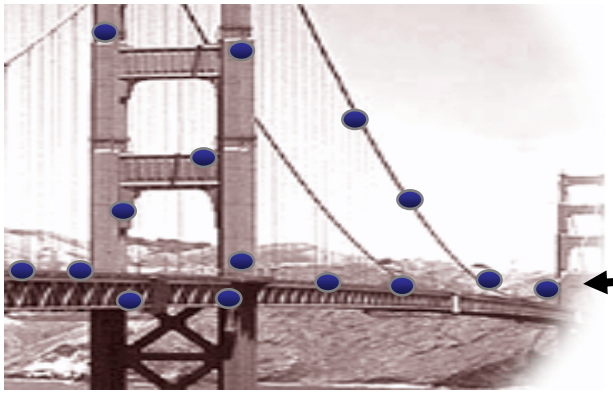
Ziele:

- Untersuchung von Verfahren:
  - Selbstkonfiguration von ad-hoc vernetzen Sensornetzwerken
  - Kooperatives Zusammenarbeiten unabhängiger Knoten
  - Fehlertoleranz
  - Kontextabhängigkeit der zur Verfügung stehenden Dienste
- Lokationsbestimmung
- Adaptierbare, wiederverwendbare Middleware für Sensor-Netzwerke
- Entwicklung einer Referenzinfrastruktur

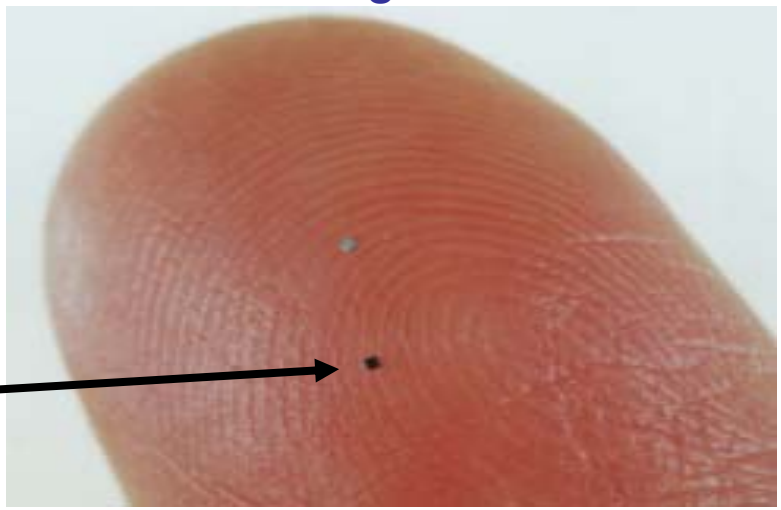
Charakteristik: Sensor-Knoten

- Limitierungen:
  - Speicher (Flash/ROM: 8KB, RAM: 512 Byte)
  - Batterie (Lebenszeit: Tage – 10 Jahre)
  - Begrenzte Rechenfähigkeit
  - Übertragungsreichweite (5…20 m)
  - Datenraten: Bit/s … KB/s
- Funktechnologie:
  - Bluetooth
  - ZigBee
  - proprietär: nanoNET, Chipcon, ...

