Open Geospatial Consortium

Publication Date: 2013-11-07

Approval Date: 2013-09-25

Posted Date: 2013-09-13

Reference number of this document: OGC 13-054r1

External identifier of this OGC® document: http://www.opengis.net/doc/PER/Geo4NIEM

Category: OGC Public Engineering Report

Editor: Richard Martell

Summary and Recommendations of the Geospatial Enhancement for the National Information Exchange Model (Geo4NIEM) Interoperability Program Pilot

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Preface

Geospatial information technologies are increasingly a foundation for supporting Information Sharing Environment (ISE), homeland security (HLS), homeland defense (HLD), law enforcement (LE), emergency management (EM) and public safety missions in the US. The inability to transport, deliver and exchange geospatial information for critical geospatial assets increases the risk to the nation.

Many ISE HLS/HDS/LE mission partners have developed stand-alone geospatial information systems (GIS) or Common Operating Picture (COP) applications to support their stakeholder communities during incidents and for daily operational support. While different missions, these GIS/COP capabilities rely upon much of the same data or generate specific data during an event. The data are often stove-piped and not exposed to a broader community that could benefit from these data, resulting in duplication and delayed or incorrect decisions. While mission partners do not need to use the same GIS/COP tools, they could benefit from shared access to the common operating data and services used within these systems if they were exposed and exchanged using open standards.

Under the auspices of the Program Manager for the Information Sharing Environment (PM-ISE), an identified government-wide information sharing shortfall will be resolved by funding work to enhance the National Information Exchange Model (NIEM). The focus of this work is to further enhance the framework's geospatial exchange capability in light of guidelines and standards issued by the Open Geospatial Consortium (OGC) so as to significantly improve inter-government information sharing.

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, Inc. ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Contents

Patent Call

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

1 Introduction

1.1 Scope

The main impetus of the Geo4NIEM project was to enhance interoperability with respect to the handling of geospatial information based on OGC standards and NIEM-conformant information exchanges. Much of the work was focused on the GML (ISO 19136) data exchange standard and the mechanisms by which GML and NIEM data could be intermingled. A key driver was to clarify how data conforming to one framework could be included or "embedded" in the other using various encapsulation strategies.

A secondary goal was to conduct various software demonstrations in order to assess the feasibility of the various approaches and to explore the prospects for making use of fundamental OGC web services such as the Web Feature Service (WFS) and Web Map Service (WMS).

The Geo4NIEM project sponsors worked with OGC staff to outline specific functional requirements to meet the following objectives within the Geo4NIEM Pilot:

- Develop recommendations for the inclusion and standard use of embedded GML with NIEM Information Exchange Package Documentation sets (IEPDs).
- Develop recommendations for the standardize use of Naming and Design Rules and the use of adaptors (e.g. NIEM wrapper for GML).
- Test and demonstrate use of a standardized embedded GML and adaptors within NIEM IEPDs.
- Develop architecture documentation and a "Fact Sheet" for the use of embedded GML and adaptors for use with NIEM IEPDs.
- Develop recommendations for the inclusion of a geospatial domain within NIEM.

In order to accomplish these objectives, three primary tasks were identified:

Task 1: *Assessment and recommendations for embedded GML and adaptors*

The purpose of this task was to assess the support for geospatial data and the embedding of GML data structures within NIEM to determine if the current 'fragmented' structure allows for the most efficient and effective use of location information so as to improve understanding on the part of users and developers. The assessment will include the review of real-world IEPDs.

During this task participants reviewed the NIEM Naming and Design Rules (NDR, v1.3) for the efficient and effective use of adaptors where the use of a 'wrapper' around a GML element may impact the access and use of the GML. The use of external schema components by means of adapter types is described in section 7.7 of the NDR document.

Task 2: *Test and demonstration*

The purpose of this task is to apply the findings and recommendations for embedded GML and adaptors identified in Task 1 and perform a test and demonstration of the recommended architecture. The results of this task will be documented in an engineering report (ER) that presents the findings and recommendations, incorporating any refinements arising from implementation experience.

The performance of this task entails the following activities:

- Developing software components to demonstrate recommended enhancements in the context of the selected information exchanges (RFI and Vessel Position).
- Sharing and tracking information requests, including location information pertaining to pending RFIs.
- Exchanging messages that include data about vessel position and related information items such as vessel type.

Task 3: *Project documentation*

This task involves the preparation of two documents:

- An OGC engineering report that provides a technical summary of the project;
- A brief fact sheet (1-2 pp.) suitable for distribution by the project sponsors.

1.2 Sponsor objectives

The Geo4NIEM project sponsors have worked with OGC staff to articulate specific functional requirements in order to meet the following objectives:

- Develop recommendations for the inclusion and standard use of embedded GML with NIEM IEPDs.
- Develop recommendations for the standardize use of Naming and Design Rules and the use of adaptors (e.g. NIEM wrapper for GML).
- Test and demonstrate use of a standardized embedded GML and adaptors within NIEM IEPDs.
- Develop architecture documentation and “Fact Sheet” for the use of embedded GML and adaptors for use with NIEM IEPDs.
- Develop recommendations for the inclusion of a Geospatial Domain within NIEM

1.3 Participating organizations

1.3.1 Sponsoring Organizations

Geo4NIEM was sponsored by the following organizations:

- US Department of Homeland Security (DHS)
- NIEM Program Management Office
- Office of the Program Manager for the ISE (PM-ISE)

1.3.2 Geo4NIEM IP Team

The IP Team is an engineering and management team to oversee and coordinate an OGC Interoperability Initiatives. The IP Team facilitates architectural discussions, synopsizes

technology threads, and supports the specification editorial process. The IP Team is comprised of OGC staff and representatives from member organizations. The Geo4NIEM IP Team was as follows:

- Interoperability Program Executive Director: George Percival, OGC
- Initiative Director: Lewis Leinenweber, OGC
- IP Architect: Kurt Buehler, Image Matters LLC
- IT and Demonstration Support: Greg Buehler, OGC; Mark Buehler, OGC

1.3.3 Participating Organizations

The following organizations played one or more roles in Geo4NIEM as participants (responded to the RFQ/CFP and provided in-kind contributions)

- ArdentMC
- Carbon Project
- Galdos Systems
- Luciad

1.3.4 Document Contributors

The following participants (listed in alphabetical order by surname) made substantial contributions to the content of this report. All questions regarding this document should be directed to the editor or any of the contributors.

Name	Organization
Kurt Buehler	Image Matters LLC
Jeff Harrison	Carbon Project
Robin Houtmeyers	Luciad
Richard Martell (ed.)	Galdos Systems, Inc.

Thanks are extended to the reviewers who submitted comments over the course of the project.

1.4 Revision history

Date	Release	Editor	Primary clauses modified	Description
2013-09-05	13-054	R. Martell		Final project deliverable.
2013-09-13	13-054r1	R. Martell	1.1, 3.1.1, 5.1, 5.3, 6.2	- Clarified scope (cl. 1.1) - Replaced Fig. 7 - Added Listings 13, 14 - Added Figure 8 - Added new content to cl. 5.3 (Integrated scenario).

2 Technical context

This section provides a summary of the technical context that underpinned the project and established a foundation for the recommendations put forth in section 3. Two data exchange frameworks were considered:

- National Information Exchange Model (NIEM) v2.1
- Geography Markup Language (GML) v3.2, ISO 19136

The OGC has also developed many specifications for accessing, portraying, and managing geospatial data; many of these have been produced in collaboration with ISO/TC 211 (Geographic information/Geomatics).

2.1 Overview of NIEM 2.1

2.1.1 Naming and Design Rules

The *NIEM Naming and Design Rules* (NDR 1.3) document articulates rules and principles for developing conforming data components and schemas; these aim to define conformance criteria and to promote coherent use of the NIEM framework across all application domains. Conformance targets (Table 1) are used to characterize the scope of a particular rule.

Table 1 - NIEM conformance targets

Conformance target	Code
Reference schemas	REF
Subset schemas	SUB
Extension/Exchange schemas	EXT
Constraint schemas	CON
XML data (instance document)	INS

With respect to incorporating GML content, the modeling rules (section 7) are most relevant. The rules governing the use of external schemas were of especial interest since these are the ones that are concerned with the creation and use of external adapter types. The adapter mechanism is the only means of incorporating content from an external namespace into NIEM-conformant schemas.

External content is defined by schema components residing in some ‘foreign’ namespace. As suggested by Figure 1, an adapter type makes external content available to other NIEM elements; it should express a single, coherent concept so it can be used in a stand-alone manner.

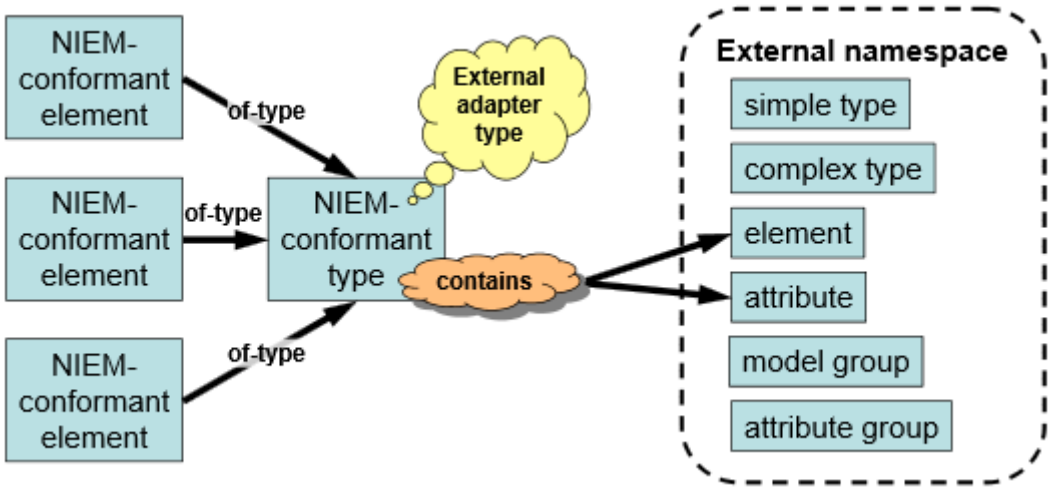

Figure 1: NIEM adapters[1]

Within NIEM Core, geospatial components are defined in the schema `geospatial.xsd` (with target namespace "http://niem.gov/niem/geospatial/2.1"). This schema imports schema components from GML 3.2.1 and defines several adapter types that include GML elements. Listing 1 shows the `PointType` adapter type, which exemplifies the definition of a GML adapter type.

Listing 1: PointType geometry adapter

```
<xsd:complexType name="PointType">
  <xsd:annotation>
    <xsd:documentation>A 2D or 3D geometric
point.</xsd:documentation>
    <xsd:appinfo>
      <i:Base i:namespace="http://niem.gov/niem/structures/2.0"
              i:name="Object"/>
     <i:ExternalAdapterTypeIndicator>true
     </i:ExternalAdapterTypeIndicator>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="gml:Point" minOccurs="0"
maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

[1] Reproduced from NDR 1.3, Figure 7-10.

The current content of the "http://niem.gov/niem/geospatial/2.1" namespace addresses community requirements that have emerged to date. Several of the recommendations proposed in section 3 propose to augment this namespace so as to extend its reach and align it with the GML simple features (GMLSF) profile.

2.1.2 Model Package Description

The NIEM Model Package Description (MPD, v1.1) specification establishes rules and provides guidance for assembling a set of related schema documents and other supporting files (artifacts) that constitute one of the five types of NIEM package:

- *Release* (major, minor, micro): contains a full set of harmonized reference schemas
- *Domain update* (of a release): contains reference schemas that represent changes to NIEM domains
- *Core update* (of a release): includes changes that apply to a particular NIEM core version
- *Information Exchange Package Documentation* (IEPD): defines and describes an implementable NIEM information exchange
- *Enterprise Information Exchange Model* (EIEM): incorporates Business Information Exchange Components (BIECs) that meet enterprise business needs for exchanging data using NIEM and provides a basis for developing a family of IEPDs

An MPD is distributed as a ZIP archive, the content of which depends on what type of MPD it is. The Geo4NIEM project focused on IEPDs used in the maritime and intelligence communities. A conforming IEPD package will generally contain the following kinds of artifacts:

- A catalog file
- A master document
- A change log
- One or more XML schemas that define the data exchange
- Sample XML instances
- A collection of subset, extension, and/or constraint schemas

2.2 Overview of GML 3.2

The Geography Markup Language (GML, ISO 19136:2007) is an XML vocabulary (grammar) for representing geospatial data. It is designed primarily to enable data exchanges between heterogeneous systems, and is not unlike NIEM in this regard. In essence, GML provides a foundation for building application-specific vocabularies; data are typically visualized using a separate portrayal language such as KML or SVG.

The GML standard does not stand alone and draws on many other standards, including:

- W3C XML 1.0
- W3C XML Schema (Parts 1 and 2)
- W3C XML namespaces
- W3C XLink
- W3C XPointer Framework (from shorthand pointers to XPath expressions)

- The ISO 19100 series of geomatics standards (19103, 19107, 19108, 19109, 19111, 19112, 19123)

2.2.1 GML application schemas

GML is best regarded as a data framework—it defines many base types and abstract entities that are used to define application-specific data elements within an application schema that imports GML schema components. An application schema makes use of GML schema components in order to define more specialized constructs for some domain (e.g. hydrography, transportation), and it may also reuse/extend components from other application schemas.

One of the most fundamental concepts is that of a feature, which is defined as an "abstraction of real world phenomena". New feature types are defined in an application schema as shown in Listing 2. Two things should be noted here: a feature instance can substitute for the (abstract) gml:AbstractFeature element wherever it may appear; furthermore, its type definition must derive from the base type gml:AbstractFeatureType.

Listing 2: Defining a feature type

```
<xsd:element name="SomeFeature" type="SomeFeatureType"
  substitutionGroup="gml:AbstractFeature" />
<xsd:complexType name="SomeFeatureType ">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="duration" type="xsd:duration" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

Clause 21 of ISO 19136 imposes these and other rules for constructing GML application schemas. Some of the most basic rules are listed below.

