Wavellite: A software framework for the representation of knowledge about real-world phenomena observable by a sensor network
About (me)

• PhD student at UEF in environmental technology (informatics)
  – Also collaborating with Åbo Akademi
• MSc student at UEF in environmental science
• Background in informatics (MSc UZH, 2007)
• Interned with HPL Bristol (UK) and Clark & Parsia (USA) on various projects related to the semantic web
• In Kuopio since summer 2009
• Music, sports, cooking, photography, reading
Introduction
Environmental monitoring (using sensors)

Real-world phenomena and their properties (e.g. the temperature of indoor air)
Introduction

Sensor data makes *no* sense (almost)
Introduction

... and

• There is lots of it, including in environmental science
• Typically automatically sampled, sometimes at “high frequency”
• National Ecological Observatory Network (NEON, USA)
  – Bioclimate, biodiversity, biogeochemistry, ecohydrology, infectious disease, land use change
  – ~60 observatory sites in 20 eco-climatic domains over 30 years
  – 539 ”basic calibrated data products”[1]
  – ”Billions of data points collected by thousands of sensors and hundreds of people”[2]
• Measuring vehicle-induced vibration on a road section
  – 3 vibration sensors at 2 kHz; 15 billion measurements a month
Introduction
Self-describing sensor data

By sensor
For property
(Of the feature)
Of feature
(The real-world phenomena)

+ quality, uncertainty, location, ...

When (time)
Introduction
The problem

• More sensors $\rightarrow$ more data
  – Very likely
• More data $\rightarrow$ more information/knowledge?
  – Desirable but (more) challenging
  – If we continue developing and improving
    • Processing, storage, and retrieval of sensor data
    • Acquisition of information and knowledge from sensor data
    • Integration, reuse, and sharing of data, information, and knowledge
• Currently,
  – Large gap between sensor data and abstract information/knowledge
  – Generic systems to visualize (spatiotemporal) sensor data
  – *Ad hoc* systems to make more sense of sensor data
Wavellite Overview

• Software framework for
  – Continuous, real-time, and distributed
  – Representation of knowledge (for real-world phenomena)
  – Acquired from sensor data

• Software interface to represented knowledge for
  – Users via Web application
  – Software via RESTful API
Wavellite Overview

• Manage sensor data and its metadata
  – Sensor data and the when-by-for-of (self-describing sensor data)
  – In other words, sensor data and information on
    • when is the sampling time
    • the sensor by which sampling occurs
    • the property for which measurement is performed
    • and the feature of which the property is measured
  – For instance, data sampled now by this beta attenuation monitor for the concentration of outdoor PM$_{2.5}$

• Represent knowledge for real-world phenomena observed by sensors
  – Events, episodes, scenes, changes: more generally situations
  – For instance, events of unhealthy exposure to PM$_{2.5}$
Wavellite Architecture
Wavellite Implementation

• Wavellite builds on Storm for distributed processing
• Storm is a distributed real-time computation system\(^3\)
• Storm topology
  – Consists of nodes and streams
  – Nodes can implement “arbitrarily” complex logic
  – Can be deployed on single machine or cluster
• By building on Storm, Wavellite
  – Components (engines and stores) are Storm nodes
  – Measurement, observation, situation streams are Storm streams
• Knowledge base is a third-party software (Stardog RDF database)
• Front end is implemented in Spring Framework
Wavellite
Measurement layer

• Measurement engine
  – Software wrapper for (one or more) sensors
    • More generally, a source of data (e.g. sensing device, databases, files, …)
    • Very heterogeneous
  – Responsible for the retrieval of sensor data
  – Parsing may be necessary
  – Generates measurement(s) forwarded via stream(s)
  – Support relevant communication protocol(s) and data format(s)
  – Domain specific
Wavellite
Observation layer

• Observation engine
  – Subscribes to measurement stream(s)
  – Semantic enrichment of measurement data
    • Add the when-by-for-of to sensor data
    • Turn measurements into observations
  – Forward observations via stream(s)
  – Uses domain terminology defined in the knowledge base
    • Describe your sensors, features, properties, locations (types)
  – Domain terminology extends from the SSN ontology
    • Generic/reusable terminology for sensor networks and data

• Observation store
  – Subscribes to Observation stream(s)
  – Takes care to store observations to knowledge base
Wavellite
Situation layer

• Situation engine
  – Subscribes to Observation stream(s)
  – Represents situations acquired from observations
  – May use DSP, ML, CEP, … for acquisition
  – Uses domain terminology defined in the knowledge base
    • Describe your situation types, relevant objects in situations, …
  – Domain terminology extends from the STO
    • Generic/reusable terminology for situations
    • Based on Situation Theory[^4]

• Situation store
  – Subscribes to Situations stream(s)
  – Takes care to store situations to knowledge base
Wavellite Modules

• Provide services to Wavellite components, for instance
  – FFT of signal observed over time
  – Pattern classification
  – Storage to knowledge base
  – Query from knowledge base

• Implement logic to acquire knowledge
  – Learning or reasoning
  – Typically domain specific
  – Implementation in Java (possibly also JVM script languages)

• Typically software libraries
Example BeijingAir

• Sensor data acquired from @beijingair Twitter stream
• U.S. Embassy in Beijing runs a BAM for PM2.5 monitoring
• One status update every hour, e.g.