Beispiel: Materialüberwachung



Zukünftige Sensorknoten



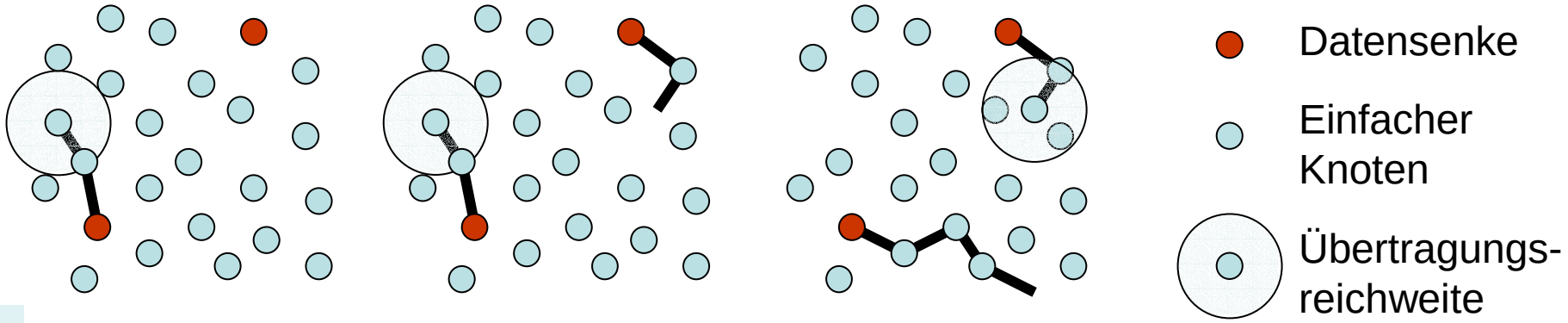
Charakteristik: Sensor-Netzwerk

- Selbstorganisation
  - Ad hoc Netzwerkbildung
  - Autonomer Verbindungsaufbau
  - Mobilitätsbehandlung
- Netzwerkpflege
  - Adressierung von Knoten, Routing
  - Adaptive Reaktionen auf Umgebungsänderungen
  - Kompensation von Ausfällen
- Kooperative Datenverarbeitung
  - Context awareness
  - Location positioning und location awareness
  - Messdatenreduktion

Context Awareness

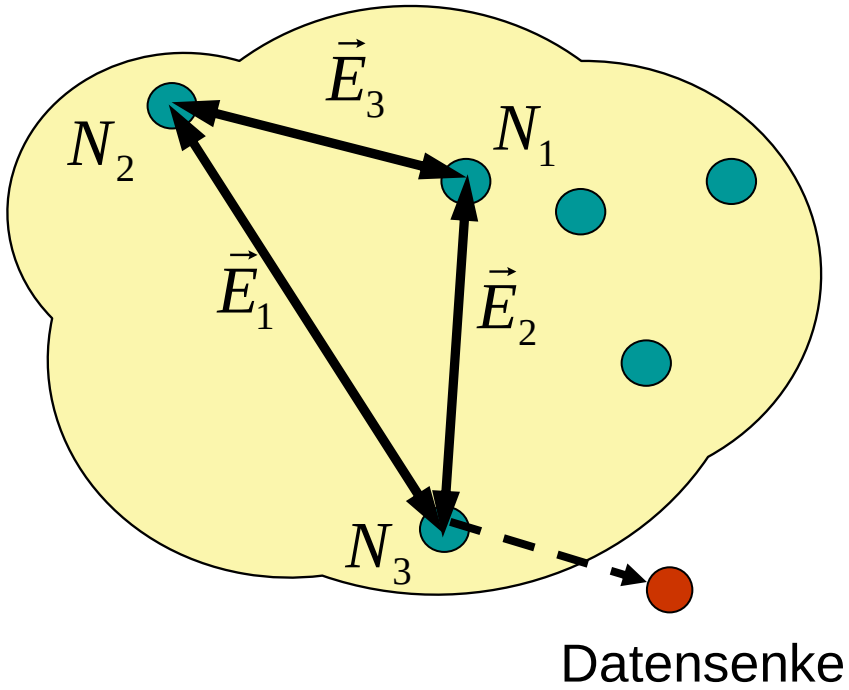
Context Awareness: Anpassen des Verhaltens eines Knotens an seine aktuelle Umgebung.

- Infrastruktur-Kontext
  - Beziehung zur umgebenden **Infrastruktur** eines Knotens
  - Beispiel: Abschätzung der Bandbreite und der Zuverlässigkeit
- Domain-Kontext
  - Beziehung zur **Systemumgebung** eines Knotens
  - Beispiel: Kenntnis über Datensenken im Netzwerk



Kooperative Algorithmen

- Reduktion der Daten durch Datenvorverarbeitung und Datenaggregation
- Minimierung des Datenstromes zur Datensenke



