

# Datenbankmanagementsysteme in Sensornetzwerken

René Mäkeler  
17.12.2009, Magdeburg

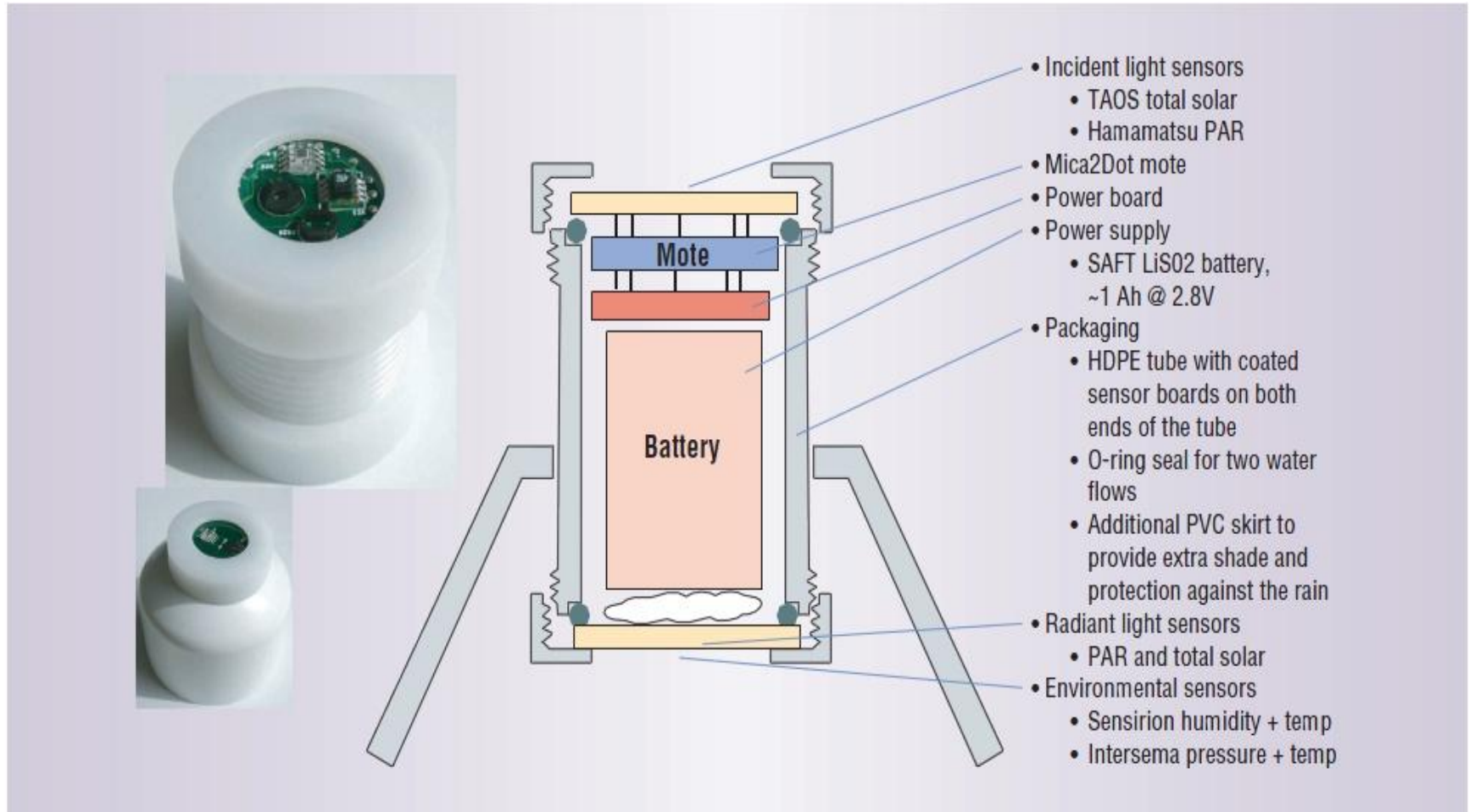
# Gliederung

- ▶ Einleitung
- ▶ Motivation
- ▶ TinyOS & TinyDB
- ▶ Datenströme – Motivation
- ▶ Optimierung kontinuierlicher Anfragen
- ▶ Zusammenfassung & Ausblick
- ▶ Quellen

