A NEW PEER-TO-PEER-BASED INTEROPERABLE SPATIAL SENSOR WEB ARCHITECTURE

S.H.L. Liang

Department of Geomatics Engineering, University of Calgary, Calgary, Alberta, CANADA T2N 1N4
steve.liang@ucalgary.ca

Commission I, ThS-1

KEY WORDS: GIS, Internet/Web, Interoperability, Multisensor, Distributed

ABSTRACT:

With the rapid advances in sensor network, and information and communication technologies, the vision of a World-Wide Sensor Web (WSW) is becoming a reality. However, there is a lack of a spatial information infrastructure that could aggregate the independent geo-sensor networks into a coherent Spatial Sensor Web (SSW). The SSW vision brings two architectural challenges to today's GIService systems: (1) scalability and (2) interoperability. This paper introduces GeoSWIFT 2.0, a new scalable and interoperable SSW architecture. GeoSWIFT 2.0 is scalable – it removes all single points of failure and system performance bottlenecks by using a fully decentralized P2P spatial query framework. GeoSWIFT 2.0 is also interoperable – it integrates interoperable sensor web standards with the scalable P2P framework.

1. BACKGROUND

Distributed sensor networks are attracting more and more interest in applications for large-scale monitoring of the environment, civil structures, roadways, animal habitats, etc. With the rapidly increasing number of large-scale sensor network deployments, the vision of a World-Wide Sensor Web (WSW) is becoming a reality. Similar to the World-Wide Web (WWW), which acts essentially as a "World-Wide Computer", the Sensor Web can be considered as a "World-Wide Sensor" or a "cyberinfrastructure" that instruments and monitors the physical world at temporal and spatial scales that are currently impossible. Ranging from video camera networks that monitor real-time traffic to matchbox-sized wireless sensor networks embedded in the environment to monitor habitats, the WSW will generate tremendous volumes of priceless data, enabling scientists to observe previously unobservable phenomena.

One major reason that the development of the WSW has been greatly limited is the lack of an infrastructure that connects many heterogeneous sensor networks to the applications that desire sensor network data. Data is the raison d'être of any sensing exercise. However, today's WSW researchers have focused on distributed sensor networking rather than on sensor data management (Balazinska et al., 2007).

Moreover when it comes to sensor data, the phrase "spatial is special" is particularly relevant. Sensing is essentially a spatially based sampling process in which each sensor data can generally be associated with location information. Within the context of the WSW, the phrase means that handling spatial properties of sensor data requires special algorithms, data models, databases, data presentations, system architectures, etc. There is a desire for a spatial information infrastructure designed specifically for the WSW. The spatial information infrastructure would aggregate the independent geo-sensor networks into a coherent Spatial Sensor Web (SSW). The main goal of this paper is to propose a new scalable and interoperable GIService architecture for the Spatial Sensor Web.

2. ARCHITECTURAL DESIGN CHALLENGES AND REQUIREMENTS

The SSW vision brings exciting and innovative applications. However, it also brings its architectural design unique challenges. Below, we outline two major challenges.

2.1 Scaling to accommodate an enormous amount of mobile and transient sensors

The number of sensors in the WSW could be enormous. These geographically distributed sensors would generate a massive amount of spatially referenced data streams. Many of the sensors may be mobile because they have to update locations often, while many wireless and battery-powered sensors may be transient because they frequently have to connect/reconnect. The above factors are challenging today's Internet GIService's scalability. The existing GIService architectures' centralized topologies are not designed for such large-scale and highly dynamic data sources. When existing architectures scale to accommodate users and sensors, its centralized components make the system vulnerable in that they are single points of failure. The centralized components are also system performance bottlenecks in that all additional system loads are added to them. A solution to making the system scalable by removing the architectures' centralized components is critical.