Beispiel: Ortsbestimmung

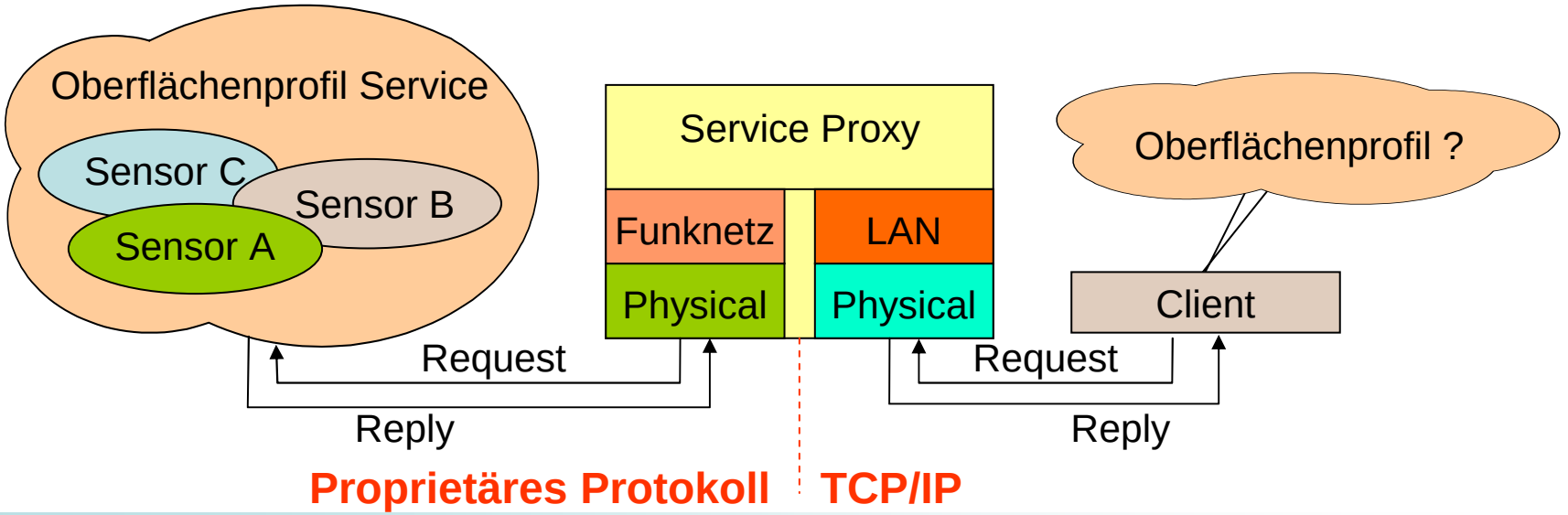
- Übertragung von $\vec{E}_3$ nach N_3
- Messen der Feldstärke von $\vec{E}_1$ und $\vec{E}_2$
- Berechnung (Multilateration)
- Übertragung der ermittelten Positionen zur Datensenke
- Max. 6 Übertragungen
- Dezentralisierter Algorithmus

Dienste

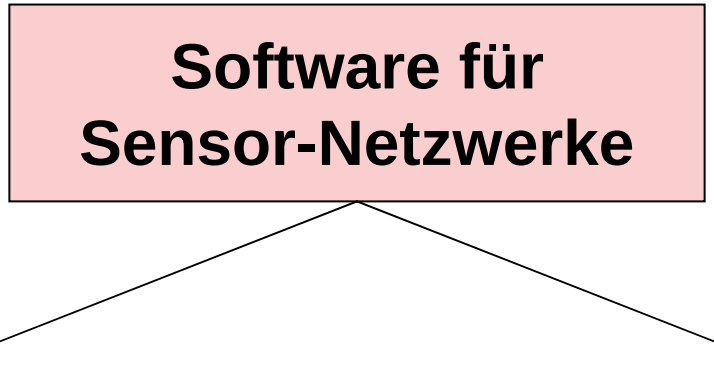
Definition: Programm, das über standardisierte Schnittstellen vom Netzwerk aus gesteuert wird.

- Suche nach Diensten
- Kaskadieren von Diensten ohne vorherige Kenntnis

Service-Infrastruktur für entfernten Zugriff:



Software-Aspekte



Programmier-Aspekt

- Systemweite API
- Hardware-Abstraktion
- Verbergen der Heterogenität des verteilten Systems
- Programmoptimierungen