- Declare a target namespace that does **not** match the GML namespace
- Import the full GML schema
- Adhere to the "object-property pattern"
- Represent object properties as elements, not attributes
- Feature instances must substitute for gml:AbstractFeature (similar for other types)
- Follow widely adopted naming conventions (e.g. type names in UpperCamelCase notation)

2.2.2 Geometry model

The GML geometry model is based on ISO 19107, *Geographic information — Spatial schema*. The core of the geometry model is illustrated in Figure 2. The core model represents a high-level view, the "tip of the iceberg" so to speak. There are subtypes defined for almost all of the non-abstract geometry primitives shown in the diagram. One important thing to

appreciate is that while a reference to a coordinate reference system (CRS) is shown as optional (`GM_Object.CRS`), in practice it is required for every geometry object; however, in some cases the reference is inherited from a parent (or ancestor) geometry and may be omitted.

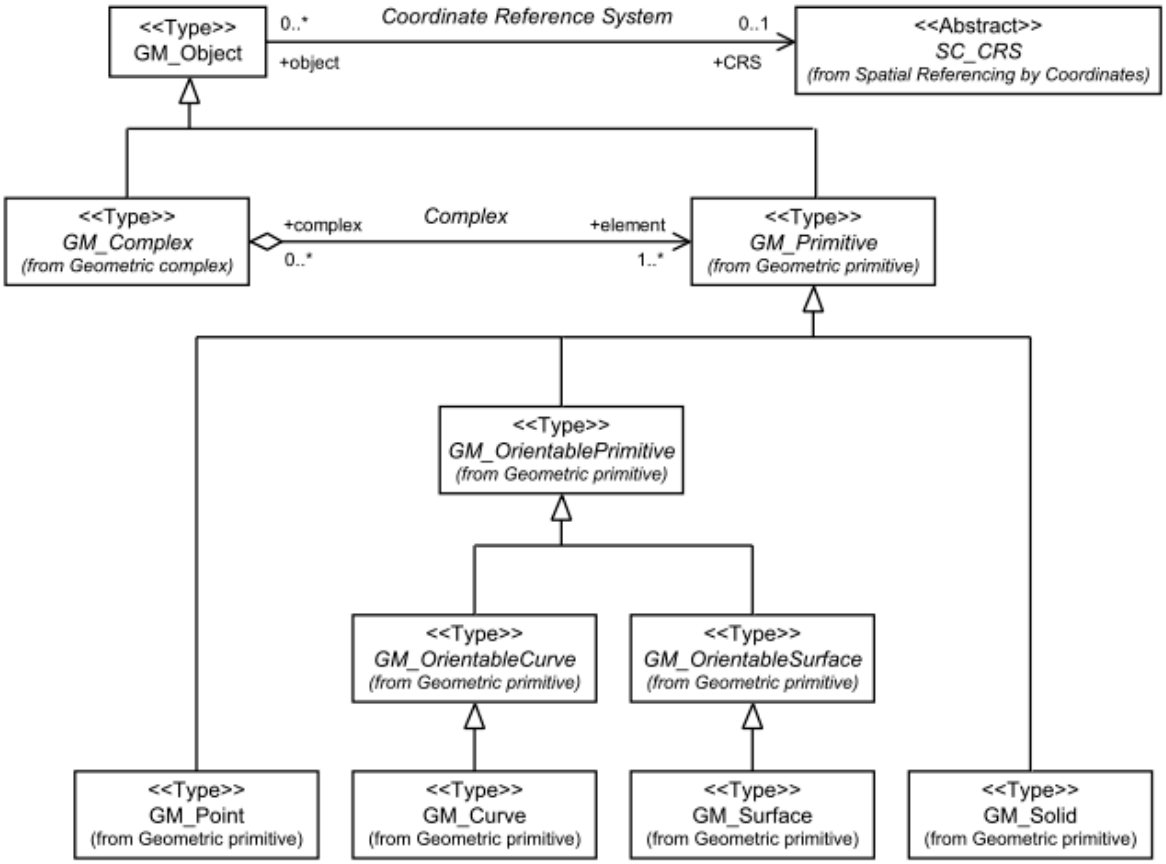

Figure 2: GML geometry model (from ISO 19107)

2.2.3 Simple Features Profile

Given that GML supplies a vast number of elements and types, profiles—strict subsets of GML—have been developed so as to restrict the schema components available for use in an application. The "Simple Features Profile" (OGC 10-100r3) is one of these: it imposes additional constraints with respect to the usage of both GML and XML Schema. Three conformance levels are defined, where each level (N) builds upon the lower level (N-1); these are summarized below.

SF-0

- Permits only simple non-spatial properties (0..1)
- Permits basic spatial properties having the following geometry values: Point, Curve (LineString), Surface (Polygon), plus collections of these

SF-1
- Allows user-defined property types (with simple or complex values)
- Relaxes the cardinality constraints from SF-0

SF-2
- No restrictions placed on non-spatial properties
- Is aligned with ISO 19125-2 (*Geographic information -- Simple feature access -- Part 2: SQL option*)

2.3 Data integration approaches

One of the most significant findings of this project revealed some fundamental conflicts between NIEM and GML. It was discovered early in the project that conforming to both NIEM 2.1 and GML 3.2 *at the same time* was not possible due to contradictory constraints. This discordance gave rise to a two-pronged approach: by adopting one of the standards as the principal framework, it is possible to integrate 'foreign' data from external namespaces in a conformant manner.

Two constraints in particular—one from each framework—frustrated efforts to achieve a "super-compliant" integration:

1. ISO 19136, cl. 7.1.3: "A GML property shall not be derived from gml:AbstractGMLType, shall not have a gml:id attribute, or any other attribute of XML type ID."
2. NIEM NDR v1.3, sec. 7.4.1: Object Types [Rule 7-39]: NIEM object type derivation

As a consequence of these rules, a conforming NIEM adapter type cannot also be a conforming GML property type. Furthermore, a valid GML object cannot also be a conforming NIEM object. However both frameworks do allow for the inclusion of 'foreign' content, thus giving rise to the two approaches mentioned above (that is, GML in NIEM vs. NIEM in GML).

2.3.1 Incorporating GML objects into NIEM

Over the years the GML user community has devised a number of recommendations for "embedding" GML elements within non-GML data structures. The general guidelines are presented below:

- Only embed GML objects: elements derived from gml:AbstractGMLType (substitute for gml:AbstractGML)
- Include any GML dependent attributes, elements and content models
- Embed GML objects in "property" elements that express entity relationships (i.e. semantics)
- Create a GML profile schema to simplify schema-aware processing

Some additional constraints can be imposed in order to distinguish an embedded GML object:

- It does not appear as the document element;
- It is contained within a non-GML (parent) element;
- It is valid with respect to the applicable GML or GML application schema(s).

NIEM employs an "adapter" mechanism to incorporate data (element and attribute infoset items) from non-NIEM namespaces in NIEM-conformant elements. An important aspect of this project was to assess to what degree existing GML adapters are consistent with prevailing practices and conventions in the OGC community. Most of the recommendations appearing in section 3.1 strive to achieve harmonization in this area.

2.3.2 Encapsulating NIEM objects in GML features

Almost all complex GML content models reflect what is known as the "object-property" pattern, which is rooted in the RDF model exemplified by the triple: Subject (node) → Predicate (arc) → Object (node). The pattern also arises as a consequence of deriving XML Schema components from UML models as shown in Figure 3. In fact, such mapping rules are codified in ISO 19118 (*Geographic information – Encoding*).

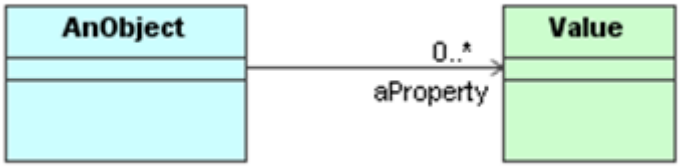

Figure 3: Object and properties in UML

The value of a GML property may be simple or complex. Listing 3 shows a Road feature instance with several properties.

Listing 3: Road feature properties

```
<tns:Road gml:id="R1" xmlns="http://example.org">
  <tns:centerLine>
    <gml:LineString>…</gml:LineString>
  </tns:centerLine>
  <tns:numLanes>4</tns:numLanes>
  <tns:category>boulevard</tns:category>
  <tns:intersection>
    <Road gml:id="R2">
      <tns:centerLine>
        <gml:LineString>…</gml:LineString>
      </tns:centerLine>
      <tns:numLanes>2</tns:numLanes>
      <tns:category>street</tns:category>
    </Road>
  </tns:intersection>
</tns:Road>
```

Feature properties are the basic construct used to include a NIEM data object within a GML object.

3 Findings and recommendations

This section proposes a set of findings and recommendations covering various aspects of the work undertaken by project participants with the overall aim of enhancing interoperability with respect to the exchange, processing, and visualization of geospatial data. Two fundamental approaches were investigated, reflecting which data framework is selected as the principal 'container':

a) Incorporate GML objects into NIEM (adapters);
b) Encapsulate NIEM objects in GML feature representations (feature properties).

3.1 GML adapters in NIEM

The recommendations in this section are mainly concerned with the GML adapters defined in the geospatial.xsd schema, a NIEM reference schema. In general, the standard adapter mechanism should be employed to import valid GML content into NIEM-conformant data structures as it is consistent with the guidelines put forth in section 2.3.1. If a desired GML object is not available through a predefined adapter type (e.g. 3D solid geometry), IEPD developers are encouraged to define one in an IEPD schema using the standard adapters as a template.

This section recommends a number of specific changes to the geospatial.xsd reference schema. The amendments should be considered for inclusion in a future NIEM release following the 3.0 release in fall 2013, perhaps under the umbrella of a formal geospatial domain should one be inaugurated.

3.1.1 Support GML Simple Features (GMLSF) profile

The NIEM framework currently lacks a formal geospatial domain. However, several geospatial types of general utility are defined in the geospatial.xsd schema. This broad recommendation urges that the reference schema define geometry adapters that cover the basic geometry types allowed by the *GML Simple Features Profile* (OGC 10-100r3). The GMLSF profile includes a subset of the complete GML geometry model that is sufficient for most applications; the following geometry types are permitted:

- gml:Point
- gml:LineString
- gml:Curve (with restrictions; see 3.1.5)
- gml:Polygon
- gml:Surface (with restrictions; see 3.1.5)
- gml:MultiPoint
- gml:MultiCurve
- gml:MultiSurface
- gml:MultiGeometry

The same set of geometry types are permitted in all three GMLSF conformance levels (see clause 8.4.4.11 in OGC 10-100r3), so conformance level SF-0 would suffice; higher levels impose additional constraints that are unrelated to geometry representations.

3.1.2 Define a generic feature adapter

A generic feature adapter may contain any GML feature representation—an element that can substitute for the (abstract) *gml:AbstractFeature* element. The inclusion of such an adapter

would provide a simple mechanism for incorporating GML features within any NIEM application domain. For example, a building feature constructed in accord with the CityGML application schema could be inserted where appropriate.

The element declaration and corresponding type definition are shown in Listing 4. Within an IEPD exchange schema it may also be desirable to restrict allowable features to specific types by using the XML Schema restriction mechanism or by defining Schematron constraints that validate message content. Or one could define a domain-specific adapter using the generic adapter type as a template.

