Verarbeitung und spatio-temporale Anfragen von mobilen Sensordaten auf Basis des OGC-Sensorbeobachtungsdienstes SOS 2.0

THOMAS BRINKHOFF[1] & JONAS TOLZIN[2]

Zusammenfassung: Eine wichtige Anforderung für die effektive Nutzung und Analyse von Sensordaten ist deren Bereitstellung über standardisierte Webdienste. Für den Bereich der raumbasierten Sensoren und Sensornetze hat das OGC eine Architektur mit entsprechenden Geodiensten erarbeitet, die verbreitet Akzeptanz gefunden hat. Hierbei ist insbesondere der „Sensor Observation Service" (SOS) als Geodienst zur Abfrage von Sensorbeschreibungen und Messwerten zur nennen. Für die Analyse von Sensormesswerten sind neben den eigentlichen Messwerten deren Raum- und Zeitbezug von Bedeutung. Die Kombination von Raum und Zeit ist von speziellem Interesse, wenn die Sensoren ihre Position verändern können, so dass „bewegte Objekte" beobachtet werden können. In diesem Beitrag wird eine Arbeit vorgestellt, die die Verarbeitung von mobilen Sensordaten auf Basis des Geodienstes SOS 2.0 erlaubt. Hierfür wurde eine Umgebung aufgebaut, mit der der Dienst mit simulierten Positionsangaben und mit realen bewegten Objekten erprobt werden kann. Eine Anzeige der Anfrageergebnisse wird über einen grafischen Client ermöglicht.

1 Einleitung

„Die Kenntnis über den Zustand und die langfristige Entwicklung von Natur und Umwelt ist die Grundlage für viele wirtschaftliche und umweltpolitische Entscheidungen. Veränderungen können nur durch eine intensive Umweltbeobachtung (…) nachgewiesen werden."[3] Dieser Bedarf wird durch die aktuelle technische Entwicklung unterstützt: Es wird zunehmend möglich, Sensoren, Prozessoren, Kommunikationseinrichtungen und autarke Energiequellen preisgünstig und kompakt auf sogenannten Sensorknoten zu integrieren. Die Aufgabe eines Sensorknotens besteht aus dem Auslesen der Messgrößen der Sensoren, deren Vorverarbeitung und der Weiterleitung von Sensordaten (MATTERN & RÖMER, 2003).

Für die Analyse von Sensormesswerten sind (neben dem eigentlichen Messwert) insbesondere deren Raum- und der Zeitbezug von höchster Bedeutung: Räumliche Nähe ist ein essenzieller Faktor, um Korrelationen zwischen verschiedenen Messungen herstellen zu können; das Erkennen von zeitlichen Entwicklungen ist Grundvoraussetzung für Prognosen und eventuelle Alarmierungsaufgaben. Die Kombination von Raum und Zeit ist bereits im Normalfall, aber noch mehr von Interesse, wenn die Sensoren ihre Position (z.B. angebracht auf Fahrzeugen oder

[1] Thomas Brinkhoff, Jade Hochschule Oldenburg, Institut für Angewandte Photogrammetrie und Geoinformatik, Ofener Str. 16/19, 26121 Oldenburg, E-Mail: thomas.brinkhoff@jade-hs.de
[2] Jonas Tolzin, Jade Hochschule Oldenburg, Institut für Angewandte Photogrammetrie und Geoinformatik, Ofener Str. 16/19, 26121 Oldenburg
[3] Umweltbeobachtungskonferenz 2010: Monitoring im Bereich Umwelt und Biodiversität, http://www.lanuv.nrw.de/umwelt/ubk/start.htm

Tieren) verändern können. Hierüber lassen sich dann „bewegte Objekte“ beobachten. Mit ISO 19141 (ISO, 2008) liegt ein Datenmodell vor, das ausgehend vom Feature-Geometry-Modell (ISO 19107) sich bewegende Geoobjekte repräsentieren kann.