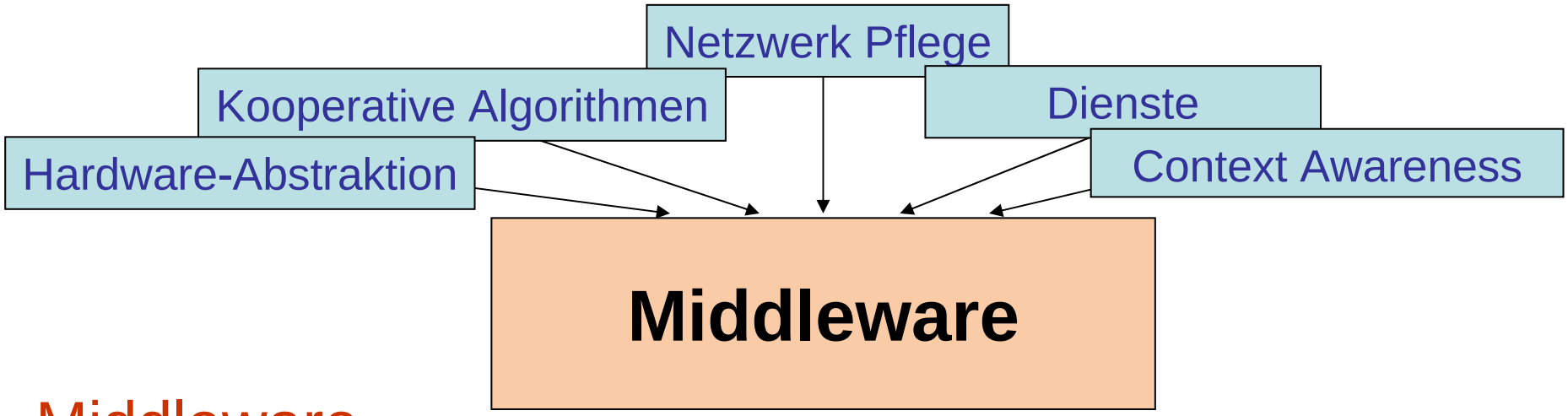
Pre-Deployment Phase

Verhaltens-Aspekt

- Zugriff auf entfernte Ressourcen ohne vorherige Kenntnis
- Anpassung der Anwendung an Umgebungsänderungen
- Task-Änderungen
- Evolution des Netzwerkes

Post-Deployment Phase

Software Engineering



Middleware:

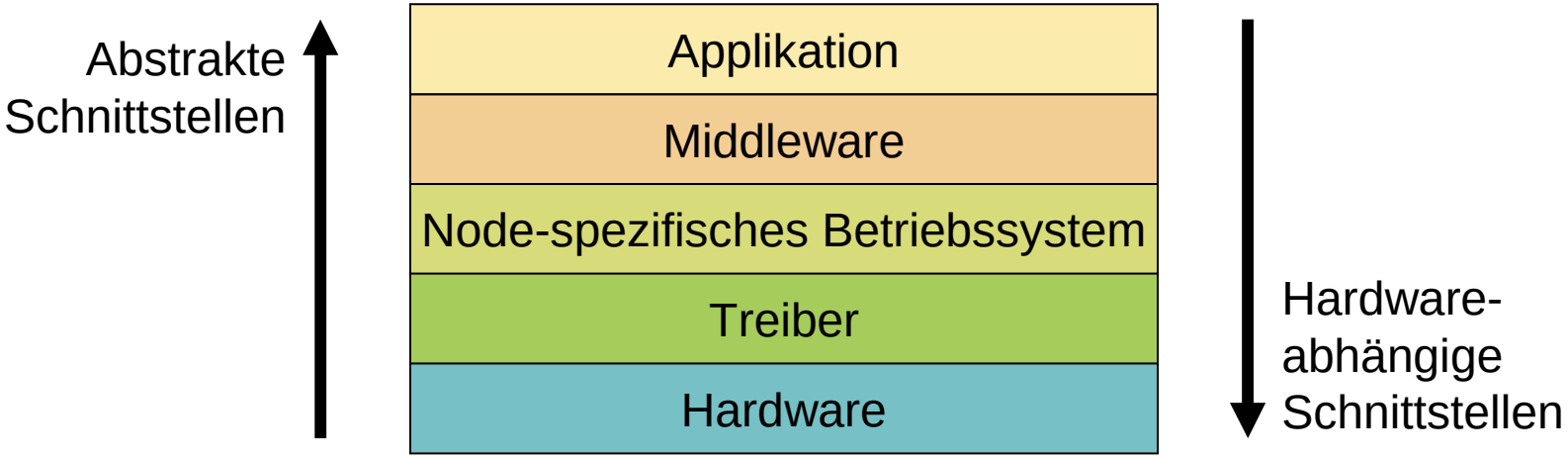
- Kapselung der Komplexität eines verteilten Systems
- Standardisierte Schnittstellen zur Knoten-Anwendung
- Remote-Access durch Dienste
- Anpassung der Software an dynamische Systemänderungen

Ziel: Ressourcen-optimierte service-orientierte Middleware für verschiedene Plattformen.