2.2 Allowing heterogeneous sensor networks to interoperate

Today's sensor networks are not interoperable. In other words, they cannot transparently allow each other to access, interchange, understand, and use the sensing resources. One of the major reasons for this inability to interoperate is that sensor networks are computers deployed in the fields. In order to accommodate the severely constrained environments, these sensor networks are built vertically with specialized hardware, operating systems, programming languages, and database systems. A solution to making these heterogeneous sensor networks interoperable while retaining their functionality and autonomy is necessary.

3. METHODOLOGY: PEER-TO-PEER SPATIAL ACCESS METHOD

We use a fully decentralized Peer-to-Peer (P2P) architecture to improve the system scalability. In addition, we integrate the P2P architecture with the Open Geospatial Consortium (OGC) interoperable Sensor Web Enablement (SWE) standards (Botts et al., 2006; Na and Priest, 2005) to make the system interoperable.

3.1 The power of decentralization: a P2P-based solution

The recent development of P2P systems, such as Gnutella (Ripeanu, 2001) and Skype (Guha et al., 2006) has demonstrated the viability of another new architecture for constructing large-scale distributed systems. Instead of depending on the centralized components to provide the necessary resources, the P2P systems operate on a cooperative model, where each peer leverages each other's available resources (*i.e.*, CPU, storage, bandwidth, etc.) for mutual benefit.

Our goal here is to find out whether the existing P2P architectures are applicable and useful to build a large scale GIService for the SSW. A suitable P2P-based GIService architecture at least needs to meet two criteria: (1) scalable and (2) supports spatial queries. From literature and existing systems, there are two types of P2P architectures (Androutsellis-Theotokis and Spinellis, 2004) : (1) unstructured P2P networks and (2) structured P2P networks. Unstructured P2P networks support spatial queries but cannot scale. Structured P2P networks can scale but do not support spatial queries. They cannot be directly used for building a GIService. However, the structured P2P networks offer a very attracting feature. They provide simple *hash table* functions and interfaces, *i.e.*, providing the insertion, lookup and deletion of (key, value) pairs across the P2P network. They are also referred as *Distributed Hash Table*s (DHT).

Attracted by DHTs scalability and simple hash table interfaces, we are motivated to design a solution to enable spatial queries on DHTs. Our solution is called **Peer-to-Peer Spatial Access Method** (P2PSAM). P2PSAM allows accessing to spatial objects according to spatial constraints. P2PSAM achieves this by building and maintaining dynamic spatial indexes using DHT's simple hash table functions (*i.e.*, put(*key*, *value*), remove(*key*), and get(*key*):*value*). Next, we briefly describe the P2PSAM. More details can be found in (Liang, 2008).

P2PSAM is a multi-step spatial query processing mechanism. The function offered by P2PSAM can be described as follows: '*Given a location-independent spatial identifier* X_s *of the data* Y *stored at some dynamic set of P2P nodes in the system, find* Y '. P2PSAM uses three steps to answer the above spatial query: (1) Filter step: the filter step accesses a spatial index stored on a DHT and locate the URLs pointing to the physical location of some candidate nodes. Candidate nodes are the P2P nodes hosting some data objects that "potentially" satisfy the spatial query constraint X_s. (2) Forward step: The forward step forwards X_s to the candidate P2P nodes in parallel. (3) Refinement step: Lastly, in the refinement step, the candidate nodes, after receiving X_s, execute the spatial query locally against the actual geometries of their hosting data objects, and then return the query results independently and in parallel.

In this paper, we use P2PSAM as the underlying P2P-based architectural framework. Next section, we present the design of GeoSWIFT 2.0 system. GeoSWIFT 2.0 is our implementation of using P2PSAM to build an interoperable and fully decentralized SSW service.