# Einleitende Worte

- ▶ Moore's Law (1965/75)
- ▶ Vorhandene Technik jährlich günstiger und kleiner
  - Einsatz im physikalischen Bereich

# Was ist ein Sensornetzwerk?



[Overview of Sensor Networks; D. Culler, D. Estrin, M. Srivastava]

# Was ist ein Sensornetzwerk?

- ▶ Mehrzahl von Sensorknoten bilden ein Sensornetzwerk
- ▶ Sensorknoten bestehend aus:
  - Sensor
  - AD-Wandler
  - Mikroprozessor
  - Datenspeicher
  - Daten-Empfänger

# Motivation – Einsatz von Sensornetzwerken

- ▶ Weites Gebiet von Anwendungsmöglichkeiten:
  - Überwachung von Raum
    - Umwelt- und Lebensraumsüberwachung
    - Präzise Landwirtschaft
  - Überwachung von „Dingen“
    - Strukturelle Überwachung
    - Ökophysiologie
  - Überwachungen der Interaktion zwischen „Dingen“ miteinander oder dem Raum

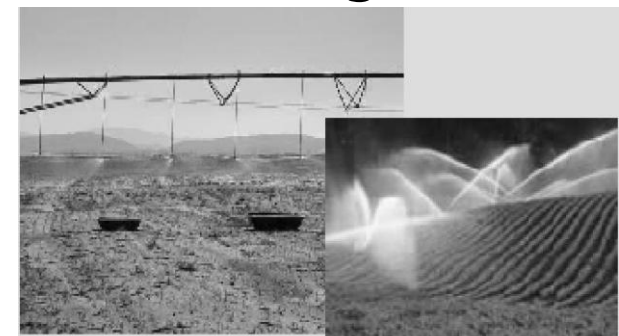


[Sensor Networks: An Overview; P. Bonnet]



A Bee Tracker

[Sensor Networks and Related Research Issues; R.Katogiri]



Precision Agriculture, Water quality management

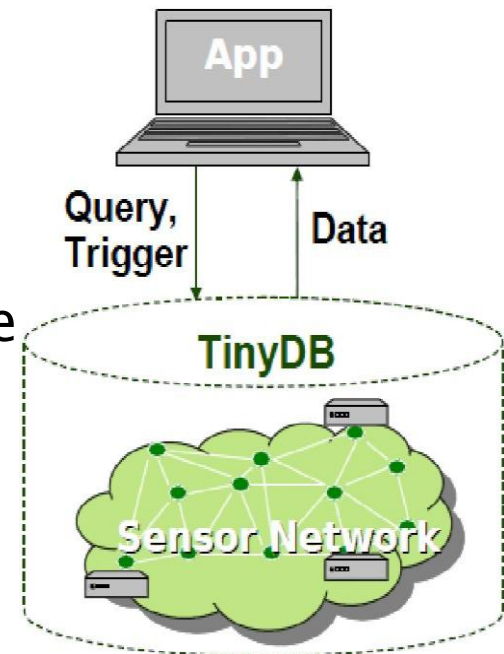
[Sensor Networks: An Overview; P. Bonnet]

# TinyOS

- ▶ Grundstruktur für anwendungsspezifische Systeme
  - Verwaltung einer Vielzahl gleichzeitiger Zugriffe trotz begrenzter Ressourcen
- ▶ Unterstützung ereignisgestützter Funktionalität
- ▶ Prinzip: Jede Anwendung enthält nur Komponenten, die sie benötigt
  - Bsp.: ReadOnly-Sensoren