Listing 4: General-purpose GML feature adapter

```
<xsd:element name="gmlFeature" type="geo:FeatureAdapterType">
  <xsd:annotation>
    <xsd:documentation>A generic GML feature
adapter.</xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:complexType name="FeatureAdapterType">
  <xsd:annotation>
    <xsd:documentation>A data type that encapsulates a GML feature
instance.</xsd:documentation>
    <xsd:appinfo>
      <i:Base i:namespace="http://niem.gov/niem/structures/2.0"
              i:name="Object"/>

<i:ExternalAdapterTypeIndicator>true</i:ExternalAdapterTypeIndicator
>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="gml:AbstractFeature" minOccurs="1"
maxOccurs="1">
          <xsd:annotation>
            <xsd:documentation>This abstract element denotes any GML
feature representation.</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

3.1.3 Define a general-purpose geometry adapter

A general-purpose geometry adapter may contain any GML geometry representation: namely, an element that can substitute for the *gml:AbstractGeometry* element. Such an adapter type provides a simple means of incorporating any type of spatial reference using geographic coordinates and emulates the gml:GeometryPropertyType permitted by the GML Simple

Features (GMLSF) profile. A loosely constrained geometry adapter like this can be useful in contexts where no specific geometry representation can be assumed.

The element declaration and corresponding type definition are shown in Listing 5. As for a feature adapter, more specialized geometry adapters could be defined in an IEPD exchange schema if a need arises for a geometry type that is not already available, such as a gml:Solid that lies beyond the ambit of the GML Simple Features profile.

Listing 5: General-purpose GML geometry adapter

```
<xsd:element name="gmlGeometry" type="geo:GeometryAdapterType">
  <xsd:annotation>
    <xsd:documentation>A general-purpose GML geometry adapter.
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:complexType name="GeometryAdapterType">
  <xsd:annotation>
    <xsd:documentation>A data type that encapsulates a GML geometry
element.</xsd:documentation>
    <xsd:appinfo>
      <i:Base i:namespace="http://niem.gov/niem/structures/2.0"
              i:name="Object"/>
      <i:ExternalAdapterTypeIndicator>true
      </i:ExternalAdapterTypeIndicator>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="gml:AbstractGeometry" minOccurs="1"
                     maxOccurs="1">
          <xsd:annotation>
            <xsd:documentation>This abstract element denotes any GML
geometry representation.</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

3.1.4 Remove adapters for curve segments

Several of the GML elements in the geospatial.xsd schema do not represent geometry types; in fact, they are not GML objects at all[2]. rather they are curve segments that only appear as constituents of gml:Curve[gml:segments]:

[2] According to ISO 19136, a GML object must have a content model derived from gml:AbstractGMLType. Curve segments do not satisfy this constraint.

- gml:Arc
- gml:ArcByCenterPoint
- gml:Circle
- gml:CircleByCenterPoint

Adapters for curve segments should be removed as they do not contain GML objects and are redundant; they can be included in the context of a gml:Curve, for which an adapter type already exists.

3.1.5 Add Schematron (ISO 19757-3) constraints for GMLSF conformance

The preceding recommendation proposes to remove adapters for curve segments. Now, the definition of the gml:Curve element imposes no restrictions whatsoever on the type of curve segments that may appear within it. In keeping with recommendation 3.1.1 to adhere to the GMLSF profile, it is recommended that Schematron (ISO 19757-3) constraints be defined to restrict allowable curve segments and surface patches in accord with the GML Simple Features Profile v2.0 (OGC 10-100r3), Level SF-0.

Allowable curve segments include the following elements: LineStringSegment, Arc, Circle, and CircleByCenterPoint. For a gml:Surface element only PolygonPatch is allowed. Appendix A includes a Schematron schema that could be used for this purpose. The phase "SF-0" enables two rule sets pertaining to allowable curve segments (gml:Curve/gml:segments) and surface patches (gml:Surface/gml:patches).

3.1.6 Deprecate specific GML adapters (elements)

There is an element declaration for every GML adapter type definition in the geospatial.xsd schema. However, these are unlikely to be used very widely because they have very weak semantics. It is much more likely that IEPD developers will reuse the type definitions but elect to declare elements that are more meaningful within their domain.

Previous versions of GML defined several geometry properties as a convenience to schema developers, such as gml:centerLineOf. But over time these have all been deprecated since they were rarely used. It is the property type definitions that are reused, not the property elements.

The specific GML adapter elements—those that contain concrete GML geometry elements—should be deprecated and eventually removed altogether. Instead, encourage IEPD developers to reuse the adapter type definitions within IEPD schemas. For example, if the footprint of a transmission tower is described by a gml:Polygon, then it's better to declare the "footprint" property (of type geo:PolygonType), rather than "geo:Polygon" which says almost nothing about the relationship. The situation becomes even more vexing if another polygon is associated with the tower, say one that characterizes the service area.

3.1.7 Add a general-purpose geometry adapter to nc:LocationType

In the niem-core.xsd schema, the `LocationType` schema component provides "A data type for a geophysical location". The type definition has some 21 child elements: all of these are optional, and none make use of an existing geometry adapter to specify a spatial reference by coordinates. There are several elements that may be used to specify a point location:

- LocationMapLocation
- LocationMGRSCoordinate
- LocationTwoDimensionalGeographicCoordinate
- LocationUTMCoordinate)

These elements do not represent point locations in a consistent manner and also include information about the coordinate reference system (CRS). In summary, this core type definition is rather more complicated than it need be.

We recommend that spatial referencing by coordinates be accomplished by adding a general-purpose geometry adapter to nc:LocationType. Following from recommendation 3.1.3, the geo:gmlGeometry adapter could be used to include any simple GML geometry representation, not just a gml:Point element (see Listing 6). The four existing point location elements mentioned above would then become redundant and could be deprecated.

Listing 6: Spatial referencing by coordinates in nc:LocationType

```
<xsd:complexType name="LocationType">
  <xsd:complexContent>
    <xsd:extension base="s:ComplexObjectType">
      <xsd:sequence>
        <xsd:element ref="geo:gmlGeometry" minOccurs="0" />
        <xsd:element ref="nc:LocationAddress" minOccurs="0"
maxOccurs="unbounded"/>
        <!-- other elements omitted -->
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

Every GML geometry must refer to a CRS such that the location is unambiguous. The srsName attribute is an absolute URI value that identifies the applicable reference system. In practice, this URI denotes a "well-known" CRS such as those defined in the EPSG geodetic parameter dataset. All of the standard EPSG definitions can be accessed from a publicly-available registry service at <http://www.epsg-registry.org/>. In fact, http URIs can be resolved to obtain CRS definitions (as GML) if desired; for example:

- WGS 84: http://www.opengis.net/def/crs/EPSG/0/4326
- WGS 84 (3D): http://www.opengis.net/def/crs/EPSG/0/4979
- WGS 84 / UTM zone 10N: http://www.opengis.net/def/crs/EPSG/0/32610

It is also worth noting that NGA has released a comprehensive specification dealing with location information based on geographic coordinates and geographic identifiers. The *Time-Space-Position Information* (TSPI 2.0) specification includes several XML Schema documents that define custom geometry elements derived from GML geometry types. Any of these geometry elements may appear as the child of a geo:gmlGeometry adapter element since they are all substitutable for `gml:AbstractGeometry`.

3.2 GML application schemas

Since NIEM is not itself a messaging framework, NIEM information exchanges can conceivably be realized using a variety of implementation methods such as web services or asynchronous messaging. The OGC has developed a wide-ranging family of service specifications for accessing, processing, and visualizing geospatial data; many of these can operate on GML feature representations. Thus being able to incorporate NIEM objects within representations of GML features greatly expands the possibilities for operationalizing NIEM information exchanges.

The recommendations in this section deal with various aspects of incorporating content from a NIEM exchange message into a GML feature representation. These recommendations are more forward-looking in that they anticipate scenarios in which a user wishes to make full use of OGC web services. That is, the goal is to make NIEM data ready to be consumed, processed, or portrayed by various kinds of OGC services without requiring extensive customization of standard implementations.

3.2.1 Convert NIEM exchange messages to GML feature types

Map elements of message entities to a GML feature type to expedite the use of OGC web services for search, retrieval, and portrayal. Such a transformation can be realized by defining a feature type within a simple GML application schema, where the values of the feature properties are NIEM data objects; this is very similar to the way in which adapters are used in the NIEM framework. In effect, a feature property plays the role of a NIEM adapter.

Since both the source and result are XML instances, it would be quite straightforward to automate the conversion using an XSLT processor. In this case, the mapping rules would be embodied in a stylesheet that declares how the GML feature instance is constructed from a NIEM message. Listing 7 provides an example based on the vessel position report, where an incoming posx:Message[3] is transformed to produce a VesselTrack feature in some GML application namespace. The main NIEM objects are highlighted in **boldface**. Appendix C includes a simple XSLT stylesheet that was used to perform the transformation.

Listing 7: A VesselTrack feature instance

```
<tns:VesselTrack gml:id="IMO0000001"
  xmlns:tns="http://example.org/maritime"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:nc="http://niem.gov/niem/niem-core/2.0"
  xmlns:m="http://niem.gov/niem/domains/maritime/2.1"
  xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2">
  <tns:exchangeMetadata>
    <tns:PositionMessageMetadata>
      <!-- message metadata elements -->
    </tns:PositionMessageMetadata>
  </tns:exchangeMetadata>
  <tns:vesselInfo>
    <mda:Vessel>
```

[3] The posx prefix here denotes the "http://niem.gov/niem/domains/maritime/2.1/position/exchange/3.2" namespace.

```
      <!-- vessel properties -->
    </mda:Vessel>
  </tns:vesselInfo>
  <tns:track owns="true">
    <!-- mda:Position elements indicating vessel's course over
ground -->
  </tns:track>
</tns:VesselTrack>
```

A GML application schema could even be supplied alongside the exchange schema, thereby facilitating easy and consistent access to OGC web services. For any NIEM information exchange that may benefit from an OGC processing or visualization service, IEPD developers should include a supplementary GML application schema along with a set of rules for creating the GML (see section 3.2.2).

3.2.2 Minimize the number of GML mapping rules

In developing a GML application schema, developers should strive to minimize the number of mapping rules so as to not completely obliterate or "shred" the original NIEM data structures. Several advantages arise from this:

- The mappings are simpler to implement.
- It obviates the need for an inverse mapping (to reconstitute the message), since most of the original data structures remain intact in the result.

Note that in Listing 7 most of the original message content is preserved in the feature representation; the mda:Vessel and mda:Position elements are unaltered. As a consequence, if such a feature were inserted into a WFS there would be little need to support a NIEM exchange message as an alternative output format since the original elements are preserved. However, if a particular NIEM IEP does not admit such a simple mapping then it may be necessary to implement an inverse mapping.

3.2.3 Separate out message-level metadata

Exchange messages often contain metadata elements—possibly of an ephemeral nature—that characterize the context for using or interpreting message content. Some of the metadata may not even be needed once the message has been received. This recommendation deals with the disposition of message-level metadata elements.

For example, a vessel position report inherits a variety of information items from mda:DocumentType, including:

- mda:RecordIDURI
- mda:MessageExerciseName
- mda:MessageStatusCode
- mda:MessageSourceSystemName
- mda:ICISMMarkings
- @mda:securityIndicatorText

Message-level metadata elements like these should be extracted and consolidated within a separate metadata entity, which can then be accessed from a WFS (with a feature) or perhaps by means of a CSW-based catalogue service. In Listing 7 the value of the

tns:exchangeMetadata property is a tns:PositionMessageMetadata element, which basically serves as a simple container for all message-level metadata information items.

3.3 Web services

3.3.1 Extend WFS v2 implementations to accept NIEM messages

The Web Feature Service (WFS) is the most fundamental OGC data access service (see section 4.1 for an overview). Defining a GML feature type that encapsulates NIEM data is a necessary step in order to make use of a WFS, but this can be further expedited by extending a WFS implementation to accept a NIEM exchange message as an alternative input format. In practice a WFS ingests a NIEM exchange message and automatically transforms it into a feature and inserts it into the underlying data store.

A NIEM-aware WFS has an appropriate entry for the `inputFormat` request parameter in its capabilities document. Listing 8 shows how to indicate that a WFS 2.0 implementation can accept a vessel position message in the body of a transaction request.

Listing 8: Advertising support for inserting NIEM message entities (WFS 2.0)

```
<Operation xmlns="http://www.opengis.net/ows/1.1"
name="Transaction">
  <DCP>
    <HTTP>
      <Post xlink:href="http://localhost/wfs2"/>
    </HTTP>
  </DCP>
  <Parameter name="inputFormat">
    <AllowedValues>
      <Value>

http://niem.gov/niem/domains/maritime/2.1/position/exchange/3.2
      </Value>
    </AllowedValues>
  </Parameter>
</Operation>
```

The parameter value is the namespace name (URI) of the NIEM message entity. The listing above advertises the fact that a WFS that can accept vessel position reports; the URI value is the target namespace from the exchange schema.

Note that it is also possible to extend a WFS implementation to support NIEM messages as an alternative output format for a GetFeature request. The mapping rules would be applied in reverse so as to construct a NIEM message from an existing GML feature representation. The value of the outputParameter would be the appropriate message namespace name (Listing 9).

Listing 9: Advertising support for returning NIEM message entities (WFS 2.0)

```
<Operation xmlns="http://www.opengis.net/ows/1.1" name="GetFeature">
  <DCP>
    <HTTP>
      <Get xlink:href="http://localhost/wfs2"/>
```