4. ARCHITECTURE

4.1 Function Offered

As an interoperable SSW service, the function offered by GeoSWIFT 2.0 can be described as follows:

Given a rectangle ***W****, find all observations of phenomenon* ***X*** *within* ***W***

To be more specific, GeoSWIFT accepts the following OGC Sensor Observation Service XML example request:

```
<GetObservation>
  <offering>SSW-Offering</offering>
    <observedProperty>urn:ssw:def:phenomenon:AirTemperature</observedProperty>
    <featureOfInterest>
      <ogc:BBOX>
        <gml:Envelope srsName="EPSG:4326">
          <gml:lowerCorner>56.00 -112.19</gml:lowerCorner>
          <gml:upperCorner>56.05 -112.16</gml:upperCorner>
        </gml:Envelope>
      </ogc:BBOX>
    </featureOfInterest>
</GetObservation>
```

Upon receiving the above request, the GeoSWIFT 2.0 member service nodes process the above query cooperatively and answer with the following OGC O&M XML example response:

```
<om:ObservationCollection>
  <om:member>
    <om:Observation>
      <om:time>
        <gml:TimeInstant>
          <gml:timePosition>2007-02-22T12:00:00Z</gml:timePosition>
        </gml:TimeInstant>
      </om:time>
      <om:location xlink:href="#location1"/>
      <om:procedure xlink:href="urn:ogc:ssw:sensor: thermometer#2057809"/>
      <om:observedProperty xlink:href=" urn:ssw:def:phenomenon:AirTemperature "/>
      <om:featureOfInterest>
        <om:Station>
          <om:position>
            <gml:Point gml:id="location1">
              <gml:pos srsName="EPSG:4326">56.0123 -112.1763</gml:pos>
            </gml:Point>
          </om:position>
        </om:Station>
      </om:featureOfInterest>
      <om:result uom="urn:ogc:def:uom:OGC:1.0.0:degCelcius">24.6</om:result>
    </om:Observation>
  +<om:member>
  +<om:member>
  +<om:member>
  +<om:member>
</om:ObservationCollection>
```

In the next section, we describe the GeoSWIFT 2.0 architecture.

4.2 GeoSWIFT 2.0 Architecture

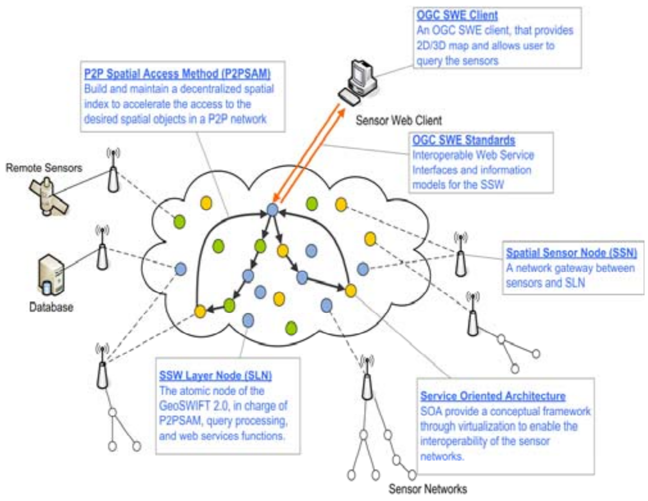

OGC SWE Client
An OGC SWE client, that provides 2D/3D map and allows user to query the sensors
P2P Spatial Access Method (P2PSAM)
Build and maintain a decentralized spatial index to accelerate the access to the desired spatial objects in a P2P network
Sensor Web Client
OGC SWE Standards
Interoperable Web Service Interfaces and information models for the SSW
Remote Sensors
Database
Spatial Sensor Node (SSN)
A network gateway between sensors and SLN
SSW Layer Node (SLN)
The atomic node of the GeoSWIFT 2.0, in charge of P2PSAM, query processing, and web services functions.
Service Oriented Architecture
SOA provide a conceptual framework through virtualization to enable the interoperability of the sensor networks.
Sensor Networks

Figure 1 The GeoSWIFT 2.0 architecture and the key components of the architecture

GeoSWIFT 2.0 architecture consists of the following four components:

1. **SSW Layer:** Logically, the GeoSWIFT 2.0 system is composed of multiple SSW Layers. Each SSW Layer offers a distinct sensing service for a specific phenomenon. For example, an SSW air temperature layer provides a service for air temperature observations. At the implementation level, each SSW Layer is composed by multiple SSW Layer Nodes of the same observation type.