Aktuelle Arbeiten

Teilbereich	Aktivität
Middleware	• Architektur für Middleware (SeNeTs) • Schnittstellenoptimierung • Positionsbestimmung in TinyOS
Positionierung	Untersuchung und Verbesserung von Algorithmen • Trilateration (einfach, iterativ und kollaborativ) • Coarse Grained • APIT • Hop Terrain Mathematische Verfahren (Kalman, MMSE)
Routing	• Algorithmus „Positionsbasiertes Routing“
Referenzinfrastruktur	• Emulation von Sensor-Netzwerken

Software-Architektur



Eigenschaften von Schnittstellen

+ Hardwareunabhängig
+ Austauschbare Komponenten
- Overhead durch Anpassungen
- Höhere Laufzeit & höherer Energieverbrauch

Werden alle diese Eigenschaften in Sensor-Netzwerken benötigt?

Eigenschaften von Schnittstellen

Eigenschaft	Software-Schicht	
	Treiber, OS, Middleware	Dienste, Applikation
Hardwareunabhängigkeit		
Austauschbare Komponenten		
Programmgröße		
Laufzeit		
Energieverbrauch		

Bedeutung der Eigenschaften richtet sich nach Lage im Schichtenmodell.

Schnittstellen-Optimierung

Ansatz in Sensornetzwerken: Schnittstellen-Optimierung

+ Programmgröße
+ Geschwindigkeit
+ Energieverbrauch
- Austauschbarkeit
- Plattformunabhängigkeit

Generische Schnittstelle

```
void setBaudrate(int handle, int baudrate)
{
  hardware_addr=getIOAddress(handle);
  hardware_addr->BTR0=baudrate;
}
```

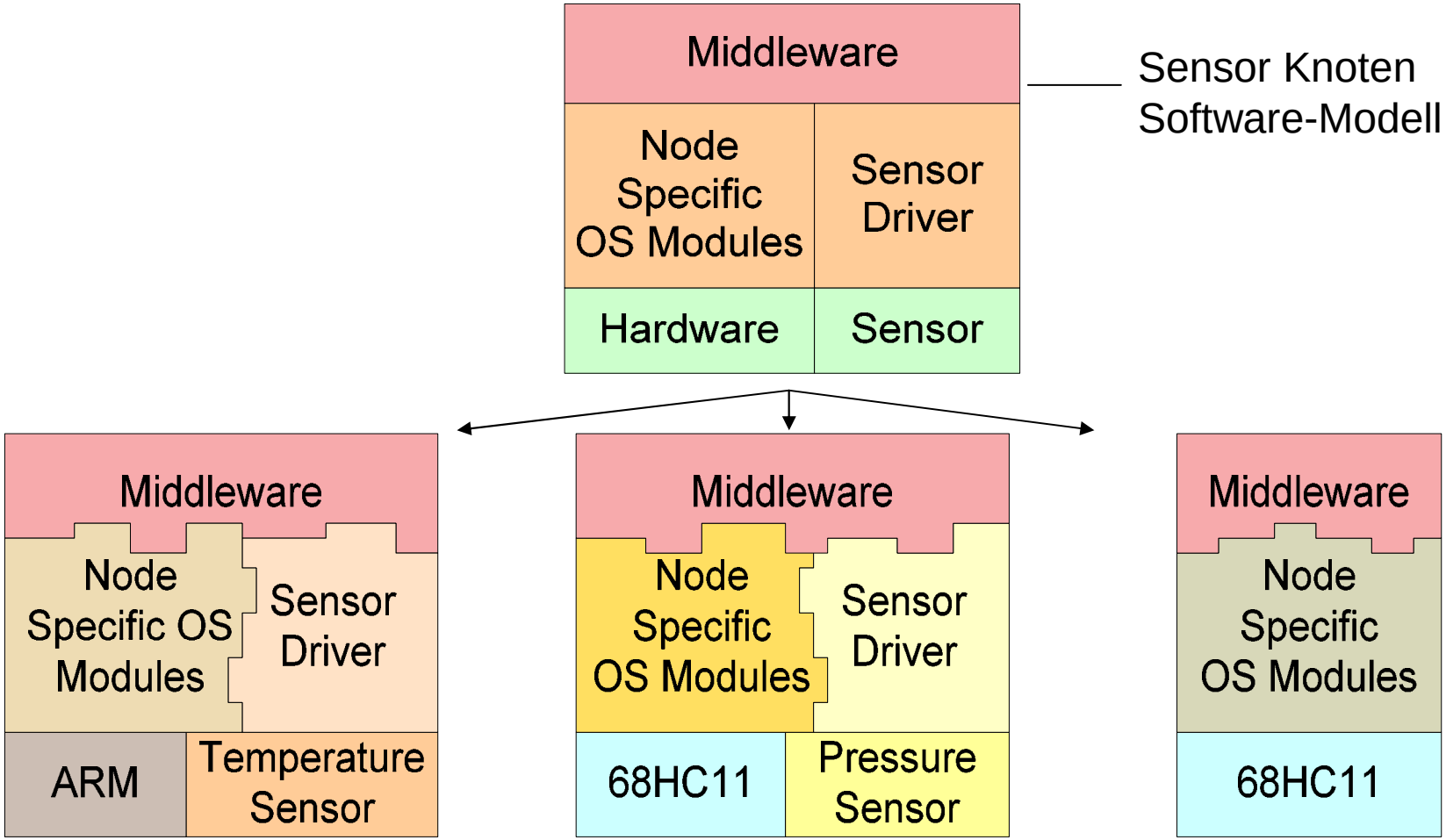
Optimierte Schnittstelle

```
void setBaudrate(int baudrate)
{
  // getIOAddress(handle);
  BTR0=baudrate;
}
```