```
    </HTTP>
  </DCP>
  <Parameter name="outputFormat">
    <AllowedValues>
      <Value>
http://niem.gov/niem/domains/maritime/2.1/position/exchange/3.2
      </Value>
    </AllowedValues>
  </Parameter>
</Operation>
```

3.3.2 Use a registry service to manage MPDs

A NIEM-conformant IEPD includes a variety of artifacts such as schemas, instance documents, code lists, and supporting documentation. The IEPD search facility does provide a rudimentary means of finding and downloading publicly available IEPDs. However this seems to be primarily a repository for completed artifacts—there doesn't appear to be any support for lifecycle management, versioning, provenance (lineage), or more complex classification-based searches (e.g. by NIEM domain).

Employing a registry service to manage IEPD artifacts throughout their entire lifecycle would be beneficial from a governance perspective, and it would certainly provide more powerful search capabilities. For example, the kinds of discovery scenarios that become possible include the following:

- Find all IEPDs that apply to the maritime domain.
- Find IEPDs that use schema components from the geospatial schema.
- Find all IEPDs that are derived from a particular EIEM.

Section 4.3 provides an overview of the OGC CSW-ebRIM registry service, which marries the OGC catalogue interfaces to a general, extensible information model that can be readily adapted for specific applications. The development of a 'NIEM' extension package would be one way to customize a registry implementation to support the creation and maintenance of MPD artifacts.

4 OGC web service enablement

Perhaps the most significant advantage that accrues from encapsulating the content of a NIEM exchange message within a GML feature is the capability of utilizing various OGC web services for searching, retrieving, and visualizing data. Most of the fundamental OGC web services can operate on GML feature representations, so taking this step enables the use of a multitude of open-source and commercial software products that implement OGC interfaces. The following sections present summaries of OGC services that are highly relevant in NIEM domains.

4.1 Web Feature Service (WFS)

A Web Feature Service (WFS) provides interfaces that operate on GML object representations. Feature representations may be searched, retrieved, inserted, updated, and

deleted. While a given WFS may support alternative data formats, all conforming implementations must be capable of handling GML feature representations.

The WFS 1.1 specification (OGC 04-094) is coupled to GML 3.1; feature instances must substitute for the (abstract) *gml:_Feature* element. The current WFS 2.0 version (OGC 09-025r1, also published as ISO 19142:2010) is not tied to any particular version of GML, although the default input/output format is GML 3.2 (target namespace is "http://www.opengis.net/gml/3.2").

The WFS 2.0 standard and its related specifications are illustrated in Figure 4. The base standard constitutes the "tip of the iceberg" as quite a few other specifications also govern WFS behavior. WFS 1.1 and 2.0 use OGC Filter 1.1 and 2.0, respectively, for data filtering. Filter 2.0 adds the concept of temporal filtering, which could be useful for working with time-dependent NIEM data. In WFS 2.0 the inputFormat/outputFormat parameter value is a string or MIME media type that is advertised in the capabilities document; an implementation that can handle other data formats will list them in the service metadata.

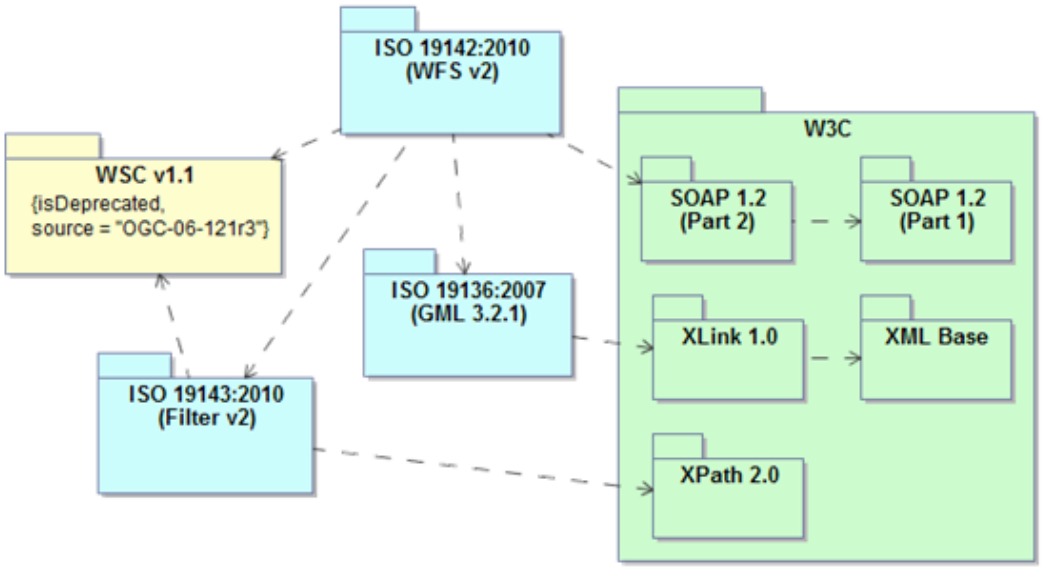

Figure 4: WFS2 specifications

Adding support for NIEM objects is best accomplished by incorporating them within a GML feature type, instances of which can then be accessed by means of standard WFS interactions. In order to do this, a WFS must be extended to support a NIEM message entity (see section 3.3.1). Given a WFS implementation capable of ingesting a NIEM message entity, a request to insert a NIEM message would be structured as shown in the following listing.

Listing 10: Inserting a NIEM message entity into a WFS

```
<wfs:Transaction version="2.0.0" service="WFS"
 xmlns:wfs="http://www.opengis.net/wfs/2.0">
  <wfs:Insert
```

```
    srsName="http://www.opengis.net/def/crs/EPSG/0/4326"
    inputFormat="http://niem.gov/niem/domains/maritime/2.1/position/
exchange/3.2">
    <posx:Message

xmlns:posx="http://niem.gov/niem/domains/maritime/2.1/position/
exchange/3.2">
      <!-- message content omitted -->
    </posx:Message>
  </wfs:Insert>
</wfs:Transaction>
```

4.2 Web Map/Feature Portrayal Service (WMS/FPS)

A Web Map Service (WMS) provides an API for producing graphical representations (maps) of spatial data. A Feature Portrayal Service (FPS) is a specialized WMS that implements the SLD (Styled Layer Descriptor) profile and is not tightly coupled to particular data sources; it is thus characterized as a "component" WMS. An FPS implementation has no predefined layers or styles and can portray feature data obtained from a remote WFS.

An SLD document (see OGC 05-078r4) controls how a map is generated from some source data. It is an XML resource that contains a sequence of styled layer definitions that may include both system-defined and user-defined elements. Listing 11 shows a simple user-defined layer that might appear in an SLD document; it includes a styling rule to mark the (point) location of a feature instance with a red triangle.

Listing 11: User-defined style

```
<UserLayer xmlns="http://www.opengis.net/sld"
           xmlns:se="http://www.opengis.net/se">
  <se:Name>PrimitiveGeoFeature</se:Name>
  <UserStyle>
    <se:Name>PrimitiveGeoFeature-point</se:Name>
    <IsDefault>true</IsDefault>
    <se:FeatureTypeStyle>

<se:FeatureTypeName>sf:PrimitiveGeoFeature</se:FeatureTypeName>
      <se:Rule>
        <se:Name>pointProperty</se:Name>
        <se:PointSymbolizer>
          <se:Name>Triangle-Red</se:Name>
          <se:Geometry>
            <ogc:PropertyName>sf:pointProperty</ogc:PropertyName>
          </se:Geometry>
          <se:Graphic>
            <se:Mark>
              <se:WellKnownName>triangle</se:WellKnownName>
              <se:Fill>
                <se:SvgParameter
name="fill">#FF0000</se:SvgParameter>
              </se:Fill>
            </se:Mark>
```

```
            <se:Size>8.0</se:Size>
          </se:Graphic>
        </se:PointSymbolizer>
      </se:Rule>
    </se:FeatureTypeStyle>
  </UserStyle>
</UserLayer>
```

Any WMS can optionally support dimensions, or parameterized layers; typical examples include time or elevation data. This facility can be useful for working with time-dependent NIEM data. For example, one might desire to obtain a plot of all vessel tracks within some time interval.

4.3 Registry Service (CSW-ebRIM)

A registry service is a specialized catalogue service that provides a web-based API for storing and retrieving shared information resources, including "metadata" resources such as code lists, classification schemes, schemas, service descriptions, dataset descriptions, and styled layer descriptions. Furthermore, a registry typically exemplifies a formal governance process such as those documented in ISO 19135 or ISO 11179-6.

The OGC CSW-ebRIM Registry Service specification is an application profile based on the CSW part (cl. 10) of the *OGC Catalogue Service Implementation Specification* (v2.0.2, OGC 07-006r1). The specification includes three parts:

- Part 1: ebRIM profile of CSW (OGC 07-110r4)
- Part 2: Basic extension package (OGC 07-144r4)
- Part 3: Abstract test suite

The CSW-ebRIM profile binds the CSW interfaces to the OASIS ebXML registry information model (ebRIM 3.0), a general, extensible model that can accommodate a wide variety of information resources. The main specification and its dependencies are shown in Figure 5.

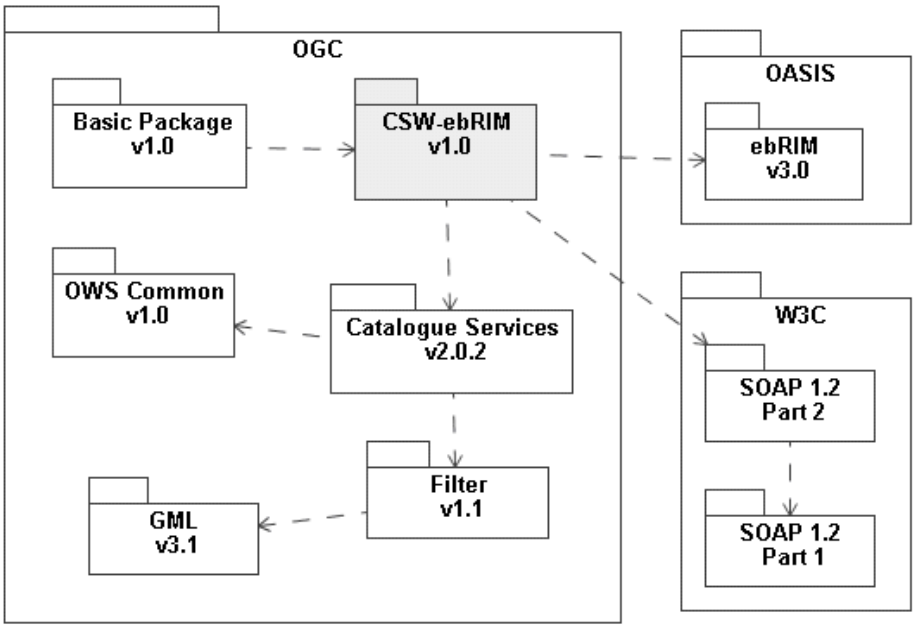

Figure 5: CSW-ebRIM specifications

The ebRIM information model provides several extension points (Figure 6):

1. Named ‘slots’, which denote arbitrary properties having simple or complex values;
2. ExtrinsicObject types with associated repository items represented by some MIME entity;
3. Various kinds of associations that relate registered items;
4. Classification nodes (terms) that extend an existing classification scheme;
5. Classification schemes (e.g. code lists, taxonomies) used to classify a registered item in some manner.

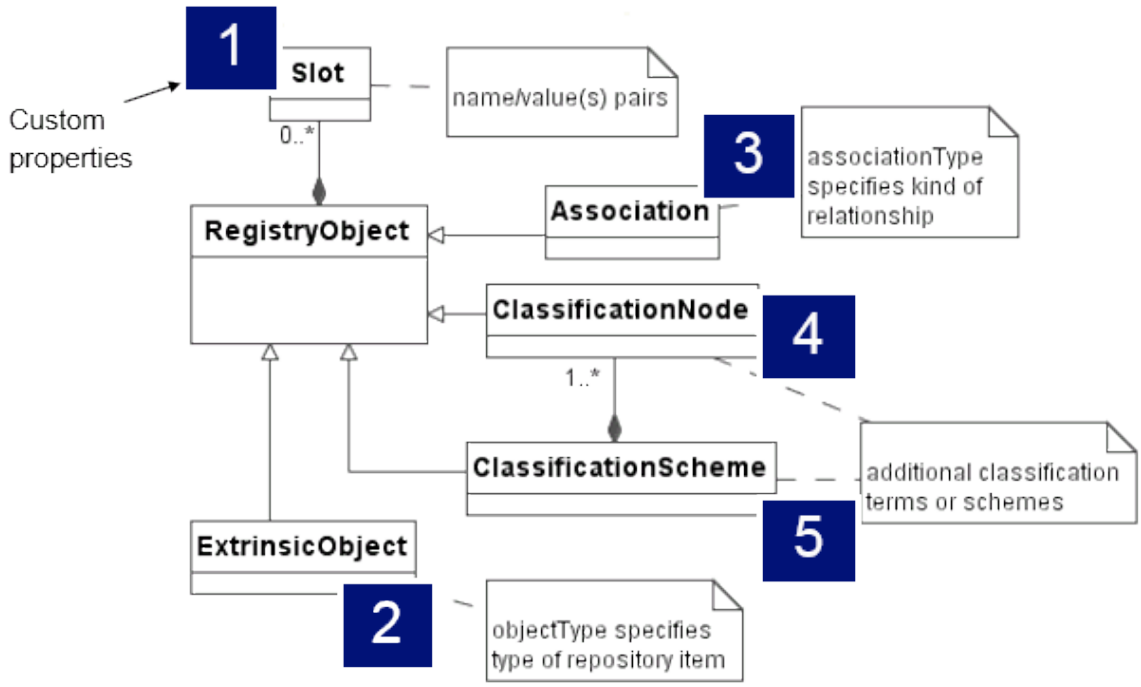

Figure 6: ebRIM extension points

The registry information model can be tailored for particular application domains, allowing a registry service to be customized to meet the needs of diverse user communities. For example, a geodetic registry can include elements for defining coordinate reference systems and related components such as datums and prime meridians. A 'Portrayal' package might include elements for working with the style descriptors and symbol collections used in map production.

5 Software demonstrations

This section provides an overview of software demonstrations that were conducted in order to assess the feasibility of the recommendations and to explore the prospects for making use of OGC web services in terms of two NIEM IEPDs:

- Vessel position reports (Position IEPD v3.2)
- DHS RFI (RFI IEPD v2.0)

The demonstrations were predicated on the premise that incorporating the content of NIEM exchange messages within GML feature representations enables the use of a variety of OGC web services, many of which operate on GML data. The data transformation may be performed by client-side or server-side software components; both styles were demonstrated.

Web Feature Server (WFS v2) implementations figured prominently in the demonstration. Incoming NIEM message entities were transformed into GML features (see Table 2) and inserted into a WFS; the features were subsequently retrieved for display by other visualization components.

Table 2: NIEM data in GML

IEPD	NIEM message entity	GML feature
MDA Position and Tracks IEPD 3.2	pos:Message	ns1:VesselTrack
DHS RFI 2.0	rfi:RFIRequest_DHS-SPS	cp:niemrfipoly

5.1 Vessel position exchange

The Position IEPD defines an XML Schema for exchanging vessel position reports; it depends on the Maritime Domain Awareness (MDA) EIEM, which in turn draws on the NIEM Maritime domain. In keeping with recommendation 3.2.1 (*Convert NIEM exchange messages to GML feature* types), a GML application schema was defined to map the content of an exchange message to a GML feature. The application schema appears in Appendix B, and a sample VesselTrack instance is shown in Listing 12.

Listing 12: VesselTrack feature instance

