

Open Geospatial Consortium

Publication Date: 2013-10-25

Approval Date: 2013-09-27

Posted Date: 2013-09-04

Reference number of this document: OGC 13-080r3

Reference URL for this document: <http://www.opengis.net/doc/PER/MOGIE>

Category: OGC Public Engineering Report

Editor(s): Frank Klucznik
Matthew Weber
Robin Houtmeyers
Roger Brackin

OGC[®] Military Operations Geospatial Interoperability Experiment (MOGIE)

Copyright © 2013 Open Geospatial Consortium.

To obtain additional rights of use, visit <http://www.opengeospatial.org/legal/>.

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.

Document type: OGC[®] Engineering Report
Document subtype: NA
Document stage: Approved for public release
Document language: English

Abstract

There are many commercially-available geospatial tools that exploit OGC standards. Using NIEM messages, this experiment demonstrated that GML content can be embedded in NIEM conformant XML and be exploited by commercial and open source tools without loss of precision (e.g., right number of bits) or accuracy (e.g., physical location on a map). Embedding GML in NIEM conformant XML was accomplished in MOGIE using the NIEM adapter.

Preface

This Engineering Report (ER) summarizes the results of the of the Open Geospatial Consortium (OGC) MilOps Geospatial Interoperability Experiment (MOGIE).

The purpose of the MOGIE was to test the hypothesis that using the National Information Exchange Model (NIEM) v2.1 and v3.0 technical concepts does not introduce any changes in data related to accuracy and precision. MOGIE also tested the theory that data may be read by a client without a priori access and knowledge of the data to demonstrate sharing data via OGC web services provides broader community interoperability than data shared without OGC services.

MOGIE conducted five experiments in two different scenarios (e.g., land environment and maritime environment). The experiments included:

1. Employ “GML Validator” currently being developed in OWS-9 as appropriate to determine compliance of GML in a MilOps exchange to GML Encoding Specifications, WFS, WMS, etc.
2. Extract IEP content including geospatial data that includes GML and transform it into an OGC Standard format with military symbology as appropriate, and then display the data on a client.
3. Extract geospatial data that includes GML content and add additional NIEM attributes and then display the data on a client.
4. Demonstrate no loss of precision or accuracy when transforming GML content embedded in a NIEM conformant IEP.
5. Demonstrate implementation of the latest draft version of MIL-STD-2525D.

Details in this report show that MOGIE demonstrated the following findings:

1. GML can be embedded in a NIEM conformant data exchange and transformed into an OGC conformant format.
2. GML and other data was transmitted in NIEM conformant XML, stored in a database, and delivered by OGC conformant web services in WFS/WMS for display on multiple client software and hardware.
3. Data can be read by clients without a priori access/knowledge of the data.
4. NIEM XML does not change numerical data being exchange, and therefore had no impact on the accuracy and precision of position data exchanged in MOGIE.
5. Envitia and Luciad independently demonstrated implementation of the Jan 2013 draft of MIL-STD-2525D in their client applications, and reported the changes in 2525D were an improvement over previous versions of the standard.
6. Implementing NIEM XML did not require any specialty skills; commodity software development and XML skills provided a sufficient base to learn and use NIEM from training and materials available publicly online.
7. The bit level storage requirements and binary structure prescribed by IEEE 754 Standard for floating-point arithmetic introduced limits on accuracy and precision
8. Minor limitations of the OGC validator tool added some time to experiment #1’s completion. However, the accessibility and flexibility of the validator’s architecture provided the means to accomplish the validation.
9. An evaluation of the OGC GML validation tool produced three recommended changes:

OGC 13-080r3

- a. Provide Authentication Support
 - b. Provide Better Internal Error Reporting
 - c. Provide additional Source Code Comments in TEAM Engine
10. Supporting graphics, not the data, set a practical limit of accuracy that can be distinguished on a graphical display.

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("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.