Eine weitere wichtige Anforderung für die effektive Nutzung und Analyse von Sensordaten ist die Bereitstellung der Sensordaten über standardisierte Schnittstellen, die über Webdienste genutzt werden können („Geosensor Web“). Für den Bereich der raumbasierten Sensornetze hat das Open Geospatial Consortium (OGC) im Rahmen seiner Initiative „Sensor Web Enablement“[4] (SWE) eine Architektur mit entsprechenden Geodiensten erarbeitet, die verbreitet Akzeptanz gefunden hat. Von besonderer Bedeutung ist hierbei der 2007 verabschiedete „Sensor Observation Service“ (SOS) zur Abfrage von Sensorbeschreibungen und Messwerten. Von BRÖRING et al. (2012) wurde eine fortentwickelte OGC-Spezifikation SOS 2.0 vorgelegt, für die inzwischen auch eine erste Implementierung von 52°North verfügbar ist[5]. Die Rückgabe von Messwerten kann über das Datenmodell „Observation and Measurements“ erfolgen, das seit 2011 als ISO-Norm 19156 (ISO, 2011) vorliegt. Sensoren werden über die „Sensor Model Language“ (SensorML) beschrieben (BOTTS & ROBIN, 2007).

Betrachtet man diese Entwicklung in ihrer Gesamtheit, so ist eine wichtige Aufgabenstellung, standardisierte Geodienste für die Abfrage von Beobachtungen über bewegte Objekte nutzbar zu machen. In diesem Beitrag werden Arbeiten vorgestellt, die dazu dienen sollen, dies auf Basis des SOS 2.0 für bewegte Objekte zu erproben. Die vorgestellten Arbeiten sollen es künftig ermöglichen, dienstbasiert spatio-temporale Anfragen auf bewegten Objekten durchzuführen.

Im folgenden Abschnitt wird betrachtet, wie bewegte Objekte aktuell mit dem SOS 2.0 verwaltet und abgefragt werden können. Abschnitt 3 stellt die Speicherung von künstlich erzeugten bewegten Objekten und von realen Beobachtungen vor. Im vierten Abschnitt wird ein grafischer Client dargestellt, der zur Visualisierung der bewegten Objekte entwickelt wurde. Der Beitrag schließt mit einer Zusammenfassung und einem Ausblick auf künftige Arbeiten.

2 SOS 2.0 für bewegte Objekte

Drei Operationen muss jeder SOS-Dienst bereitstellen (BRÖRING et al., 2012): `GetCapabilities` fragt die Diensteigenschaften, `DescribeSensor` die Sensorbeschreibungen und `GetObservation` die Beobachtungen ab. Das Einfügen von Beobachtungen kann im Fall einer transaktionalen Erweiterung über die Operation `InsertObservation` erfolgen.

Für jedes bewegte Objekt muss ein zugehöriges SensorML-Dokument erzeugt werden und dem Dienst bekannt gemacht werden. Dabei ist es zweckmäßig, die möglichen Beobachtungen eines Objektes (Position und ggf. andere Messwerte) zu gruppieren.

[4] Sensor Web Enablement DWG: http://www.opengeospatial.org/projects/groups/sensorwebdwg
[5] 52°North, Sensor Observation Service: http://52north.org/communities/sensorweb/sos/index.html

Die Bereitstellung der Messwerte kann bei der Operation `GetObservation` über Anfragebedingungen eingeschränkt werden. Dabei ist es möglich, das bewegte Objekt (über eine ID) und die gewünschte Beobachtung zu spezifizieren. Zusätzlich können Einschränkungen über Zeitpunkte und -intervalle sowie über räumliche Eigenschaften[6] mit Hilfe der entsprechenden Teile des „Filter Encodings“ (ISO, 2010) angegeben werden. Damit werden die „Timeslice Query“ und „Time-Window Query“ unterstützt, nicht aber die „Moving Query“ mit variierender Anfrageregion (ŠALTENIS et al., 2000). Man beachte, dass der SOS 2.0 keine Filterung über Messwerte anbietet. Auch können keine Bedingungen über abgeleitete spatio-temporale Größen wie Geschwindigkeit, Beschleunigung oder Bewegungsrichtung formuliert werden, wie sie zum Beispiel in (GÜTING et al., 2000) und in ISO 19141:2008 vorkommen. Derartige Filterungen müssen damit nachgelagert im Client erfolgen (vgl. Abschnitt 4).