  - Statuses -> Measurement -> Observations -> Situations
  - Situations of unhealthy exposure (> 65.5 μg/m³; 24-hour)
Example
BeijingAir (Observation)
Example

BeijingAir (Observation metadata)
Example
BeijingAir (Observation data)
Example
BeijingAir (Situation)
Wavellite
Demo

• Use Web application to visualize
  – Observations by the BAM operated by the U.S. Embassy to Beijing
  – Visualize situations of (unhealthy) exposure

• Show RESTful API
Future work

• Continue developing Wavellite functionality
  – Core, Web application, RESTful API
  – For instance
    • Summary statistics view for queried observations
    • Support for KB-driven knowledge acquisition tasks

• Apply Wavellite to new domains
  – Environmental science
  – Smart grids
    – Monitoring of industrial machinery

• Write a PhD thesis
Related work

- Terminologies to describe sensor networks and sensor data\textsuperscript{[5]}
- Utilize SSN ontology\textsuperscript{[6]} or STO\textsuperscript{[7,8,9]}
  - But not both in a hybrid manner
- Extract from sensor data physiological properties of athletes\textsuperscript{[10]}
  - Representation using XML, not expressive ontology language
- Generic architecture to extract information from sensor network\textsuperscript{[11]}
  - Three-layered architecture
  - Bridging sub symbolic layer (measurement) with symbolic layer
  - Via conceptual layer that implements a metric space (similarity)
- System that can be queried for high-level events\textsuperscript{[12,13]}
  - Does not require handling of sensor data
- Ontology-based environmental information systems\textsuperscript{[14,15]}
Conclusions

• More and more sensors; more and more sensor data
• Are we keeping up with knowledge acquisition?
• Concerns
  – Large gap between sensor data and abstract knowledge
  – Complex knowledge acquisition tasks
  – Automated representation of acquired knowledge
• Wavellite aims at tackling some of these concerns
• Generic framework extensible with domain specific
  – Software that implements sensor data and knowledge acquisition
  – Modelling of sensors, observations, situations
References

Appendix

Situation Theory

• A situation, \( s \), supports (one or more) infon(s) \( \sigma \)
  – Formally, \( s \models \sigma \)

• An infon is a tuple \( \langle \langle R, a_i \ldots a_m, 0/1 \rangle \rangle \) where
  – \( R \) is a relation
  – \( a_i \ldots a_m \) are objects appropriate to the relation \( R \)
  – And 0/1 is the polarity, whereby 1 means that \( a_i \ldots a_m \) “stand in” \( R \)

• Example
  – Situation of “unhealthy exposure to PM\(_{2.5}\) in Kuopio over two days”
  – \( s \models \langle \langle \text{unhealthy-exposure, PM}_{2.5}, \text{Kuopio}, 2013-04-15, 2013-04-16, 1 \rangle \rangle \)

• Developed by Barwise and Perry, extended by Devlin
• Relates to Situational Awareness by Endsley
Appendix
Ontology

• An ontology formally represents knowledge as a set of concepts within a domain, and the relationships between pairs of concepts [Wikipedia: Ontology (information science)]

• A document that describes concepts and relationships of a domain; such documents are “written” by means of a (ontology) language

• An ontology is external to, and shared among, software systems

• Software systems commit to one (or more) ontology by adopting the defined terminology and semantics

• Key technology in knowledge representation and reasoning

• Developed within artificial intelligence research

• Used in many domains, e.g. bioinformatics (see Gene Ontology)
Appendix
Semantic Sensor Network (SSN) ontology

• Vocabulary for the representation of knowledge for sensors, properties, features, observations, ...
• Adopted at the Wavellite observation layer
• Domain-specific sensors and measured properties of features are accommodated by extending from SSN ontology
• Examples
  – BetaAttenuationMonitor subClassOf ssn:SensingDevice
  – beijingAirBAM isA BetaAttenuationMonitor
  – pm25 isA ssn:FeatureOfInterest
  – concentration isA ssn:Property
Appendix
Situation Theory Ontology (STO)

• Vocabulary for the representation of knowledge for situations
• Borrows from Situation Theory
• Adopted at the Wavellite situation layer
• Domain-specific situations and relations are accommodated by extending from STO
• Example
  – AirPollutionEvents subclassof sto:Situation
  – unhealthy-exposure isa sto:Relation