Nur gültig bei Nutzung eines IO-Geräts

Einsparungen: Parameterübergabe, Funktionsaufruf, Stackoperation, Rückgabewert + Zuweisung, Feld-Operation

Knoten-Anwendungen



- Änderungen der Schnittstellen an Bedürfnisse des Programms
- Proprietäre Schnittstellen innerhalb eines Knotens

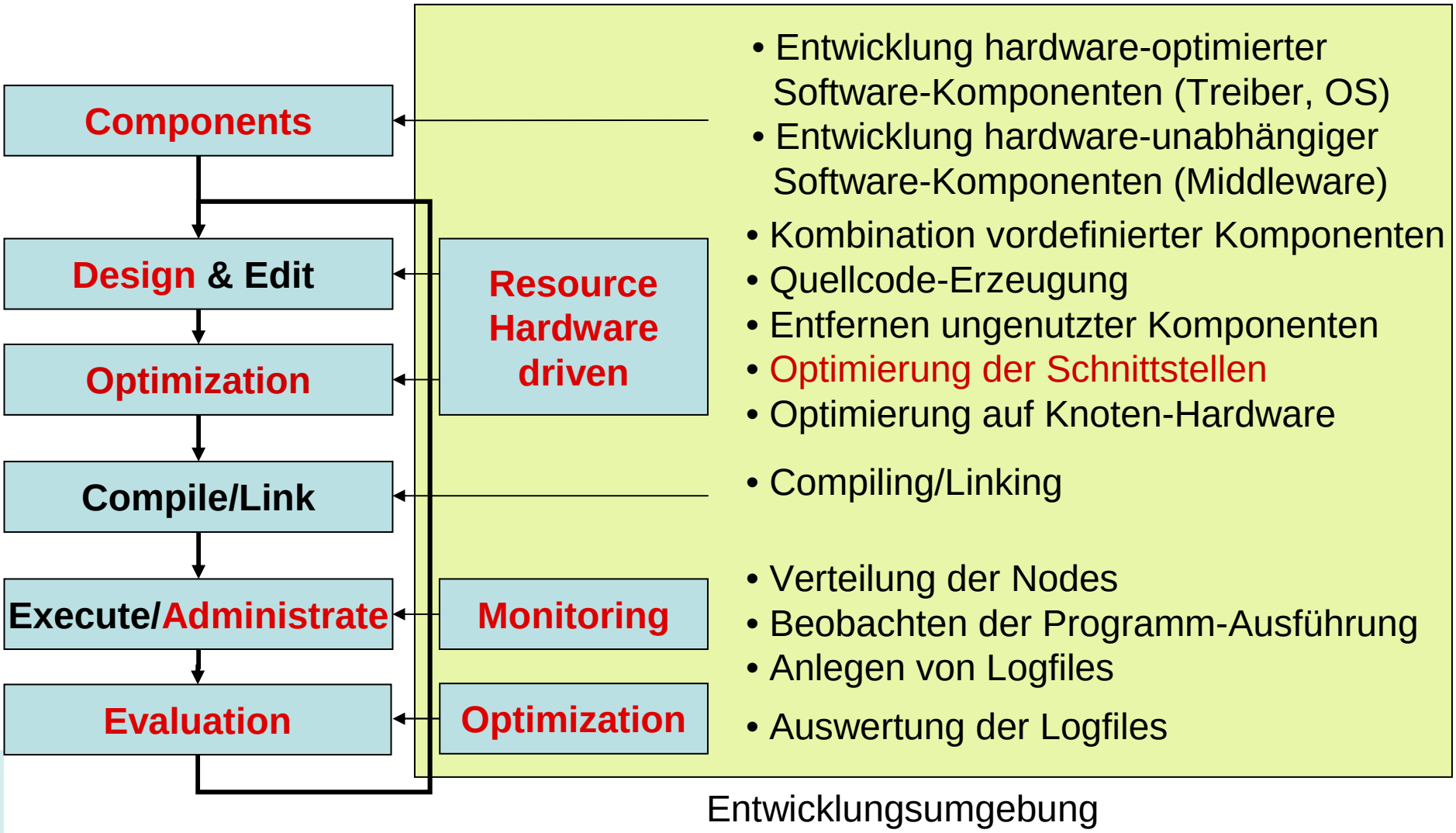
➢ Entwicklung eines Precompilers notwendig