3 Erzeugung und Speicherung bewegter Objekte

In die Datenbasis des SOS-Dienstes können sowohl künstlich erzeugte bewegte Objekte als auch reale Beobachtungen eingespielt werden.

3.1 Simulierte bewegte Objekte

Für die Erzeugung bewegter Objekte wurde der „netzwerkbasierte Generator für bewegte Objekte“[7] von BRINKHOFF (2002) eingebunden. Dieser Generator, der in Java programmiert ist, hat sich inzwischen zu einem Standardwerkzeug für die Untersuchung zeit-räumlicher Datenstrukturen entwickelt[8]. Abbildung 1 zeigt links die Benutzeroberfläche des Generators.

Der Generator lässt die Integration benutzerdefinierter `Reporter`-Klassen zu, die während des Programmablaufs alle Informationen über ID, Standort und Geschwindigkeit von bewegten Objekten erhalten. Dementsprechend wurde eine Klasse `SOSReporter` implementiert, die diese Informationen in die Datenbasis des SOS einfügen kann (vgl. Abb. 1, rechts). Die Klasse kann über ein Dialogfenster gesteuert werden. Es gibt drei Wahlmöglichkeiten bezüglich des Einfügens (siehe auch Abschnitt 3.4):

- Einfügen über die `InsertObservation`-Operation des SOS
- Einfügen über die Hibernate[9]-Schicht der SOS-Implementierung von 52°North
- Direktes Einfügen in die Datenbank über SQL

[6] Die SOS-Implementierung von 52°North unterstützt als räumliche Anfrage derzeit nur die Rechteckanfrage mittels `BBOX`.

[7] Der Generator ist verfügbar unter: http://iapg.jade-hs.de/personen/brinkhoff/generator/

[8] Eine Untersuchung von 2008 (http://iapg.jade-hs.de/personen/brinkhoff/generator/AnalysisOfUsage.pdf) zeigt auf, dass der Generator in über 100 wissenschaftlichen Veröffentlichungen für experimentelle Untersuchungen verwendet wurde.

[9] Hibernate ist ein Framework, das die Abbildung zwischen einer objektorientierten Programmiersprache und relationalen Datenbanken unterstützt und von der SOS-Implementierung von 52°North genutzt wird.