```
<tns:VesselTrack gml:id="IM9304045"
  xmlns:tns="http://example.org/maritime"
  xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
  xmlns:m="http://niem.gov/niem/domains/maritime/2.1"
  xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <tns:exchangeMetadata>
    <tns:PositionMessageMetadata>
      <nc:DocumentCreationDate>
        <nc:Date>2013-06-28</nc:Date>
      </nc:DocumentCreationDate>
      <nc:DocumentExpirationDate>
        <nc:Date>2013-06-29</nc:Date>
      </nc:DocumentExpirationDate>
      <nc:DocumentCreator>
        <nc:EntityOrganization>
          <nc:OrganizationDescriptionText>Example
Organization</nc:OrganizationDescriptionText>
          <nc:OrganizationName>Example
Organization</nc:OrganizationName>
        </nc:EntityOrganization>
      </nc:DocumentCreator>
    </tns:PositionMessageMetadata>
  </tns:exchangeMetadata>
  <tns:vesselInfo>
    <mda:Vessel>
      <m:VesselAugmentation>
        <m:VesselCallSignText>C6TX6</m:VesselCallSignText>
        <m:VesselHullNumberText>667</m:VesselHullNumberText>
```

```
        <m:VesselIMONumberText>IM9304045</m:VesselIMONumberText>
        <m:VesselMMSIText>311827000</m:VesselMMSIText>
        <m:VesselName>Norwegian Jewel</m:VesselName>
        <m:VesselNationalFlagISO3166Alpha3Code>BHS
        </m:VesselNationalFlagISO3166Alpha3Code>
        <m:VesselOwner>
          <nc:EntityOrganization>
            <nc:OrganizationName>
            Norwegian Cruise Line
            </nc:OrganizationName>
          </nc:EntityOrganization>
        </m:VesselOwner>
        <m:VesselSCONUMText>0000001</m:VesselSCONUMText>
      </m:VesselAugmentation>
    </mda:Vessel>
  </tns:vesselInfo>
  <tns:track owns="true">
    <mda:Position>
      <m:LocationPoint>
        <gml:Point gml:id="p1"
          srsName="http://www.opengis.net/def/crs/EPSG/0/4326">
          <gml:pos>48.5632 -125.3683</gml:pos>
        </gml:Point>
      </m:LocationPoint>
      <mda:PositionSpeedMeasure>
        <nc:MeasureText>21.1</nc:MeasureText>
        <nc:SpeedUnitCode>kt</nc:SpeedUnitCode>
      </mda:PositionSpeedMeasure>
      <mda:PositionCourseMeasure>
        <nc:MeasureText>120</nc:MeasureText>
        <m:AngleUnitText>deg</m:AngleUnitText>
      </mda:PositionCourseMeasure>
      <mda:PositionHeadingMeasure>
        <nc:MeasureText>119</nc:MeasureText>
        <m:AngleUnitText>deg</m:AngleUnitText>
      </mda:PositionHeadingMeasure>
      <mda:PositionDateTime>
        <nc:DateTime>2013-06-28T15:46:00Z</nc:DateTime>
      </mda:PositionDateTime>
    </mda:Position>
  </tns:track>
</tns:VesselTrack>
```

The exchange message entity is transformed into a VesselTrack feature instance using an XSLT stylesheet (see Appendix C). In the resulting feature the position elements will be sorted in reverse chronological order (latest position first); however this ordering may not be preserved when the feature is inserted into a WFS.

An overview of the component architecture is presented in Figure 7.

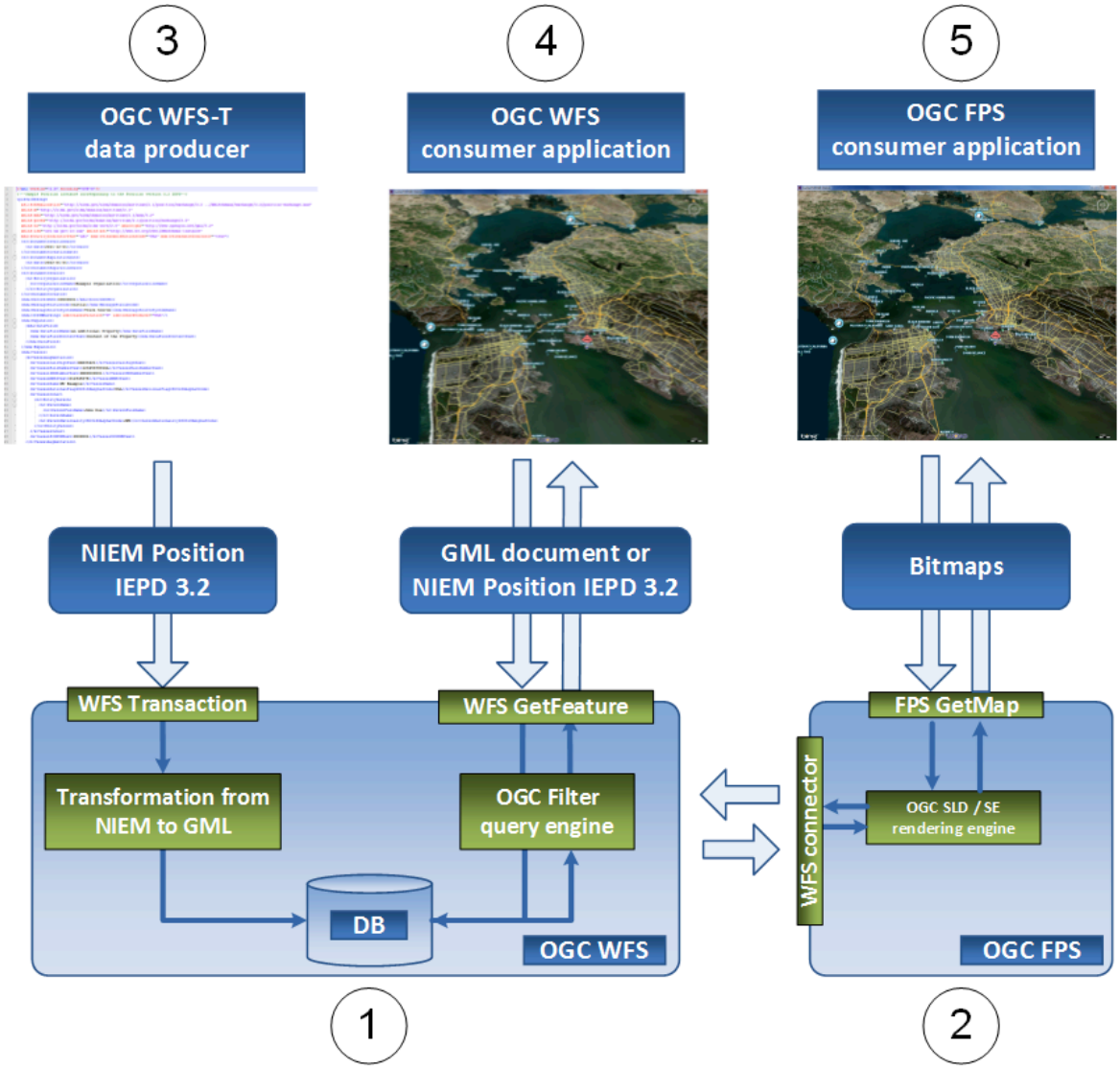

Figure 7: Component architecture for vessel tracking demo

There are five major components provided by Luciad:

1. An OGC WFS-T service that accepts vessel data in the NIEM Position IEPD 3.2 format and transforms it to the GML application schema based on that IEPD. Afterwards, clients can query the vessel data as GML features, with data content configurable by the client through OGC Filter. Additionally, clients can also retrieve the data back in the NIEM Position IEPD 3.2 format.
2. An OGC FPS service that disseminates the vessel data as maps, with styling and data content configurable by the client through OGC SLD / SE. The data content itself is retrieved from the OGC WFS-T service.
3. A data producer client that adds new vessel data in the NIEM Position IEPD 3.2 format to the WFS-T service.
4. A data consumer client that connects to the OGC WFS-T and consumes the vessel data in the NIEM Position IEPD 3.2 format.

5. A data consumer client that connects to the OGC FPS and consumes the vessel data as maps.

Vessel position data were provided by eXactEarth as GML data that conform to a vendor-specific application schema. The source data (LVI elements) were transformed to NIEM position reports as indicated by the mapping rules shown in Table 3.

Table 3: Mapping eXactEarth LVI data to NIEM vessel position

Latest Vessel Information (LVI)	NIEM Vessel Position
callsign	Vessel/VesselAugmentation/VesselCallSignText
imo	Vessel/VesselAugmentation/VesselIMONumberText
mmsi	Vessel/VesselAugmentation/VesselMSSIText
vessel_name	Vessel/VesselAugmentation/VesselName
vessel_type	Vessel/VesselAugmentation/VesselCategory
vessel_type_cargo	Vessel/VesselAugmentation/VesselCargoCategory
draught	Vessel/VesselAugmentation/VesselDraftLoadedMeasure
length	Vessel/VesselAugmantation/VesselOverallLengthMeasure
flag_country / flag_code	Vessel/VesselAugmentation/VesselNationalFlag (ISO 3166)
destination	*No mapping defined*
position	Position/LocationPoint
sog	Position/PositionSpeedMeasure/MeasureText, Position/PositionSpeedMeasure/SpeedUnitCode (kt)
cog	Position/PositionCourseMeasure/MeasureText, Position/PositionCourseMeasure/AngleUnitText (deg)
rot	*No mapping defined*
heading	Position/PositionHeadingMeasure/MeasureText,

Latest Vessel Information (LVI)	NIEM Vessel Position
	Position/PositionHeadingMeasure/AngleUnitText (deg)
nav_status	Position/PositionNavigationStatus/StatusText
ts_pos_utc	Position/PositionDateTime/DateTime (ISO 8601 notation)
ts_static_utc	*No mapping defined*
eta	*No mapping defined*

To add the data to the WFS-T, the NIEM position reports were wrapped in WFS Transaction requests. An example request is shown in Listing 13.

Listing 13: WFS request to insert a NIEM position report

```
<wfs:Transaction service="WFS" version="2.0.0"
    xmlns:wfs="http://www.opengis.net/wfs/2.0"
    xmlns:posex=
"http://niem.gov/niem/domains/maritime/2.1/position/exchange/3.2"
    xmlns:nc="http://niem.gov/niem/niem-core/2.0"
    xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
    xmlns:gml="http://www.opengis.net/gml/3.2"
    xmlns:m="http://niem.gov/niem/domains/maritime/2.1">
  <wfs:Insert>
    <posex:Message>
      <nc:DocumentCreationDate>
        <nc:DateTime>2013-08-07T08:48:20.508+02:00</nc:DateTime>
      </nc:DocumentCreationDate>
      <nc:DocumentExpirationDate>
        <nc:DateTime>2013-08-07T08:48:20.508+02:00</nc:DateTime>
      </nc:DocumentExpirationDate>
      <nc:DocumentCreator>
        <nc:EntityOrganization>
          <nc:OrganizationName>eXactEarth</nc:OrganizationName>
        </nc:EntityOrganization>
        <nc:EntityOrganization>
          <nc:OrganizationName>Luciad</nc:OrganizationName>
        </nc:EntityOrganization>
      </nc:DocumentCreator>
      <mda:Vessel>
        <m:VesselAugmentation>
          <m:VesselCallSignText>WDE5489</m:VesselCallSignText>
          <m:VesselCategoryText>HSC</m:VesselCategoryText>
          <m:VesselDraftLoadedMeasure>
            <nc:MeasureText>2.0</nc:MeasureText>
            <nc:MeasureUnitText>meter</nc:MeasureUnitText>
```