Optimierungskriterien

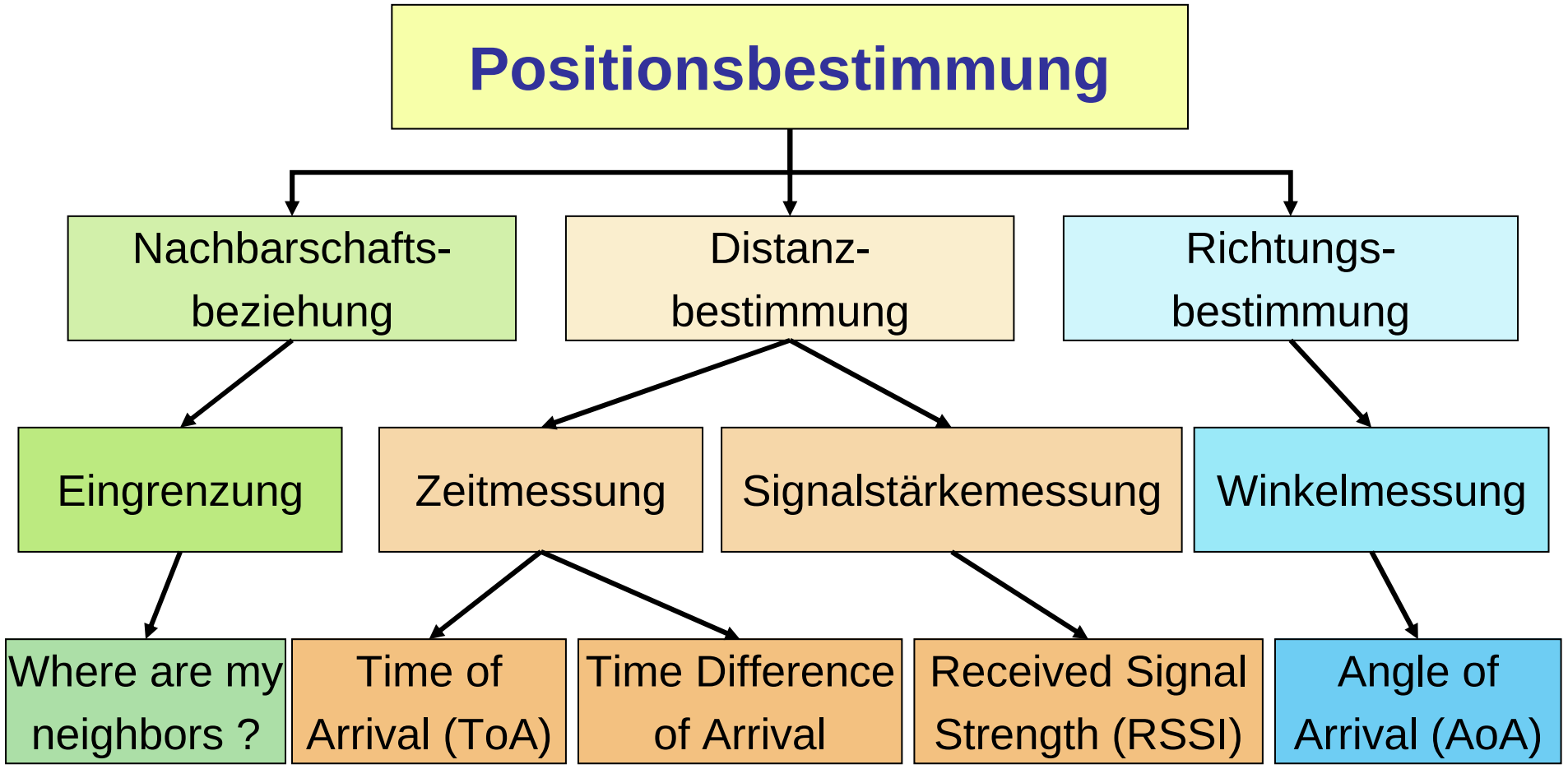
Optimierungskriterium	Beschreibung
Parameter-Eliminierung	• Entfernen unbenutzter Parameter aus dem Funktionskopf
Statische Parameter	• Entfernen von Funktionsaufrufen mit konstanten Parametern • Nutzen von statischen Variablen und Konstanten
Parameter-Anordnung	• Ändern der Übergabereihenfolge von Parametern • Vorteilhaft bei Kaskadierung von Funktionen
Parameter-Zusammenfassung	• Kombinieren von Parametern