# TinyDB

- ▶ Query-Processing System zum Extrahieren von Informationen
- ▶ Voraussetzung: Sensoren mit TinyOS ausgestattet
- ▶ Idee:
  - Hohe Abstraktionsebene
  - Datenzentrierte Programmierung
  - Deklarative, SQL-ähnliche Anfragesprache (→ TinySQL)
- ▶ Interface als Benutzerschnittstelle



# TinyDB

## ▶ Hintergrund:

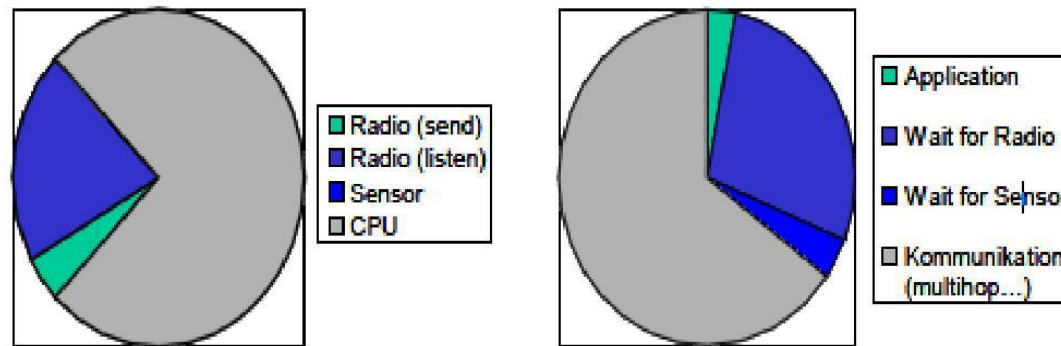


Abb. a: Energiekonsumenten

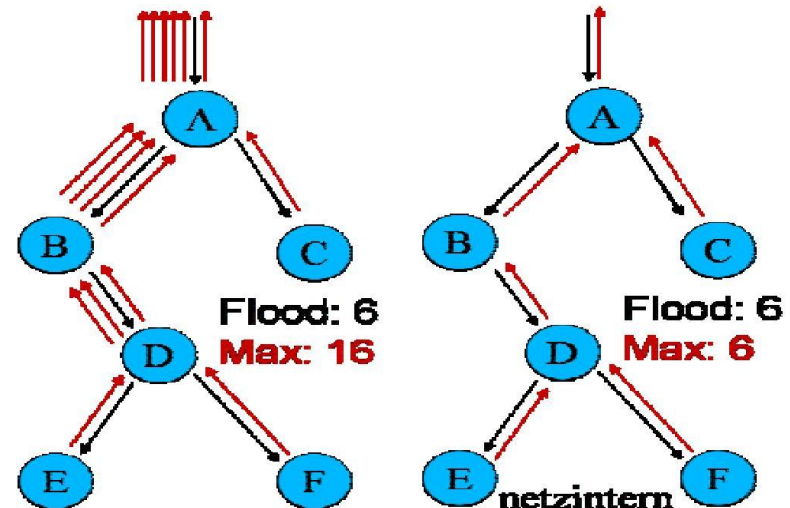
Abb. b: CPU-Zyklen-Verteilung

[Datenverarbeitung und Aggregation in Sensornetzen; C. Gebhard]

- Ca. 75% Energieverbrauch durch CPU
- Ca. 97% der CPU-Energie für die Funk-Kommunikation
  - ➔ 94% der gesamten Energie direkt und indirekt für Kommunikation verbraucht

# TinyDB

- ▶ Minimierung des Energieverbrauchs:
  - Idee: Datenaggregation
- ▶ Tiny-Aggregation-Service (TAg)
  - Übernimmt effiziente Aggregation von Daten



[Datenverarbeitung und Aggregation in Sensornetzen; C. Gebhard]

# Weitere DBMS in Sensornetzwerken

- ▶ **CougarDBMS**
  - Basis: lose gekoppeltes, verteiltes Netzwerk aus Sensoren
  - Objektorientierter Modellierungsansatz
  - SQL-ähnliches Interface
- ▶ **CometDBMS**
  - Sammlung von Werkzeugen zur Generierung von Echtzeit-DBMS
  - Datenmanagement-, Analyse- und Konfigurations-Programme
- ▶ ...