```
          </m:VesselDraftLoadedMeasure>
          <m:VesselIMONumberText>0</m:VesselIMONumberText>
          <m:VesselMMSIText>367367470</m:VesselMMSIText>
          <m:VesselName>ASTERIA</m:VesselName>
          <m:VesselOverallLengthMeasure>
            <nc:MeasureText>33</nc:MeasureText>
            <nc:MeasureUnitText>meter</nc:MeasureUnitText>
          </m:VesselOverallLengthMeasure>
        </m:VesselAugmentation>
      </mda:Vessel>
      <mda:Position>
        <m:LocationPoint>
          <gml:Point srsName="EPSG:4326">
            <gml:pos>-70.181525 42.04882</gml:pos>
          </gml:Point>
        </m:LocationPoint>
        <mda:PositionSpeedMeasure>
          <nc:MeasureText>2.7</nc:MeasureText>
          <nc:SpeedUnitCode>kt</nc:SpeedUnitCode>
        </mda:PositionSpeedMeasure>
        <mda:PositionCourseMeasure>
          <nc:MeasureText>129.1</nc:MeasureText>
          <m:AngleUnitText>deg</m:AngleUnitText>
        </mda:PositionCourseMeasure>
        <mda:PositionHeadingMeasure>
          <nc:MeasureText>291.0</nc:MeasureText>
          <m:AngleUnitText>deg</m:AngleUnitText>
        </mda:PositionHeadingMeasure>
        <mda:PositionNavigationStatus>
          <nc:StatusText>Under Way Using Engine</nc:StatusText>
        </mda:PositionNavigationStatus>
        <mda:PositionDateTime>
          <nc:DateTime>2013-08-07T08:48:20.509+02:00</nc:DateTime>
        </mda:PositionDateTime>
      </mda:Position>
    </posex:Message>
  </wfs:Insert>
</wfs:Transaction>
```

To facilitate this process, Luciad provided a data loader application that is able to scan a directory for NIEM position reports and publish them automatically via WFS Transaction requests to a WFS-T. A screenshot of this application is shown in Figure 8.

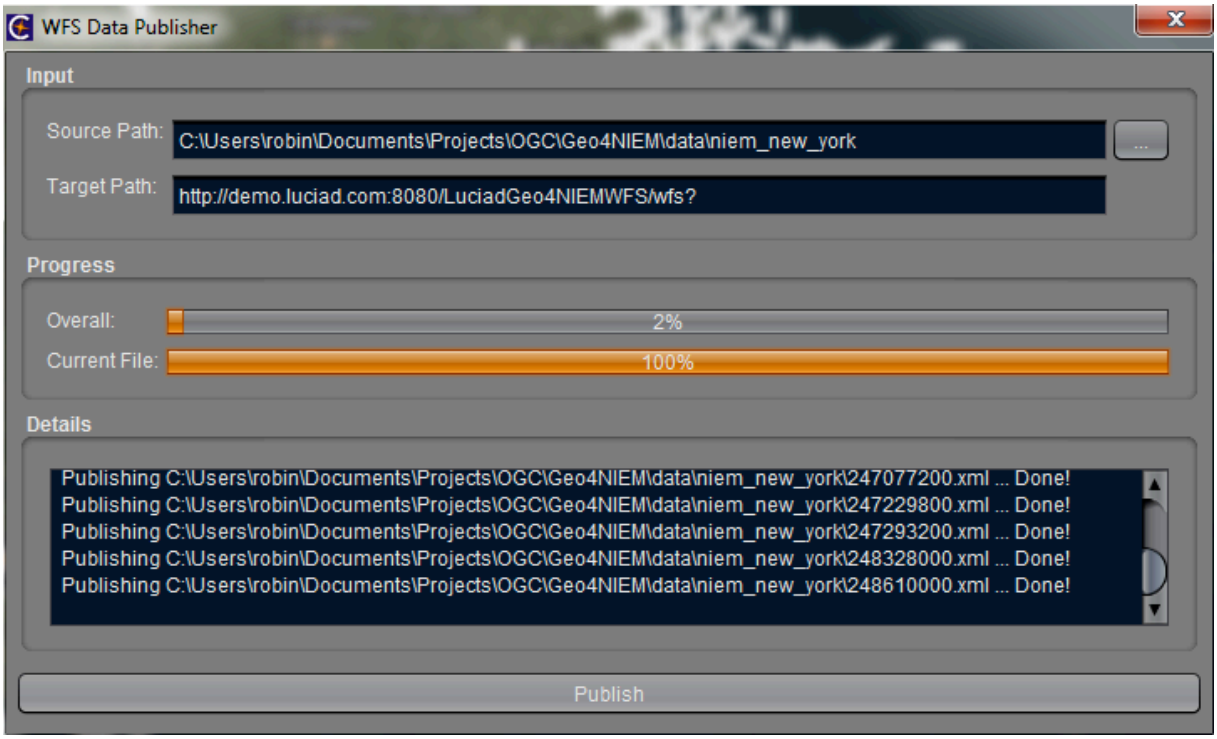

Figure 8: Luciad WFS data publisher

After the transformation in the WFS-T to GML features, the data can be queried and retrieved by WFS clients. By using OGC filter expressions clients can define the desired data content. Listing 14 shows an example of a WFS vessel feature request, using an OGC filter expression to specify that only vessels categorized as a 'Tanker' are of interest.

Listing 14: WFS request to retrieve tankers

```
<wfs:GetFeature
    xmlns:wfs="http://www.opengis.net/wfs/2.0"
    xmlns:fes="http://www.opengis.net/fes/2.0"
    service="WFS" version="2.0.0">
  <wfs:Query typeNames="VesselTrack">
    <fes:Filter>
      <fes:PropertyIsEqualTo>
        <fes:ValueReference xmlns:tns="http://example.org/maritime"
xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
xmlns:m="http://niem.gov/niem/domains/maritime/2.1">tns:vesselInfo/m
da:Vessel/m:VesselAugmentation/m:VesselCategoryText</fes:ValueRefere
nce>
        <fes:Literal>Tanker</fes:Literal>
      </fes:PropertyIsEqualTo>
    </fes:Filter>
  </wfs:Query>
</wfs:GetFeature>
```

The data can also be retrieved back in the NIEM Position IEPD 3.2 format, involving a reverse transformation from the GML feature to a NIEM position report; this demonstrates

that a full-round trip from NIEM position reports to GML features and back to NIEM is possible, without any data loss.

5.2 RFI exchange

The Request for Information (RFI) IEPD defines an XML Schema through which intelligence analysts and operations managers can request and receive relevant information for processing and analysis. In keeping with recommendation 3.2.1 (*Convert NIEM exchange messages to GML feature* types), sample RFI XML instances were developed by The Carbon Project, converted to GML features via an RFI Loader component and WFS Transactions, and made available to access and update from desktop and mobile applications via WFS operations.

The RFI component architecture fulfills three main aims:

1. To ingest XML RFI IEP samples into a cloud-based platform implementing OGC Web Services (OWS) Web Feature Service (WFS) and Geography Markup Language (GML) standards.
2. To export NIEM objects across a WFS interface as GML in response to WFS client GetFeature requests.
3. To update NIEM objects encapsulated as feature members in GML feature collections using a mobile client and WFS Insert, Update and Delete operations.

An overview of the component architecture for NIEM RFI XML Demo is presented in Figure 9. There are five major components provided by the Carbon Project:

- An RFI Loader which converts RFI XML to WFS Transactions.
- A cloud-based OGC WFS-T service that accepts NIEM RFI XML format and disseminates it in the GML format, with data content accessible by clients through OGC WFS Operations and OGC Filter Encodings.
- A desktop client
- A mobile application client capable of WFS Transactions.
- A mobile application management service that can accept updates to NIEM objects in a GML feature collection and sends them to an OGC WFS-T.

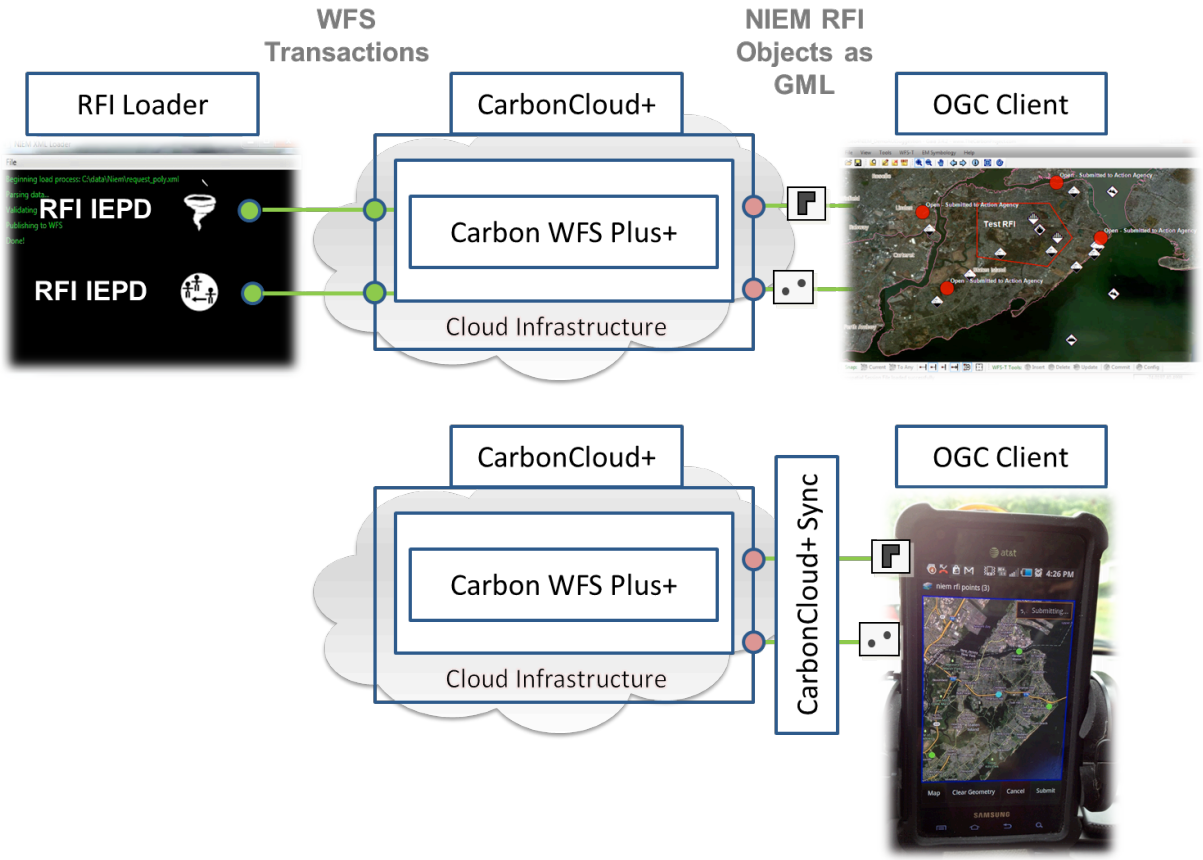

Figure 9: Component architecture for RFI demo

A sample WFS Transaction request to insert an RFI XML instance, sent by The Carbon Project RFI Loader to the Carbon WFS Plus+ service, is shown in Listing 15.

Listing 15: Request to insert niemrfipoly feature

```
<wfs:Transaction xmlns:wfs="http://www.opengis.net/wfs"
service="WFS" version="1.1.0"
xmlns:cp="http://thecarbonproject.com">
  <wfs:Insert>
    <cp:niemrfipoly>
      <cp:RFINumber>Test_Polygon_1</cp:RFINumber>
      <cp:SubmissionMethod>Email</cp:SubmissionMethod>
      <cp:USPersonDataIndicator>False</cp:USPersonDataIndicator>
      <cp:PIIDataIndicator>False</cp:PIIDataIndicator>
      <cp:UrgentIndicator>False</cp:UrgentIndicator>
      <cp:UrgentJustification>N/A</cp:UrgentJustification>

<cp:IntendedRecipientGivenName>Mark</cp:IntendedRecipientGivenName>

<cp:IntendedRecipientSurName>Mattson</cp:IntendedRecipientSurName>
      <cp:RequestDetails>Test RFI</cp:RequestDetails>
      <cp:JustificationDetails>Test RFI</cp:JustificationDetails>
```

```
<cp:RequestClassification>UNCLASSIFIED</cp:RequestClassification>
      <cp:DesiredResponseClassification>
      UNCLASSIFIED
      </cp:DesiredResponseClassification>
      <cp:HighestAcceptableResponseClassification>
      UNCLASSIFIED
      </cp:HighestAcceptableResponseClassification>
      <cp:HighestAcceptableResponseClassification>
      UNCLASSIFIED
      </cp:HighestAcceptableResponseClassification>
      <cp:RequestorSurName>Mattson</cp:RequestorSurName>

<cp:RequestorEmail>mmattson@thecarbonproject.com</cp:RequestorEmail>
      <cp:RequestorTelephone>781-820-9069</cp:RequestorTelephone>

<cp:RequestorOrganizationCode>T1:State</cp:RequestorOrganizationCode
>
      <cp:IntendedResponseUse>N/A</cp:IntendedResponseUse>
      <cp:LESDataIndicator>False</cp:LESDataIndicator>
      <cp:HSECSINGivenName>Mark</cp:HSECSINGivenName>
      <cp:HSECSINSurName>Mattson</cp:HSECSINSurName>
      <cp:HSECSINTelephone>781-820-9069</cp:HSECSINTelephone>