2. **SSW Layer Node (SLN):** GeoSWIFT 2.0 is composed of many SLNs. SLNs play a service provider role in the SSW. SLNs are P2P nodes providing SSW services using a P2P spatial query infrastructure (*i.e.*, P2PSAM) with high level OGC SWE interfaces. From an SSW client's perspective, all GeoSWIFT 2.0's participating SLNs, as a whole, act as a single interoperable SSW service node.

3. **Spatial Sensor Node (SSN):** SSN is a network gateway that links the sensors and the SLNs. It plays a data provider role in the SSW. Each SSN collects raw data from sensors and streams the data to one or more SLNs.

4. **GeoSWIFT 2.0 Client:** This is an OGC SWE client that provides a 2D/3D map and allows user to use the SSW functions.

Next, we further illustrate the design of the GeoSWIFT 2.0 architecture.

4.2.1 SSW Layer

We use the "SSW Layer" concept, which is similar to the GIS layer concept, to manage the different types of observations offered in the GeoSWIFT 2.0. Logically, GeoSWIFT 2.0 consists of multiple SSW Layers. Each SSW Layer offers one and only one distinct type of the observation. Therefore, conceptually, each SSW Layer can be considered as a map layer of a particular phenomenon. For example, the air temperature SSW Layer provides a sensing service for air temperature observations. Employing the SSW Layer concept allows us to manage multiple logical sensing services in a single physical service framework.

As a naming convention, each SSW Layer has a layer name, a unique identifier, called the ***SSW Layer URN*** ($\boldsymbol{L_{URN}}$). L_{URN} is the Uniform Resource Name (URN) of the phenomenon observed by the SSW Layer. For example, an SSW Layer for air temperature may have an $\boldsymbol{L_{URN}}$ of *urn:ssw:def:phenomenon:AirTemperature*.

With the URN, sensor owners or SSW users can find the description or definition to explain the meaning of the $\boldsymbol{L_{URN}}$ in detail. How to define the URNs and maintain an infrastructure (e.g., a URN resolving service) to provide the URN definitions is out of the scope of this paper. However, we expect in the near future that there will be an international standards organization for managing and maintaining a set of standard URNs for phenomenon definitions. One existing example is the NASA SWEET ontology (http://sweet.jpl.nasa.gov/ontology/).

An SSW Layer is a collection of entries, *i.e.*, [*mbbox*, *endpoint*] pairs. Each pair links to one SLN network location. The *mbbox* is a minimum bounding box in a 2D space that approximates the sensing area covered by the SLN's sensors (*i.e.*, the SSNs), and the *endpoint* is a service endpoint allowing service requests to be forwarded to the SLN.

What makes the SSW Layer unique is how we build, maintain, update and query it in a P2P environment. We use P2PSAM as the framework to implement the SSW Layer. At an implementation level, each GeoSWIFT's SSW Layer is in fact one Linear Quad-Tree (LQT) that is realized using a DHT overlay network. This means that the SSW Layer (*i.e.*, the [*mbbox*, *endpoint*] pairs) are split into Quadtree nodes according to data density, and the Quadtree nodes are uniformly and randomly distributed and stored in the DHT overlay network. Next, we introduce the atomic component of GeoSWIFT 2.0: the SSW Layer Node (SLN).

4.2.2 SSW Layer Node (SLN)

SSW Layer Nodes (SLNs) are the atomic component of the GeoSWIFT 2.0 system. Physically, the GeoSWIFT 2.0 system consists of many SSW Layer Nodes (SLNs) connected to the Internet. As a whole, the SLNs together provide a P2P-based cooperative SSW service.

