A live Link from GIS to the Internet of Things

Dustin Demuth

52north
exploring horizons

Geoinformatik 2013 - 13.03.

question

how to integrate live data from a sensor platform into GIS, without using third party services?

question

how to integrate data of sensor platforms into GIS, without using third party services?

answer

connect the sensor platform to the internet and turn it into a feature service!

internet of things

concept

- unique identification of things
- physical attributes have virtual representations
- internet protocols are used to transport the information

internet of things

concept

- unique identification of things
- physical attributes have virtual representations
- internet protocols are used to transport the information

internet of things

concept

- unique identification of things
- physical attributes have virtual representations
- internet protocols are used to transport the information

internet of things

concept

- unique identification of things
- physical attributes have virtual representations
- internet protocols are used to transport the information

summary

IoT is network of physical things and their virtual representations which use the internet protocols as transport mechanisms [1].

web of things

adds an application layer to the iot

- extends the IoT [2]
- each thing is a resource with an URI [3]
- HTTP [4] and REST [5]

web of things

adds an application layer to the iot

- extends the IoT [2]
- each thing is a resource with an URI [3]
- HTTP [4] and REST [5]

web of things

adds an application layer to the iot

- extends the IoT [2]
- each thing is a resource with an URI [3]
- HTTP [4] and REST [5]

web of things

adds an application layer to the iot

- extends the IoT [2]
- each thing is a resource with an URI [3]
- HTTP [4] and REST [5]

summary

WoT builds an application layer on top of the IoT and makes accessing things more simple, by using lightweight web standards.

concept

turn each measurement into a resource

- apply WoT paradigm to sensor platforms
  - each sensor is a resource, represented as a layer
  - each measurement is a resource, represented as a feature
- use methods which already fit into the GIS domain

http, rest

concept

turn each measurement into a resource

52north exploring horizons

- apply WoT paradigm to sensor platforms
  - each sensor is a resource, represented as a layer
  - each measurement is a resource, represented as a feature
- use methods which already fit into the GIS domain

`example.org/sensorlayer`

concept

turn each measurement into a resource

- apply WoT paradigm to sensor platforms
  - each sensor is a resource, represented as a layer
  - each measurement is a resource, represented as a feature
- use methods which already fit into the GIS domain

```
example.org/sensorlayer/measurementfeature
```

concept

turn each measurement into a resource

- apply WoT paradigm to sensor platforms
    - each sensor is a resource, represented as a layer
    - each measurement is a resource, represented as a feature
- use methods which already fit into the GIS domain

OGC compliance

requirements

for implementing the concept

- affordable, open, customizable hardware
- capable of reading various sensors
- storage
- sufficient processing power

requirements

for implementing the concept

- affordable, open, customizable hardware
- capable of reading various sensors
- storage
- sufficient processing power

requirements

for implementing the concept

- affordable, open, customizable hardware
- capable of reading various sensors
- storage
- sufficient processing power

requirements

for implementing the concept

- affordable, open, customizable hardware
- capable of reading various sensors
- storage
- sufficient processing power

requirements

for implementing the concept

- lightweight data format & interfaces → JSON & REST
- standardized interface → esri / OGC GeoServices REST API [6]

requirements

for implementing the concept

- lightweight data format & interfaces → JSON & REST
- standardized interface → esri / OGC GeoServices REST API [6]

GeoServices REST API

provides interface definitions for:

- Map Service
- Geocode Service
- Geometry Service
- Geoprocessing Service
- Image Service
- Feature Service

GeoServices REST API

provides interface definitions for:

- Map Service
- Geocode Service
- Geometry Service
- Geoprocessing Service
- Image Service
- Feature Service

GeoServices REST API

Request examples

/ `example.org/geoservices/`
service description and array of available layers

/<id>/ `example.org/geoservices/1/`
detailed information on the layer which is identified by 1

/<id>/query `example.org/geoservices/1/query`
list of features within a layer

/<id>/<oid> `example.org/geoservices/1/15328`
single feature of layer 1, identified by object id 15328

GeoServices REST API

Request examples

/ `example.org/geoservices/`
service description and array of available layers

/<id>/ `example.org/geoservices/1/`
detailed information on the layer which is identified by 1

/<id>/query `example.org/geoservices/1/query`
list of features within a layer

/<id>/<oid> `example.org/geoservices/1/15328`
single feature of layer 1, identified by object id 15328

GeoServices REST API

Request examples

/ `example.org/geoservices/`
service description and array of available layers

/<id>/ `example.org/geoservices/1/`
detailed information on the layer which is identified by 1

/<id>/query `example.org/geoservices/1/query`
list of features within a layer

/<id>/<oid> `example.org/geoservices/1/15328`
single feature of layer 1, identified by object id 15328

GeoServices REST API

Request examples

/ `example.org/geoservices/`
service description and array of available layers

/<id>/ `example.org/geoservices/1/`
detailed information on the layer which is identified by 1

/<id>/query `example.org/geoservices/1/query`
list of features within a layer

/<id>/<oid> `example.org/geoservices/1/15328`
single feature of layer 1, identified by object id 15328