<cp:HSECSINEmail>mmattson@thecarbonproject.com</cp:HSECSINEmail>
      <cp:RequestStatus>
      Open - Submitted to Action Agency
      </cp:RequestStatus>
      <cp:RequestTypeCode>Other</cp:RequestTypeCode>
      <cp:IncidentLocation>
        <gml:Surface xmlns:gml="http://www.opengis.net/gml"
srsName="EPSG:4326">
          <gml:patches>
            <gml:PolygonPatch>
              <gml:exterior>
                <gml:LinearRing>
                  <gml:posList count="6" srsDimension="2">-74.169818
40.628014 -74.1074278333333 40.628014 -74.0874988333333
40.5985518333333 -74.109161 40.573424 -74.169818 40.582089 -
74.169818 40.628014</gml:posList>
                </gml:LinearRing>
              </gml:exterior>
            </gml:PolygonPatch>
          </gml:patches>
        </gml:Surface>
      </cp:IncidentLocation>
    </cp:niemrfipoly>
  </wfs:Insert>
</wfs:Transaction>
```

We adopted the approach that the RFI request be incorporated within a GML feature instance. For example, the WFS operation described above inserts a new ‘niemrfipoly’ feature into the

WFS-T (with the niemrfipoly feature type derived from gml:AbstractFeatureType). When accessed by client applications, a feature collection is returned with a gml:featureMember containing a niemrfipoly element. In the niemrfipoly feature instance, the value of the <IncidentLocation> property is a gml:Surface geometry (see Listing 16).

Listing 16: Incident location represented as gml:Surface

```
<cp:IncidentLocation>
  <gml:Surface gml="http://www.opengis.net/gml/3.2"
srsName="urn:ogc:def:crs:EPSG::4326">
    <gml:patches>
      <gml:PolygonPatch>
        <gml:exterior>
          <gml:LinearRing>
            <gml:posList>40.628014 -74.169818 40.628014 -
74.1074278333333 40.5985518333333 -74.0874988333333 40.573424 -
74.109161 40.582089 -74.169818 40.628014 -74.169818</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:PolygonPatch>
    </gml:patches>
  </gml:Surface>
</cp:IncidentLocation>
```

Point feature types were also inserted, accessed and updated via WFS-T operations using a similar procedure and encodings.

In the sample RFI IEP used for ingest <IncidentLocation> was encoded using the LocationTypes <LocationTwoDimensionGeographicCoordinate> and <AreaPolygonGeographicCoordinate>. LocationTwoDimensionGeographicCoordinate and AreaPolygonGeographicCoordinate are NIEM elements used to represent point and polygon locations, and are part of the LocationType indicated for use in RFI. They were developed for testing RFI XML ingest by the project team. Sample RFIs with IncidentLocation information were not available.

5.3 Integrated scenario

5.3.1 Overview

An integrated scenario was developed for the purposes of demonstrating how NIEM IEPs (exchange messages) could be automatically transformed into GML features and subsequently accessed using several OGC web services. A Category 3 hurricane has made landfall in the vicinity of Staten Island NY, bringing major wind and water damage from powerful damaging waves and storm surge. Power and transportation infrastructures have been severely damaged, especially in the areas hardest hit including Staten Island. Emergency responders have been mobilized from all regional jurisdictions and the Federal government. During early stages of the response, activities to assess the damage and prioritize and coordinate the response are seeking additional information via the Department of Homeland Security Request for Information (RFI) capability. Executives and Emergency Operations Centers use the RFI requests and responses to monitor, coordinate and support the provision of necessary resources where and when needed.

To support the response, actions are being coordinated with Coast Guard and other regional authorities to identify and deploy appropriate maritime resources supported by the Vessel Position information exchange. This Vessel Position exchange is used to determine locations of accessible and operational harbor facilities; and vessels in the area that might be used to convey support, materials and other supplies to the region. For example, the authorities are seeking to determine the availability of fuel supplies and vessels, such as fuel barges, to transport these supplies. They also seek to identify suitable, functional and accessible fuel offloading facilities to store and dispense fuel to appropriate authorities and the public.

The following figure provides a view of the overall architecture showing integration of various software components and capabilities for the two IEP use cases: Request of Information (RFI) and MDA Vessel Position.

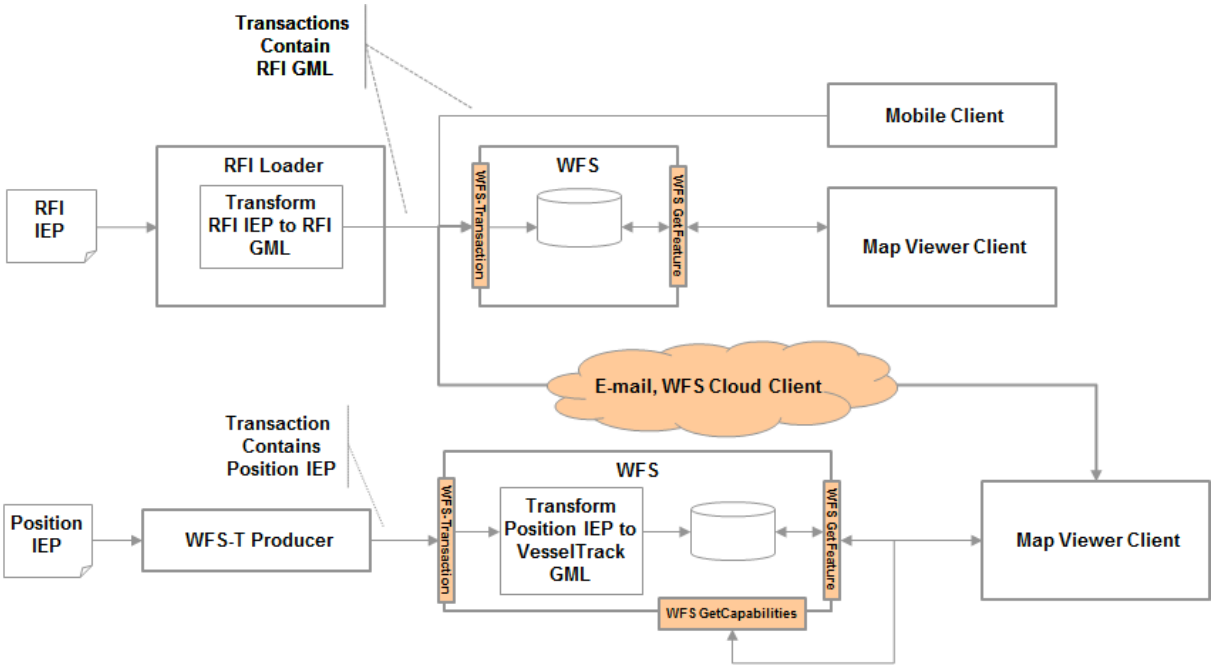

Figure 10: Demonstration architecture

5.3.2 RFI messages

The demonstration scenario posited a storm making landfall along the east coast of the United States. In this scenario, sample NIEM XML instances were converted to NIEM objects in GML using WFS Transactions and the components described in the previous section. An overview of the RFI scenario is presented in Figure 11, which shows a screenshot of the Gaia client used to display information about submitted (pending) RFI requests.

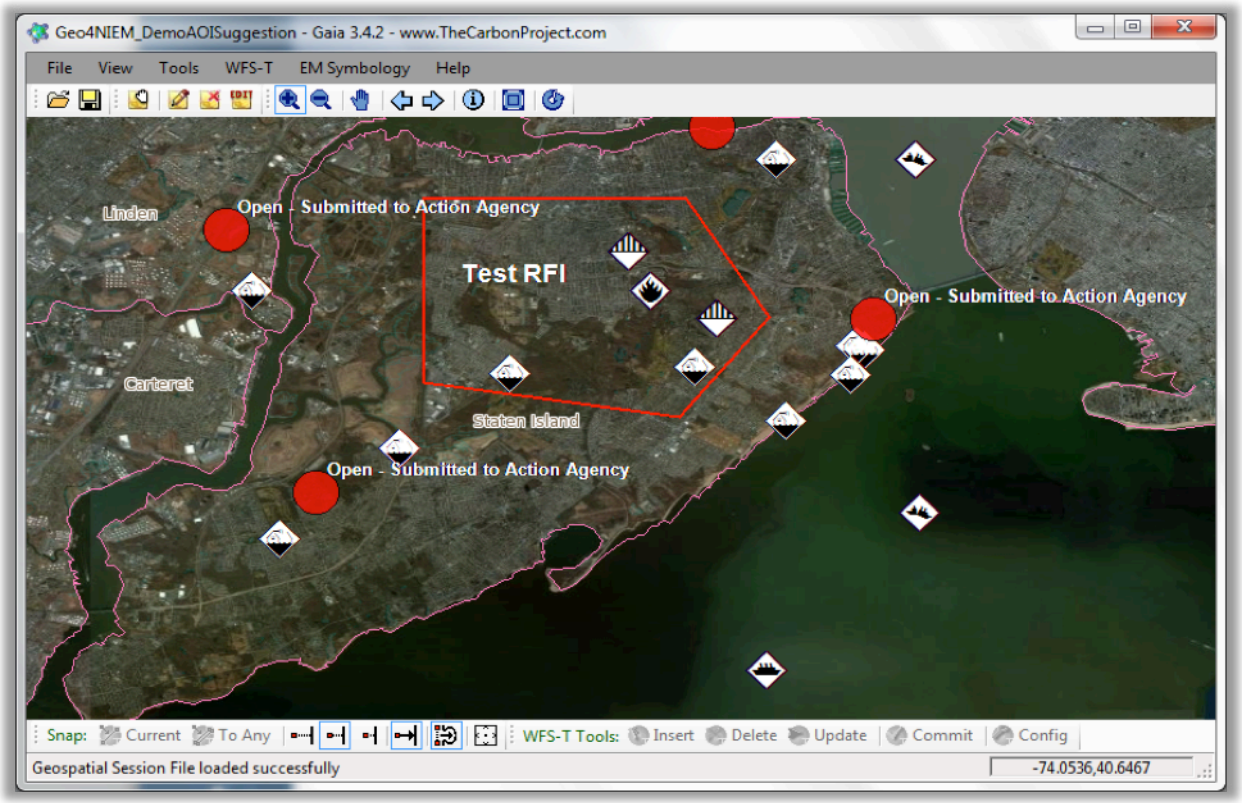

Figure 11: Demonstration scenario – Hurricane response around Staten Island NY

The GML feature data were used to construct a series of requests, such as 'What vessels are in the area that can supply fuel to Staten Island and where are they?' These request were made in the context of a simulated Fusion Center. The demonstration scenario also showed mobile update from simulated field users operating on Staten Island. Finally, the demonstration scenario received information on the locations of vessels as GML.

5.3.3 Vessel position reports

In the scenario, Luciad takes the role of the Coast Guard that can monitor and analyze vessel positions. In this role, Luciad receives an RFI from the Fusion Center, requiring information about vessels around New York that can bring fuel to Staten Island within 8 hours. To support the scenario and respond to this RFI, Luciad deployed a set of OGC services and a situational awareness client, capable of distributing and querying vessel traffic data. Section 5.1 describes the underlying component architecture and the data preparation in more detail.

At the start of the demonstration, an OGC WFS-T 2.0 service has been preloaded with vessel track data. The client connects to this service to query the data and starts preparing a vessel data set to respond to the RFI. To support this action, the client provides the user with a number of OGC Filter options to filter the data:

- A geospatial OGC Filter can be enabled to query vessel data for a particular area of interest.

- A vessel type OGC Filter can be enabled to query vessel data with a given value for the NIEM VesselCategoryText property; following the RFI, we have to search for vessels containing fuel, which is represented by the vessel type Tanker. Listing 14 in section 5.1 shows an example of a WFS request containing an OGC Filter for this vessel type.
- A travel duration (ETA) OGC Filter can be enabled to search for vessels that can arrive at a given location within a given amount of time. This filter relies on a custom OGC Filter function that has been added to the Luciad WFS-T. This function takes a position and calculates how long it will potentially take for a vessel to get to that position. A few sensible defaults about the typical speed of vessels are used to calculate this; it also uses a straight line distance measure for simplicity.

The user interface of the client allows entering a duration in hours. To specify the location, the client shows a predefined list of harbor positions that can be selected by the user. Queries for tankers located in the New York Harbor area were performed using the Luciad client (Figure 12). The screenshot shows the latest position of vessels having an approximate ETA within a given number of hours; the WFS query was implemented using a custom function.

Figure 12: Luciad situational awareness client showing vessel positions

By employing these OGC filter options, a vessel data set can be prepared that satisfies the RFI message. To be able to communicate this data and answer the RFI, the client application allows exporting the vessel data as a GML feature collection. The resulting data file adheres to the GML application schema based on the NIEM Position IEPD. The demonstration showed that the RFI requester can access and visualize this data as expected.

5.4 Outcomes

The demonstration highlighted methods by which standard WFS clients using NIEM objects encapsulated in GML feature representations were able to query, access, filter, visualize, update and manage the data without modification. The demonstration also highlighted techniques by which NIEM exchange messages can be converted to WFS transactions for ingest into WFS-T; this was accomplished using both client-side and server-side components. A demonstration video was also prepared and multiple live demonstrations were conducted.

6 Looking ahead

6.1 NIEM 3.0

NIEM 3.0 is expected to be released in fall 2013. The beta phase of the release cycle overlapped the Geo4NIEM project. While this project was centered on NIEM 2.1, the participants were mindful of the upcoming 3.0 release and took steps to ensure that the final recommendations remained relevant for NIEM 3.0.

This major revision of the NIEM framework emphasizes improved usability and schema simplification. The main areas addressed by the changes fell into three categories:

- Technical challenges—more than 200 known issues were resolved
- Organizational challenges
- New domains and requirements (e.g. biometrics, code lists)

Several technical enhancements will be introduced, including the following:

- Simplified structures
- Simplified annotations (appinfo elements replaced by attributes)
- Augmentation via substitution of elements at predefined, abstract extension points
- Support for local vocabularies

The adapter mechanism remains, but some of the finer points of how to define an adapter type have changed. Listing 17 shows how a GML geometry adapter is defined in NIEM 3.0; the changes are **highlighted in bold**.

Listing 17: Geometry adapter type in NIEM 3.0