What makes the GeoSWIFT 2.0's design unique is the SLNs' fully decentralized Peer-to-Peer model, where each participating SLN has equal responsibilities. A GeoSWIFT 2.0 client can choose any SLN from the system and send requests to the chosen node. This means that any single SLN can be the access point of the service, and can accept client requests. Upon receiving the request from a client, the SLN forwards the request to the SLNs, which can potentially answer the query (*i.e.*, forwarding the queries to the candidate nodes). After receiving the query, the candidate SLNs then independently process the query and reply with the results in parallel.

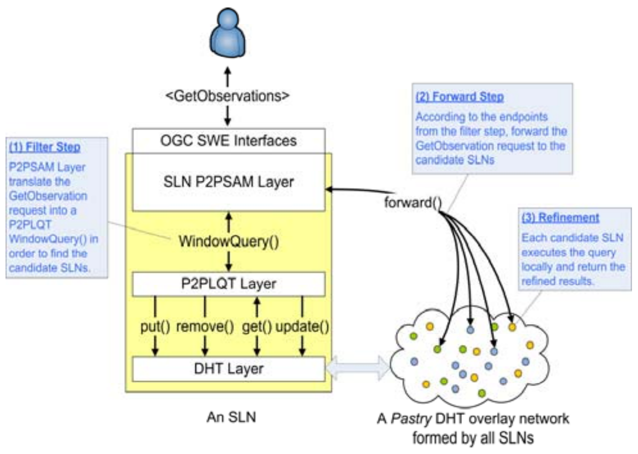
The core of the SLN's P2P model is the P2PSAM introduced in section 2. P2PSAM enables spatial lookup in a P2P environment. In order to implement the P2PSAM spatial query framework, we need a DHT overlay network to provide the P2P-based message routing and distributed file storage infrastructure. GeoSWIFT 2.0 uses *Pastry*[1] (Rowstron and Druschel, 2001) as its underlying DHT infrastructure. Each SLN owns a *Pastry* node instance. This provides the DHT put(*key*, *value*), remove(*key*), get(*key*), and update(*key*, *value*) APIs. The SLN's DHT APIs allow us to implement P2PSAM's spatial query mechanisms. Finally, on top of the DHT APIs and P2PSAM APIs, we can then implement the SLN Layer Management APIs to build, maintain, update, and query the SSW Layer. Next, we describe how the SLNs using P2PSAM, as the spatial query framework, to offer a P2P-based SSW service.

4.2.3 SLN's P2P-based SSW Service

The SSW service provided by the SLNs can be described using the following sample spatial query statement. For illustration purposes, we will use this sample query as an example throughout this section.

*Given a rectangle **W**, find all observations of phenomenon **X** within **W***

In our P2P-based SSW service architecture, the client can choose any SLN from the system and send the above request to the chosen SLN. After an SLN receive the query, it uses P2PSAM to process the query. Therefore, the SLNs process the query in three P2PSAM steps: (1) Filter Step, (2) Forward Step, and (3) Refinement Step (Figure 2).



<GetObservations>
OGC SWE Interfaces
SLN P2PSAM Layer
WindowQuery()
P2PLQT Layer
put() remove() get() update()
DHT Layer
An SLN
(1) Filter Step
P2PSAM Layer translate the GetObservation request into a P2PLQT WindowQuery() in order to find the candidate SLNs.
(2) Forward Step
According to the endpoints from the filter step, forward the GetObservation request to the candidate SLNs
forward()
(3) Refinement
Each candidate SLN executes the query locally and return the refined results.
A Pastry DHT overlay network formed by all SLNs

Figure 2 A diagram of how the SLNs offer a P2P-based SSW service using P2PSAM as the spatial query framework

1. **Filter Step:** The filter step finds the candidate nodes. The candidate nodes are the SLNs that can potentially contribute to the query, *i.e.*, the nodes whose mbbox intersects rectangle ***W*** in the query. The filter step is cooperatively performed by the SLNs, because the SLNs are also *Pastry* DHT nodes running P2PSAM application software. The final result of the filter step is a list of *endpoint*s of the candidate nodes.