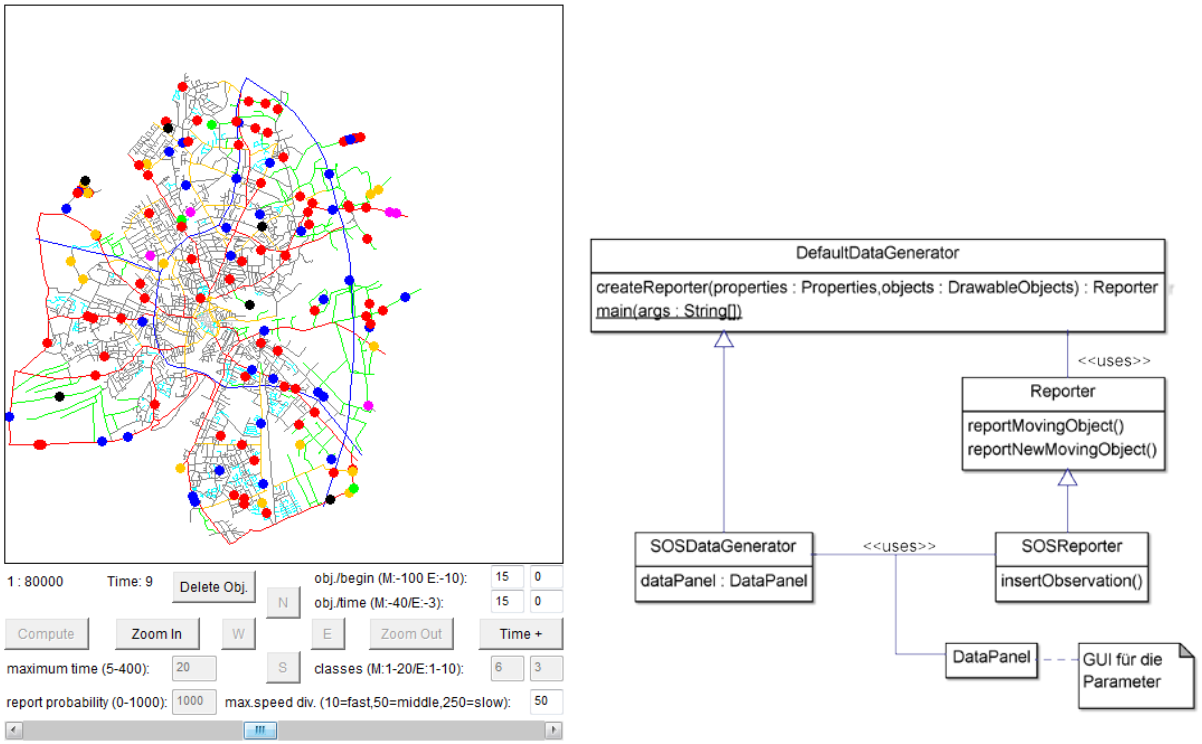

Abb. 1: Der Generator für bewegte Objekte (links) und die Einbettung der `SOSReporter`-Klasse (rechts)

3.2 Reale bewegte Objekte

Das Einfügen von realen Beobachtungen basiert auf zwei Systemen: Als bewegtes Objekt wurde der Roboter RP6 der Firma Arexx Engineering[10] verwendet (Abbildung 2). Er stellt ein Grundgerüst zur Steuerung über einen Mikrocontroller bereit, welches für eigene Zwecke durch Software angepasst werden kann. Zusätzlich werden für den RP6 vorgefertigte Hardware-Erweiterungen anboten, die den Mikrocontroller auf der Basisstation entlasten und um Funktionalität ergänzen. So wurde innerhalb dieses Projekts die WLAN-Erweiterung verwendet. Damit ist es grundsätzlich über WLAN möglich, den Roboter zu steuern sowie Daten zu senden und zu empfangen. An Sensorik stehen ein Antikollisionssystem (ACS), das mit Hilfe von Infrarotdioden realisiert ist, zwei Lichtsensoren sowie ein Drehzahlmesser zur Verfügung. Die Programmierung des Roboters erfolgt in C99. Für die Kommunikation des RP6 von und zu einem PC stehen Bibliotheken bereit. Diese Kommunikation kann über die serielle Schnittstelle oder über WLAN erfolgen.

Ein Sensornetz wurde mit Hilfe von MicaZ-Sensorboards der Firma CrossBow aufgebaut, die über ZigBee (2,4 GHz) drahtlos miteinander kommunizieren. Das Modul MDA100CB ist mit einem Temperatur- und einem Lichtsensor ausgestattet. Zudem stand ein Sensorknoten (MTS420CC/MTS400CC) mit zweiachsigem Beschleunigungssensor, Lichtsensor, barometrischem Sensor, Temperatursensor, Feuchtigkeitssensor und GPS-Empfänger zur

[10] Website RP6 Robot System: http://www.arexx.com/rp6/html/de/

Verfügung. Die MicaZ-Sensorboards (auch „Motes“ genannt) wurden auf Basis von TinyOS mit der C99-Erweiterung „nesC“ programmiert (LEVIS, 2006). Die Kommunikation mit dem PC erfolgte über das Programm „XServe“ aus dem Softwarepaket „Moteworks“ von CrossBow, welches die Datenpakete der Motes (u.a. über eine Netzwerkverbindung) abstrahiert zur Verfügung stellt.