RESTful GeoService API

Request examples: filtering

/<id>/query? `example.org/geoservices/1/query?geometryType=GeometryPoint&geometry=7,52`
features of layer 1 which are on point (7 , 52)

/<id>/query? `example.org/geoservices/1/query?f=json`
features of layer 1 encoded in json-format (standard)

& parameters `where`, `returnGeometry`, `inSR`, `outSR`, `spatialRel`,`relationParam`, `objectIds`, `outFields`, `returnIdsOnly`

RESTful GeoService API

Request examples: filtering

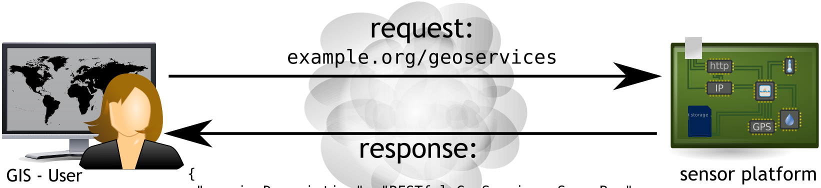
/<id>/query? `example.org/geoservices/1/query?geometryType=GeometryPoint&geometry=7,52`
features of layer 1 which are on point (7 , 52)

/<id>/query? `example.org/geoservices/1/query?f=json`
features of layer 1 encoded in json-format (standard)

& parameters `where`, `returnGeometry`, `inSR`, `outSR`, `spatialRel`,`relationParam`, `objectIds`, `outFields`, `returnIdsOnly`

but:
due to processing and memory constraints filtering is not implemented in our approach

workflow



request:
example.org/geoservices

response:

```
{
 "serviceDescription": "RESTful GeoServices SenseBox",
 "layers": [
  {
    "id": "1",
    "name": "temperature"
   }
  ]
}
```

RESTful GeoService API

..and some responses

/

```
{
  "serviceDescription": "RESTful GeoServices SenseBox",
  "layers": [
    {
    "id": "1",
    "name": "temperature"
    }
  ]
}
```

RESTful GeoService API

..and some responses

/<id>/

```
{
  "id": "1",
  "type": "Feature Layer",
  "displayField": "value",
  "capabilities": "Query",
  "geometryType": "GeometryPoint",
  "minScale": 0,
  "maxScale": 0,
  "spatialReference": {
      "wkid": 4326
  },
  "objectIdField": "objectid",
  "fields": [
    {
      "name": "objectid",
      "type": "FieldTypeOID",
      "alias": "Object ID"
    },
    <...>
  ]
}
```

RESTful GeoService API

..and some responses

/<id>/query

```
{
  "objectIdFieldName": "objectid",
  "geometryType": "GeometryPoint",
  "spatialReference": {
    "wkid": 4326
  },
  "fields": [
    {
      "name": "objectid",
      "type": "FieldTypeOID",
      "alias": "Object ID"
    },
    <...>
  ],
```

RESTful GeoService API

..and some responses

```
"features": [
{
  "geometry": {
    "point": {
      "x": 7.652118,
      "y": 51.934969
    },
    "spatialReference":
        {
      "wkid": 4326
    }
  },
```

```
  "attributes": {
    "ObjectID": "15328",
    "Time": "2013-01-08
        T14:36:03Z",
    "Value": "15"
  }
},
<...>
}]}
```

evaluation

+ low power consumption
+ simple integration of sensors into gis
+ easy customization by the users
- limited in speed
- limited in memory
- missing multithreading
o currently limited choice of clients

what comes next?

build clients

more powerful hardware

thank you

52north
exploring horizons

Dustin Demuth d.demuth@52north.org

references

[1] Erik Wilde.
Putting things to rest.
UCB iSchool Report 2007-015, 2007.

[2] Dominique Guinard, Vlad Trifa, Friedemann Mattern, and Erik Wilde.
From the internet of things to the web of things: Resource oriented architecture and best practices.
In Dieter Uckelmann, Mark Harrison, and Florian Michahelles, editors, *Architecting the Internet of Things*, chapter 5, pages 97–129. Springer, New York Dordrecht Heidelberg London, April 2011.

[3] T. Berners-Lee, R. Fielding, and L. Masinter.
Uniform Resource Identifier (URI): Generic Syntax.
RFC 3986 (Standard), January 2005.

[4] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee.
Hypertext Transfer Protocol – HTTP/1.1.
RFC 2616 (Draft Standard), June 1999.
Updated by RFCs 2817, 5785, 6266.

[5] Roy T. Fielding and Richard N. Taylor.
Principled design of the modern web architecture.
In *Proceedings of the 22nd international conference on Software engineering*, ICSE '00, pages 407–416, New York, NY, USA, 2000. ACM.

[6] ESRI.
Geoservices rest specification version 1.0.
Online, September 2010.
Whitepaper.

appendix

images

The 52° North logo is property of 52° North.
If not denoted otherwise, images are self-made or had been licensed as public domain

acknowledgements

This work has been partially supported by the project *Flexible and Efficient Integration of Sensors and Sensor Web Services* funded by the ERDF program for NRW (contract number N 114/2008), proceeding research was done during a Google Summer of Code 2012 project