# Datenströme – Motivation

- ▶ Daten werden nicht nur gemessen, sondern kontinuierlich überwacht
  - ▶ Sensornetzwerke erfassen zeitliche Variation der Eigenschaften von Objekten oder Prozessen
    - ➔ Entstehung kontinuierlicher Folge von Daten
      - ➔ Datenstrom
  - ▶ Verarbeitung dieser Daten im krassen Gegensatz zur herkömmlichen Anfrageverarbeitung:
    - Erforderliche Speicherung der Daten
    - Keine einmalige Anfrage, stattdessen wiederholte und zeitnahe Auswertung der Anfragen
    - Meist nur einmaliger Versand der Daten
- ➔ DBMS eignen sich nur sehr eingeschränkt

# Optimierung kontinuierlicher Anfragen

- ▶ Kombination aus langlebigen Anfragen und volatilem Datenstromcharakteristika
  - Erfordert sowohl statische als auch wiederkehrende Anfrageoptimierung
- ▶ Probleme der Optimierung:
  - Fehlende Informationen über die Daten
  - Dynamische Bedingungen während der Anfrageoptimierung bzw. -ausführung
- ▶ Lebensdauer kontinuierlicher Anfragen sehr lang
  - Anwendung einer anfragenübergreifenden Optimierung

# Anfrageoptimierung in häufiggenutzten DSMS

- ▶ **STREAM (Stanford Stream Data Manager)**
  - Definition kontinuierlicher Anfragen mit Hilfe der SQL-Erweiterung CQL
- ▶ **Aurora sowie verteilte Erweiterung Borealis**
  - Anfragen nicht mittels eigener Anfragesprache, sondern über gerichtete Datenflussgraphen spezifiziert
- ▶ **TelegraphCQ**
  - Erzeugt keine festen Anfragepläne, sondern eine Reihe von Anfragemodulen
    - Verwaltung dieser Module durch sogenannte „eddies“

# Optimierung kontinuierlicher Anfragen

- ▶ Kontinuierliche Anfrage = potentiell unbegrenzte Lebensdauer
  - Optimierung einer Anfrage hat nicht zwingend über die gesamte Lebensdauer Bestand
- ▶ Reoptimierung durch:
  - gravierende Änderungen in den Eigenschaften der Datenquellen oder
  - Ausführung neuer Anfragen zu völlig neuen Bedingungen
    - Falls dies häufig der Fall ist kommt es zur *kontinuierlichen Anfrageoptimierung*

# Optimierung kontinuierlicher Anfrage

- ▶ Probleme der Reoptimierungen:
  - Ersetzen der alten Anfrage durch die optimierte
    - Abschalten = ggf. Datenverlust
    - Parallele Ausführung erfordert genaue Synchronisation
- ▶ Falls Kriterien für eine Reoptimierung erfüllt, so wird der alte Plan durch einen optimierten ersetzt
- ▶ Die Wiederverwendung historischer Anfragepläne reduziert den Reoptimierungsaufwand

# Zusammenfassung & Ausblick

- ▶ Schon heute eine Vielzahl von Anwendungsgebieten für Sensornetzwerke
- ▶ Massenproduktion intelligenter Sensoren sowie zunehmende Miniaturisierung eröffnet weitere Anwendungsfelder
- ▶ (Re-)Optimierung besonders in DSMS von Bedeutung

# Vielen Dank!

## Quellen:

1. [Overview of Sensor Networks; D. Culler, D. Estrin, M. Srivastava]
2. [Sensor Networks: An Overview; P. Bonnet]
3. [Sensor Networks and Related Research Issues; R.Katogiri]
4. [Software ubiquitärer Systeme– Datenhaltung; O. Spinczyk]
5. [Datenverarbeitung und Aggregation in Sensornetzen; C. Gebhard]