Das MDA100CB-Board wurde für die Verbindung mit dem RP6 genutzt. Für die Kommunikation wurde die serielle Schnittstelle eingesetzt, d.h. der Sensorknoten ist direkt mit dem Roboter verbunden.

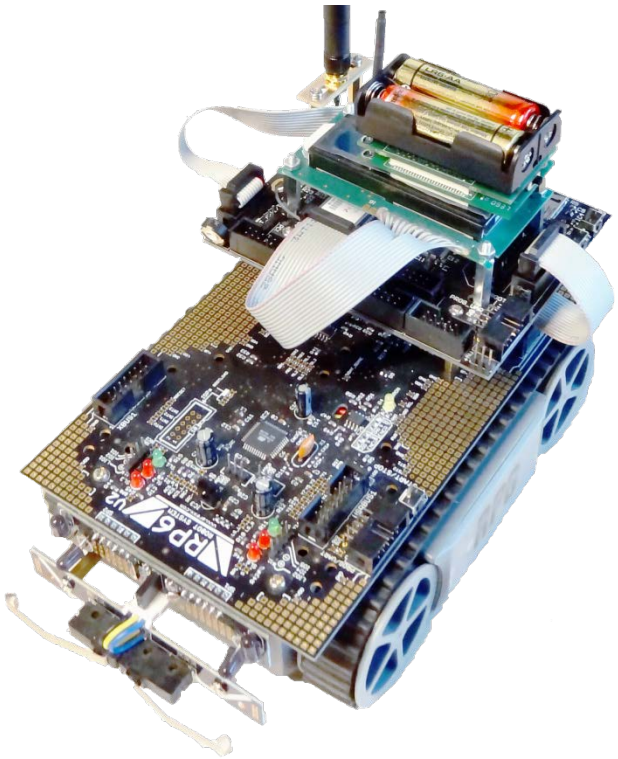

Abb. 2: Roboter Arexx RP6 mit WLAN-Aufsatz und MicaZ-Sensorboard (ganz oben)

3.3 Gesamtstruktur

Abbildung 3 stellt die Gesamtstruktur hinsichtlich Empfang und Weiterverarbeitung von Sensordaten dar. Die Benutzeroberfläche (GUI) konfiguriert die verschiedenen Software-Komponenten bzw. empfängt Statusmeldungen. Die Sensorknoten können nach erfolgter Verbindung zu jedem Zeitpunkt Daten schicken, welche bis zum vollständigen Empfang zwischengespeichert werden, um danach an die `Parser`-Komponente übergegeben zu werden. Diese verwandelt den Rohdatenstrom in eine Datenstruktur um, welche von der Komponente `InsertFactory` ohne Kenntnis der Originalstruktur verwendet werden kann. Je nach Konfiguration übernimmt diese Klasse das Anlegen der Sensoren und das Einfügen der Beobachtungen.