```
<xsd:complexType name="PointType"
  appinfo:externalAdapterTypeIndicator="true">
  <xsd:annotation>
    <xsd:documentation>A 2D or 3D geometric
point.</xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="s:ObjectType">
      <xsd:sequence>
        <xsd:element ref="gml:Point" minOccurs="1"
maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
```

```
</xsd:complexType>
```

6.2 Future work

Over the course of the project a number of issues were raised that fell outside the immediate scope of the study but were identified as possible topics for future work:

- Investigate the use of information security markings, especially those defined by the *XML Data Encoding Specification for Information Security Markings* (IC-ISM Version 9).
- Consider adopting elements of the latest *Time-Space-Position Information* specification (TSPI 2.0) to express spatiotemporal location information.
- Explore mechanisms for attaching documents to RFI exchange messages; supporting information resources (e.g. reports, images) could be included as additional MIME entities within a multipart message.

Appendix A Schematron schema for GMLSF geometry elements

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron" id="geospatial"
  xml:lang="en" queryBinding="xslt2">

  <title>Schematron schema for NIEM geospatial adapters.</title>
  <ns uri="http://www.opengis.net/gml/3.2" prefix="gml"/>
  <p>This schema specifies constraints on GML geometry adapters in
accord with the GML Simple Features Profile v2.0 (OGC 10-100r3),
Level SF-0.</p>

  <phase id="SF-0">
    <active pattern="CurveSegment"/>
    <active pattern="SurfacePatch"/>
  </phase>

  <pattern id="CurveSegment">
    <title>Allowable Curve Segments</title>
    <p>Allowable curve segments include LineStringSegment, Arc,
Circle, and CircleByCenterPoint. The following are disallowed:
ArcByCenterPoint, ArcString, ArcStringByBulge, ArcByBulge,
CubicSpline, BSpline,
      Bezier, OffsetCurve, Clothoid, GeodesicString, Geodesic.</p>
    <rule context="gml:Curve/gml:segments">
      <report test="gml:ArcByCenterPoint">gml:ArcByCenterPoint is
not an allowed curve segment.</report>
      <report test="gml:ArcString">gml:ArcString is not an allowed
curve segment.</report>
      <report test="gml:ArcStringByBulge">gml:ArcStringByBulge is
not an allowed curve
        segment.</report>
      <report test="gml:ArcByBulge">gml:ArcByBulge is not an allowed
curve segment.</report>
      <report test="gml:CubicSpline">gml:CubicSpline is not an
allowed curve segment.</report>
      <report test="gml:BSpline">gml:BSpline is not an allowed curve
segment.</report>
      <report test="gml:Bezier">gml:Bezier is not an allowed curve
segment.</report>
      <report test="gml:OffsetCurve">gml:OffsetCurve is not an
allowed curve segment.</report>
      <report test="gml:Clothoid">gml:Clothoid is not an allowed
curve segment.</report>
      <report test="gml:GeodesicString">gml:GeodesicString is not an
allowed curve segment.</report>
      <report test="gml:Geodesic">gml:Geodesic is not an allowed
curve segment.</report>
    </rule>
  </pattern>
```

```
  <pattern id="SurfacePatch">
    <title>Allowable Surface Patches</title>
    <p>Only PolygonPatch is allowed. Triangle, Rectangle, Cone,
Cylinder, and Sphere are
      disallowed.</p>
    <rule context="gml:Surface/gml:patches">
      <report test="gml:Triangle">gml:Triangle is not an allowed
surface patch.</report>
      <report test="gml:Rectangle">gml:Rectangle is not an allowed
surface patch.</report>
      <report test="gml:Cone">gml:Cone is not an allowed surface
patch.</report>
      <report test="gml:Cylinder">gml:Cylinder is not an allowed
surface patch.</report>
      <report test="gml:Sphere">gml:Sphere is not an allowed surface
patch.</report>
    </rule>
  </pattern>

  <diagnostics>
    <diagnostic id="geometry" xml:lang="en">Geometry context is
<value-of select="local-name(..)"
        />[@gml:id='<value-of select="../@gml:id"/>']</diagnostic>
  </diagnostics>
</schema>
```

Appendix B GML application schema for VesselTrack feature

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://example.org/maritime"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:gmd="http://www.isotc211.org/2005/gmd"
  xmlns:tns="http://example.org/maritime"
  xmlns:posx="http://niem.gov/niem/domains/maritime/2.1/position/
exchange/3.2"
  xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
  xmlns:nc="http://niem.gov/niem/niem-core/2.0"
elementFormDefault="qualified">

  <xsd:import namespace="http://www.opengis.net/gml/3.2"
    schemaLocation="http://schemas.opengis.net/gml/3.2.1/gml.xsd"/>
  <xsd:import
namespace="http://niem.gov/niem/domains/maritime/2.1/position/exchan
ge/3.2"
    schemaLocation="./position-
3.2.iepd/XMLSchemas/exchange/3.2/position-exchange.xsd"/>
  <xsd:import
namespace="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
    schemaLocation="../../extension/mda/3.2/mda.xsd"/>
  <xsd:import namespace="http://niem.gov/niem/niem-core/2.0"

schemaLocation="./position3.2.iepd/XMLSchemas/extension/mda/3.2/mda.
xsd"/>
  <xsd:element name="VesselTrack" type="tns:VesselTrackType"
               substitutionGroup="gml:AbstractFeature">
    <xsd:annotation>
      <xsd:documentation xml:lang="en">Encapsulates information
about a vessel track (course over ground).</xsd:documentation>
    </xsd:annotation>
  </xsd:element>
  <xsd:complexType name="VesselTrackType">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractFeatureType">
        <xsd:sequence>
          <xsd:element ref="tns:exchangeMetadata" minOccurs="0"/>
          <xsd:element ref="tns:vesselInfo"/>
          <xsd:element ref="tns:track"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="vesselInfo" type="tns:VesselPropertyType"/>
  <xsd:complexType name="VesselPropertyType">
    <xsd:sequence minOccurs="0">
      <xsd:element ref="mda:Vessel"/>
```

```
    </xsd:sequence>
    <xsd:attributeGroup ref="gml:AssociationAttributeGroup" />
  </xsd:complexType>
  <xsd:element name="track" type="tns:PositionArrayPropertyType"/>
  <xsd:complexType name="PositionArrayPropertyType">
    <xsd:annotation>
      <xsd:documentation>A sequence of time-stamped mda:Position
elements indicating the vessel's course over
ground.</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="mda:Position" />
    </xsd:sequence>
    <xsd:attributeGroup ref="gml:OwnershipAttributeGroup"/>
  </xsd:complexType>
  <xsd:element name="exchangeMetadata"
               type="tns:ExchangeMetadataPropertyType"/>
  <xsd:complexType name="ExchangeMetadataPropertyType">
    <xsd:complexContent>
      <xsd:extension base="gml:AbstractMetadataPropertyType">
        <xsd:sequence>
          <xsd:element ref="tns:PositionMessageMetadata"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="PositionMessageMetadata"
               type="tns:PositionMessageMetadataType"/>
  <xsd:complexType name="PositionMessageMetadataType">
    <xsd:sequence>
      <xsd:element ref="nc:DocumentCreationDate" minOccurs="0"/>
      <xsd:element ref="nc:DocumentExpirationDate" minOccurs="0"/>
      <xsd:element ref="nc:DocumentCreator" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

Appendix C XSLT: Transform position report to VesselTrack feature

```
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:tns="http://example.org/maritime"
  xmlns:gml="http://www.opengis.net/gml/3.2"
  xmlns:m="http://niem.gov/niem/domains/maritime/2.1"
  xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
  xmlns:nc="http://niem.gov/niem/niem-core/2.0"
  xmlns:posx="http://niem.gov/niem/domains/maritime/2.1/position/
exchange/3.2"
  exclude-result-prefixes="posx">

  <db:abstract xmlns:db="http://docbook.org/ns/docbook">
    <db:para>Transforms a NIEM vessel position message entity
(defined in the Position IEPD v3.2) to a VesselTrack feature
instance.</db:para>
  </db:abstract>

  <xsl:param name="id"
select="string(//m:VesselAugmentation/m:VesselIMONumberText[1])" />
  <xsl:param name="srsName"
select="'http://www.opengis.net/def/crs/EPSG/0/4326'" />

  <xsl:output method="xml" indent="yes" encoding="UTF-8" />
  <xsl:strip-space elements="*" />

  <xsl:template match="posx:Message">
    <tns:VesselTrack gml:id="{$id}">
      <tns:vesselInfo>
        <xsl:copy-of select="mda:Vessel"/>
      </tns:vesselInfo>
      <tns:track owns="true">
        <xsl:for-each select="mda:Position">
          <!-- sort xsd:dateTime values  -->
          <xsl:sort select="mda:PositionDateTime/nc:DateTime"
                    order="descending"/>
          <xsl:copy-of select="."/>
        </xsl:for-each>
      </tns:track>
    </tns:VesselTrack>
  </xsl:template>

</xsl:transform>
```

Appendix D Fact sheet

Scope

The principal aim of the Geo4NIEM project was to enhance interoperability with respect to the handling of geospatial information based on OGC standards and NIEM-conformant information exchanges. Much of the work was focused on the GML (ISO 19136) data exchange standard and the mechanisms by which GML and NIEM data could be intermingled. Software demonstrations were also conducted to assess the feasibility of the various technical approaches and to explore the prospects for making use of fundamental OGC web services for search, retrieval, and visualization. The demonstration scenario focused on data exchanges specified by two IEPDs: DHS RFI (v2.0) and MDA Vessel Position (v3.2).

Recommendations

It was recognized early in the project that conforming to both NIEM 2.1 and GML 3.2 at the same time was not possible due to contradictory constraints that frustrated efforts to achieve a "super-compliant" integration. However by adopting one of these standards as the principal framework, a set of recommendations emerged covering various aspects of the work undertaken by project participants. The recommendations presented below are informed by two guiding principles:

1. The existing adapter mechanism is the preferred way to import GML content into NIEM-conformant data structures.
2. Incorporating NIEM message content within GML feature representations greatly expands the possibilities for operationalizing NIEM information exchanges.

GML adapters in NIEM

- Support the GML Simple Features profile to provide an essential set of 2D geometry types suitable for use in any NIEM information exchange.
- Add a generic feature adapter that may contain any GML feature representation (that substitutes for the gml:AbstractFeature element).
- Add a general-purpose geometry adapter that accepts any GML geometry representation; this emulates the gml:GeometryPropertyType permitted by the GML Simple Features (GMLSF) profile.
- Remove adapters for curve segments that can only appear as elements of gml:Curve[gml:segments]: gml:Arc, gml:ArcByCenterPoint, gml:Circle, gml:CircleByCenterPoint (Note: these are not geometry types).
- Add Schematron (ISO 19757-3) constraints to restrict allowable curve segments and surface patches in accord with the GML Simple Features Profile v2.0 (OGC 10-100r3), Level SF-0
- Deprecate specific GML adapter elements and encourage IEPD developers to declare more meaningful, domain-specific elements.
- Add a general-purpose geometry adapter to nc:LocationType

NIEM in GML	▪ Map elements of message entities to a GML feature type to enable the use of OGC web services for search, retrieval, and portrayal. ▪ In developing a GML application schema strive to minimize the number of mapping rules so as to not obliterate the original NIEM data structures. ▪ Extract message-level metadata into a separate metadata entity (which can then be accessed from WFS or CSW-based catalogue services).
OGC web services	▪ Extend OGC WFS v2 implementations to accept NIEM message entities (where inputFormat is the target namespace of the NIEM message element). ▪ Use a registry service (CSW-ebRIM) to manage IEPD artifacts such as schemas, code lists, and supporting documentation.

OGC web service enablement

Perhaps the most significant advantage that arises from encapsulating the content of a NIEM exchange message within a GML feature is the capability of utilizing various OGC web services for searching, retrieving, and visualizing data. Most of the fundamental OGC web services can operate on GML feature representations, so taking this step enables the use of a multitude of open-source and commercial software products that implement OGC specifications such as the following:

- **Web Feature Service** (WFS): Provides an API for operating primarily on GML feature representation; features may be searched, retrieved, inserted, updated, and deleted.
- **Web Map/Feature Portrayal Service** (WMS/FPS): Provides an API for producing graphical representations (maps) of spatial data; a Feature Portrayal Service (FPS) is a specialized WMS that supports user-defined styles and is not tightly coupled to particular data sources.
- **Registry Service** (CSW-ebIM): A specialized catalogue service that provides a web-based API for storing and retrieving shared information resources such as schemas, code lists, and dataset descriptions.