2. **Forward Step:** After the initial SLN gets the *endpoint*s of the candidate nodes, according to the *endpoint*s it can then forward the query message to the candidate nodes.

3. **Refinement Step:** Upon receiving the queries the candidate SLNs independently process the queries. Each candidate SLN executes the query against the spatial properties (e.g., locations) of their hosting SSNs. This step is required because of the fact that the sensing area of the SLN (*i.e.*, the SLN's *mbbox*) overlaps the query does not mean that all of the SLN's sensor observations overlap the query. After finishing local query processing, each candidate node returns its own results to the SLN that initiated the query message. The SLN then streams the refinement results from the multiple candidate nodes back to the client that issued the query message.

[1] P2PSAM is not coupled with any DHT infrastructure. Therefore, SLNs can be easily implemented with other DHT infrastructures, such as Tapestry or CAN.

5. CONCLUSION

In this paper, we presented the GeoSWIFT 2.0, a new P2P based interoperable SSW architecture. GeoSWIFT 2.0 integrates the interoperable OGC SWE architecture (*i.e.*, open standards-based GIService for the SSW) with the scalable P2PSAM architecture in order to realize an interoperable and scalable SSW service.

GeoSWIFT's design improves existing Internet GIService's scalability by removing their centralized topologies. As we discussed in section 2, the centralized topologies are not suitable for large-scale and highly dynamic data sources. GeoSWIFT 2.0's P2P model removes the potential single point of failure and the system performance bottleneck. A GeoSWIFT 2.0 client can choose any SLN from the system and send requests to the chosen node. This means that any single SLN can be the access point of the service, and can accept client requests. This unique P2P design differentiates the GeoSWIFT 2.0 from other GIService systems or SSW service systems, such as the Univerity of Minnesota's MapServer (http://mapserver.gis.umn.edu/), Universität Münster's 52North SOS (http://52north.org/), or GeoSWIFT 1.0 (Liang et al., 2005; Liang and Tao, 2005; Tao et al., 2003).

REFERENCES

Androutsellis-Theotokis, S. and Spinellis, D., 2004. A Survey of Peer-to-Peer Content Distribution Technologies. *ACM Comput. Surv.*, 36(4): 335-371.

Balazinska, M. et al., 2007. Data Management in the Worldwide Sensor Web. *IEEE Pervasive Computing*, 6(2): 30-40.

Botts, M., Robin, A., Davidson, J. and Simonis, I., 2006. *OpenGIS Sensor Web Enablement Architecture Document*, Open Geospatial Consortium, Inc.

Guha, S., Daswani, N. and Jain, R., 2006. *An Experimental Study of the Skype Peer-to-Peer VoIP System*, The 5th International Workshop on Peer-to-Peer Systems, Santa Barbara, CA.

Liang, S.H.L., 2008. A New Fully Decentralized Scalable Peer-to-Peer GIS Architecture, ISPRS XXIth Congress, Beijing.

Liang, S.H.L., Croitoru, A. and Tao, V., 2005. A Distributed Geospatial Infrastructure for Sensor Web. *Computers and Geosciences*, 31(2): 221-231.
Liang, S.H.L. and Tao, V., 2005. *Design of an Integrated OGC Spatial Sensor Web Client*, Geoinformatics 2005, Toronto, Canada.

Na, A. and Priest, M., 2005. *OpenGIS Sensor Observation Service Implementation Specification*. OGC 05-088. Open Geospatial Consortium, Inc.

Ripeanu, M., 2001. *Peer-to-Peer Architecture Case Study: Gnutella Network*, First International Conference on Peer-to-Peer Computing (P2P'01), Linkoping, Sweden, pp. 0099.

Rowstron, A. and Druschel, P., 2001. *Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems*, 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany.

Tao, V., Liang, S.H.L., Croitoru, A., Haider, Z.M. and Wang, C., 2003. *GeoSWIFT: an Open Geospatial Sensing Services for Sensor Web*, GeoSensor Network Worshop 2003, Portland, USA.