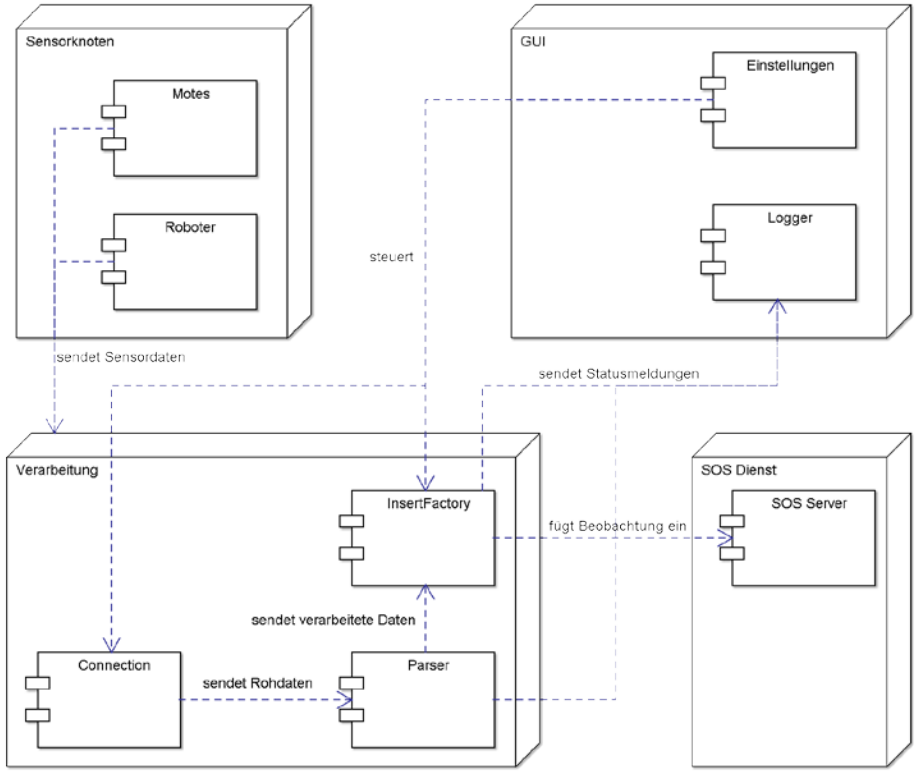

Abb. 3: Gesamtstruktur hinsichtlich Empfang und Weiterverarbeitung von Sensordaten für den SOS

3.4 Einfügen von Daten

Für die Aufnahme von Daten in die SOS-Datenbasis wurde eine Bibliothek implementiert, die das Einfügen neuer Sensoren (Klasse `SensorInserter`) und von Beobachtungen (Klasse `ObservationInserter`) unterstützt.

Neue Sensoren können über den SOS-Dienst und Hibernate eingefügt werden. Dazu wird ein XML-Dokument erzeugt, das im Wesentlichen aus einem SensorML-Dokument besteht. Mit Hilfe der Informationen, die im `SensorInserter` hinterlegt sind, kann ein `ObservationInserter`-Objekt initialisiert werden. Danach müssen nur noch die Messwerte und der ggf. neue Standpunkt hinterlegt werden. Neben dem Einfügen über den SOS-Dienst und Hibernate besteht die Möglichkeit, die Beobachtung direkt per SQL in die Datenbank einzufügen. Dieser Ansatz ist deutlich schneller als die beiden anderen Verfahren. Allerdings macht jede Änderung des internen Datenbankschemas eine Anpassung des Programmcodes der `ObservationInserter`-Klasse erforderlich.

Da uns auf dem RP6-Roboter keine Ortungssensoren zur Verfügung standen und der GPS-Empfänger auf dem MicaZ-Mote keine Indoor-Lokalisierung ermöglichte, wurde aus den empfangenen Informationen über den Kettenantrieb des Roboters per Odometrie dessen Position berechnet (TOLZIN, 2014).

4 Grafische Visualisierung

Für die grafische Visualisierung wurde (u.a.) ein Client in Java auf Basis von Java Swing entwickelt. Für die Einbettung einer OpenStreetMap-Karte (OSM) wurde der JXMapViewer[11] genutzt. Neben Kartenfunktionen unterstützt der Client die Eingabe und Weiterleitung von Auswahlkriterien. Zur Auswahl können ein Anfragefenster, zeitliche Bedingungen und thematische Attributbedingungen angegeben werden. Die Visualisierung der Positionen von bewegten Objekten erfolgt grafisch in der Karte. Zusätzlich können Messwerte über Diagramme dargestellt werden, die mit Hilfe der Java-Bibliothek JFreeChart[12] realisiert wurden.

Um abgeleitete spatio-temporale Größen, die als Attributwert in den Messdaten vorliegen[13], abfragen zu können, wurde im Client nachgelagert eine Komponente integriert, die die vom Dienst gelieferten Objekte hinsichtlich thematischer Anfragebedingungen filtern kann.

Abbildung 4 zeigt das Hauptfenster des Clients mit der Möglichkeit, Anfragebedingungen einzugeben. In Abbildung 5 wird ein Diagramm mit den Geschwindigkeiten mehrerer bewegter Objekte über einen Zeitraum dargestellt.