Kenntnisse des kompletten Programmablaufs notwendig
➢ Entwicklung eines Precompilers erforderlich

Sensor Netzwerk Design



Verfahren zur Positionsbestimmung

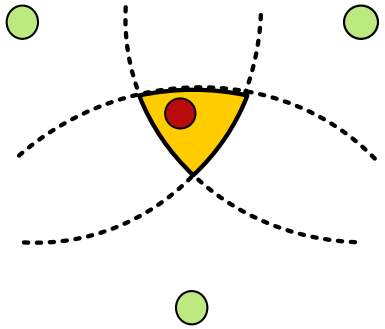


Positionierungsalgorithmen

Positionsbestimmung

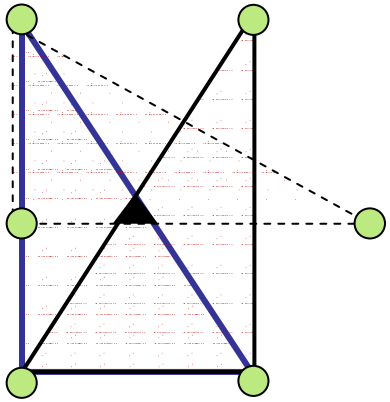
Grobkörnig, Δ>5%

Feinkörnig, Δ<5%

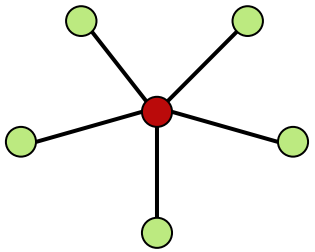


$$X_{Est} = \left(\frac{X_{i1} + \ldots X_{ik}}{k}\right)$$

Mittelpunktbildung
(Coarse Grained)



Dreiecksbildung
(APIT)



$$\Delta_i(x_0, y_0, d_i) = d_i - \sqrt{(x_i - x_0)^2 - (y_i - y_0)^2}$$

$$\underline{b} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}$$

Multilateration
(Atomar, iterativ, kollaborativ)

Δ : Fehler in der Berechnung der Position : Beacon : Einfacher Knoten

Zusammenfassung

- Herausforderungen bei Software-Entwicklung für SN
  - Dienste & Netzwerkarchitekturen
  - Kooperative Datenerfassung
- Softwareentwicklung
  - Optimierung Schnittstellen-Overhead
- Positionierung
  - Grobkörnige und feinkörnige Algorithmen

Vielen Dank