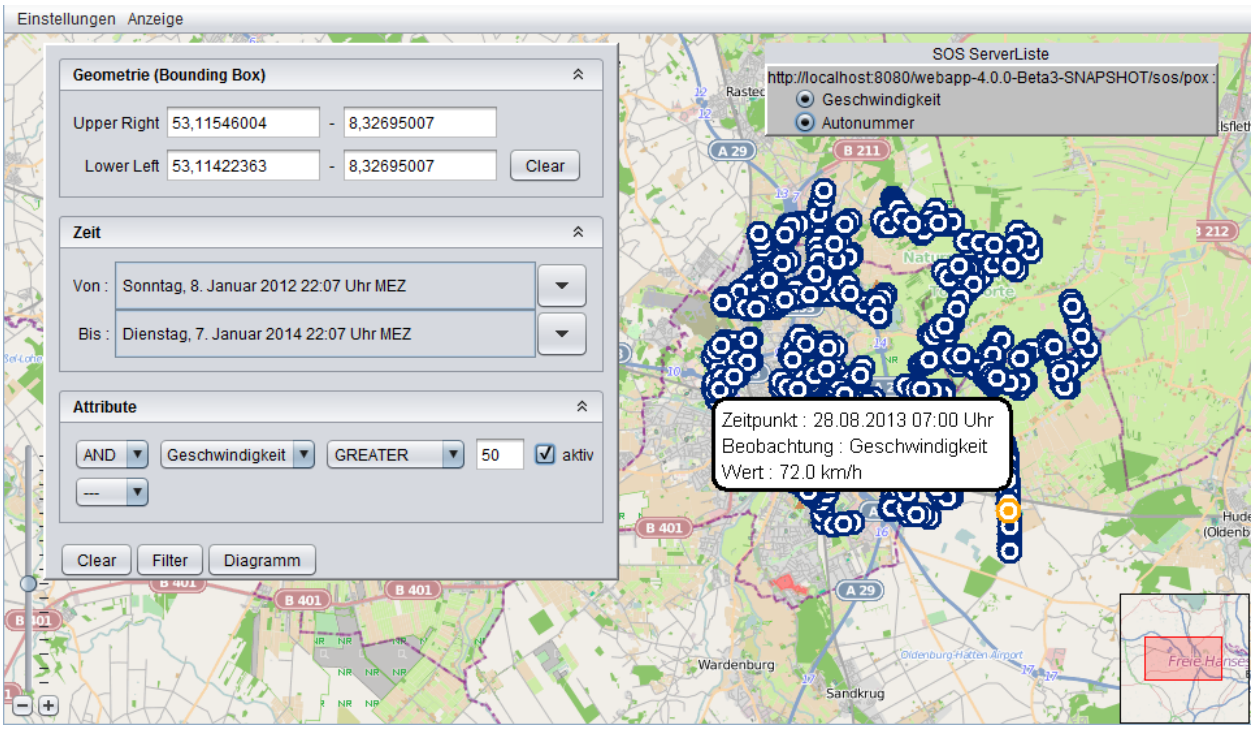

Abb. 4: Oberfläche des grafischen Clients mit Eingabe von Anfragebedingungen

[11] JXMapViewer – OpenStreetMap Wiki: http://wiki.openstreetmap.org/wiki/JXMapViewer
[12] JFreeChart Website: http://www.jfree.org/jfreechart/
[13] Bei den Daten, die durch den Generator erzeugt wurden, ist dies die Geschwindigkeit der Objekte.

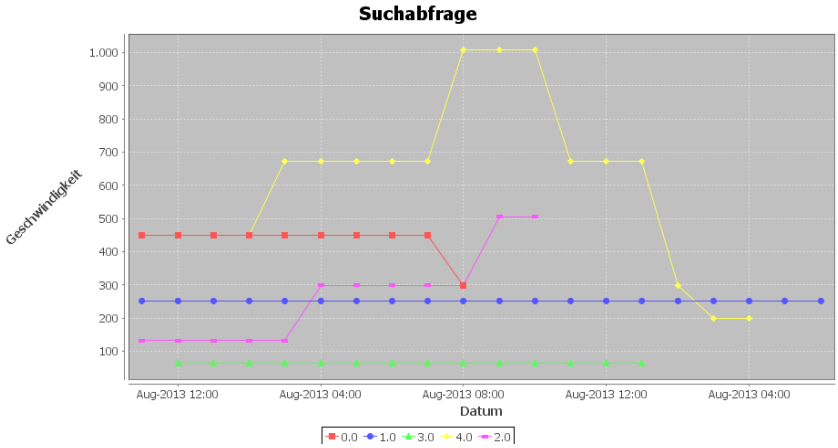

Abb. 5: Anzeige der Geschwindigkeit mehrerer bewegter Objekte als Diagramm

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde eine Arbeit vorgestellt, die die Verarbeitung von mobilen Sensordaten auf Basis des Geodienstes SOS 2.0 unter der Berücksichtigung von spatio-temporalen Anfragen ermöglicht. Dazu wurden auf Basis der Implementierung von 52°North entsprechende Erweiterungen in der Datenrepräsentation und Auswertung vorgenommen, die im Beitrag näher vorgestellt wurden. Der Ansatz wurde mit simulierten Positionsangaben und mit realen bewegten Objekten erprobt. Eine Spezifikation von Anfragebedingungen, die Anzeige der Anfrageergebnisse und eine weitere Filterung der Ergebnisse über Messwerte sind über einen grafischen Client möglich.

Zwei künftige Aufgaben leiten sich aus dem Beitrag ab: Die vorgestellte Architektur ist bezüglich ihrer quantitativen Leistungsfähigkeit zu untersuchen und zu optimieren. Außerdem ist zu betrachten, wie die Filterung nach abgeleiteten spatio-temporalen Größen wie Geschwindigkeit, Beschleunigung und Bewegungsrichtung auf die Serverseite verlagert und über Erweiterungen des SOS 2.0 unterstützt werden kann.

6 Literaturverzeichnis

BOTTS, M. & ROBIN A. (eds.), 2007: OpenGIS Sensor Model Language (SensorML) Implementation Specification, Version 1.0.0. OGC 07-000.

BRINKHOFF, T., 2002: A Framework for Generating Network-Based Moving Objects. GeoInformatica, **6** (2), 153-180.

BRÖRING, A.; STASCH, C. & ECHTERHOFF J. (eds.), 2012: OGC Sensor Observation Service Interface Standard, Version 2.0. OGC 12-006.

GÜTING, R. H.; BÖHLEN, M. H.; ERWIG, M.; JENSEN, C.; LORENTZOS, N. A.; SCHNEIDER, M. & VAZIRGIANNIS, M., 2000: A Foundation for Representing and Querying Moving Objects. ACM Transaction on Database Systems, **25**(1), 1-42.

ISO, 2008: International Standard ISO 19141:2008 – Geographic Information – Schema for Moving Objects.

ISO, 2010: International Standard ISO 19143:2010 – Geographic Information – Filter Encoding.

ISO, 2011: International Standard ISO 19156:2011 – Geographic Information – Observations and Measurements.

LEVIS, P., 2006: TinyOS Programming.

MATTERN, F. & RÖMER, K., 2003: Drahtlose Sensornetzwerke. In: GI-Informatiklexikon, http://www.gi.de/service/informatiklexikon/detailansicht/article/drahtlose-sensornetze.html

ŠALTENIS, S.; JENSEN, C.; LEUTENEGGER, S. T. & LOPEZ, M. A., 2000: Indexing the Positions of Continuously Moving Objects. Proceedings ACM SIGMOD International Conference on Management of Data, Dallas, TX, 331-342.

TOLZIN, J., 2014: Die Verarbeitung von mobilen Sensordaten auf Basis des OGC-Sensorbeobachtungsdienstes SOS 2.0 unter der besonderen Berücksichtigung von spatio-temporalen Abfragen. Bachelorarbeit, Jade Hochschule Oldenburg.