TABLE OF CONTENTS

1 INTRODUCTION1

1.1 SCOPE 1

1.2 DOCUMENT CONTRIBUTOR CONTACT POINTS 1

1.3 REVISION HISTORY 1

1.4 FUTURE WORK 1

1.5 FORWARD 1

2 REFERENCES2

2.1 DoD DATA MODEL SOURCES 2

2.2 OGC DOCUMENTS 2

3 TERMS AND DEFINITIONS2

4 ORGANIZATIONS IN MOGIE3

4.1 INITIATOR ORGANIZATIONS 3

4.2 INITIATIVE TEAM 3

4.3 COMPLETE LIST OF PARTICIPATING ORGANIZATIONS 3

5 SCHEDULE4

6 EXPERIMENT OVERVIEW5

6.1 PURPOSE 5

6.2 BACKGROUND 5

6.3 DESCRIPTION 6

6.4 USE CASES 7

6.5 EXPERIMENTS 7

6.6 METHODOLOGY AND APPROACH 7

7 SCENARIO OVERVIEW16

7.1 LAND SCENARIO 16

7.2 MARITIME SCENARIO 18

8 COMPONENTS21

8.1 DATA SOURCE AND DATA STORAGE 21

8.2 DATA DELIVERY SERVICES 29

8.3 DATA CONSUMPTION 34

8.4 SUPPORTING SERVICES 45

9 FINDINGS47

APPENDIX A: GEOLOCATION ACCURACY AND PRECISION IN NIEM49

APPENDIX B: EXPERIMENT DATA FLOW DESCRIPTION54

APPENDIX C: OGC GML VALIDATION TOOL56

APPENDIX D: DOD NIEM ADOPTION MEMO60

APPENDIX E: BASE MARITIME CSV DATA62

APPENDIX F: LAND SCENARIO BASE DATA63

APPENDIX G: DATA ANALYSIS RESULTS65

APPENDIX H: VESSEL POSITION REPORT – NIEM XML66

APPENDIX I: VESSEL POSITION REPORT – OWS EXPORT GML XML67

APPENDIX J: POSITION DATA PROCESSING68

APPENDIX K EMBEDDING GML BEST PRACTICE PAPER73

FIGURES

Figure 1: Detailed Execution Timeline4

Figure 2: Base OWS Architecture8

Figure 3: Base MOGIE Architecture9

Figure 4: Land scenario data collection points11

Figure 5: Maritime scenario data collection points11

Figure 6: Land scenario data capture methodology11

Figure 7: Maritime scenario data capture methodology12

Figure 8: Data Comparison Methodology13

Figure 9: MOGIE Land Scenario16

Figure 10: MOGIE Land Scenario Technical Architecture17

Figure 11: Maritime Terrorist Scenario19

Figure 12: Maritime Scenario Technical Architecture20

Figure 13: Position Report Database Attributes22

Figure 14: Observation Report Database Attributes22

Figure 15: Call for Fire Database Attributes23

Figure 16: Latest Vessel Position Database Attributes24

Figure 17: LVI and HT30 Database Attributes25

Figure 18: GML PointType Definition27

Figure 20 GML DirectPositionType27

Figure 21 GML doubleList27

Figure 21: Data Retrieval Process from WFS30

Figure 22: Process to retrieve data from the WFS31

Figure 23: Process of retrieving KML via WMS33

Figure 24: Participant Contribution to MOGIE34

Figure 25 - Envitia Exploitation Sub-system35

Figure 26 – Envitia Mediating WMS Architecture36

Figure 27 - Envitia Portal showing MOGIE data37

Figure 28 - Envitia MapLink Pro Android Application38

Figure 29: Land scenario in ArcGIS38

Figure 30: Maritime scenario in FalconView40

Figure 31: Playback Dialog in FalconView41

Figure 32: Military and civilian vessels in the Luciad Desktop client42

Figure 33: Vessel analysis and querying its historical track (shown by the white line)43

Figure 34: Accessing MOGIE land scenario data and making map annotations on the Luciad Mobile client through OGC GeoPackage44

Figure 35: NOAA ENC data accessed in the Luciad Desktop client45

Figure 36: Source code for processing the location data.....70

Figure 37 Source code to set up environment to run code in the previous figure72

OGC[®] MOGIE Engineering Report

1 Introduction

1.1 Scope

This OGC[®] document describes the results of the MOGIE Interoperability Experiment. This includes an overview of the experiments, the demonstration scenarios, implemented components and workflows, and overall findings.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Frank W. Klucznik	Georgia Tech Research Institute

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
2013-09-04	V1.0	Frank Klucznik	All	Draft
2013-09-04	V1.2	Frank Klucznik	6.5, 6.6.7, 9	Final Draft

1.4 Future work

The following future work items have been identified in the conduct of this study. They are intended to further development and enhancement of NIEM's interoperability with OGC standards beyond that which exists in NIEM v3.0. Although some of these concepts may have been investigated in other OGC projects, these future work items were derived from MOGIE independent of any other OGC initiatives at the time of MOGIE execution. Whether these items are explored under MOGIE follow on work or some other OGC initiative is beyond the scope of this report.

1. Consider the value of and demonstrate that OGC OWS can output data in NIEM conformant XML. The WFS v2.0 specification, paragraph 7.6.3.7 allows an OGC WFS to provide data formats other than OGC conformant formats.
2. Consider the value of, and demonstrate how an event driven OGC OWS service delivering NIEM Compliant or NIEM derived data can complement the transactional capability of the WFS delivery used in MOGIE.

1.5 Forward

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.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

2.1 DoD data model sources

- NIEM: <http://niem.gov>
- C2 Core: <https://c2core.gtri.org>

2.2 OGC Documents

ISO/DIS 19142 and OGC 09-025r1, OpenGIS® Web Feature Service 2.0 Interface Standard (2010-11-02)

OGC 07-036, OpenGIS® Geography Markup Language (GML) 3.2.1 Encoding Standard (2007-08-27)

OGC 06-103r4, OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture, version 1.2.1 (2011-05-28)

OGC 05-134, OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option, version 1.1.0 (2005-11-22)

3 Terms and definitions

C2 Core	Command and Control Core
IEP	Information Exchange Package
IEPD	Information Exchange Package Documentation
GML	Geography Markup Language
MOGIE	MilOps Geospatial Interoperability Experiment
NIEM	National Information Exchange Model
WFS	Web Feature Service
WMS	Web Map Service
XML	Extensible Markup Language

4 Organizations in MOGIE

4.1 Initiator Organizations

MOGIE is initiated by the following OGC members:

- National Geospatial-Intelligence Agency (NGA)
- MITRE
- Georgia Tech Research Institute (GTRI)

4.2 Initiative Team

The MOGIE Initiative Team members are:

- Initiative Manager: Frank W. Klucznik, GTRI
- Initiative Technical Lead: Scott Bell, JS J6 DDC21 DSD
- Initiative Facilitator: Lewis Leinenweber, OGC

4.3 Complete List of Participating Organizations

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

- Envitia
- Esri
- exactEarth
- FalconView
- Luciad

5 Schedule

The MOGIE followed the following general schedule

Date	Milestone
January 31 2013	RFQ/CFP Released
March 4, 2013	Responses due
March 20-21, 2013	Kickoff at GTRI
March – May 2013	Development, testing and bug fixing
May 2013	IE Demonstrations / Demo video / Report input
July 31, 2013	Draft Report
August 30, 2013	Final report submission

A detailed execution schedule is captured below:

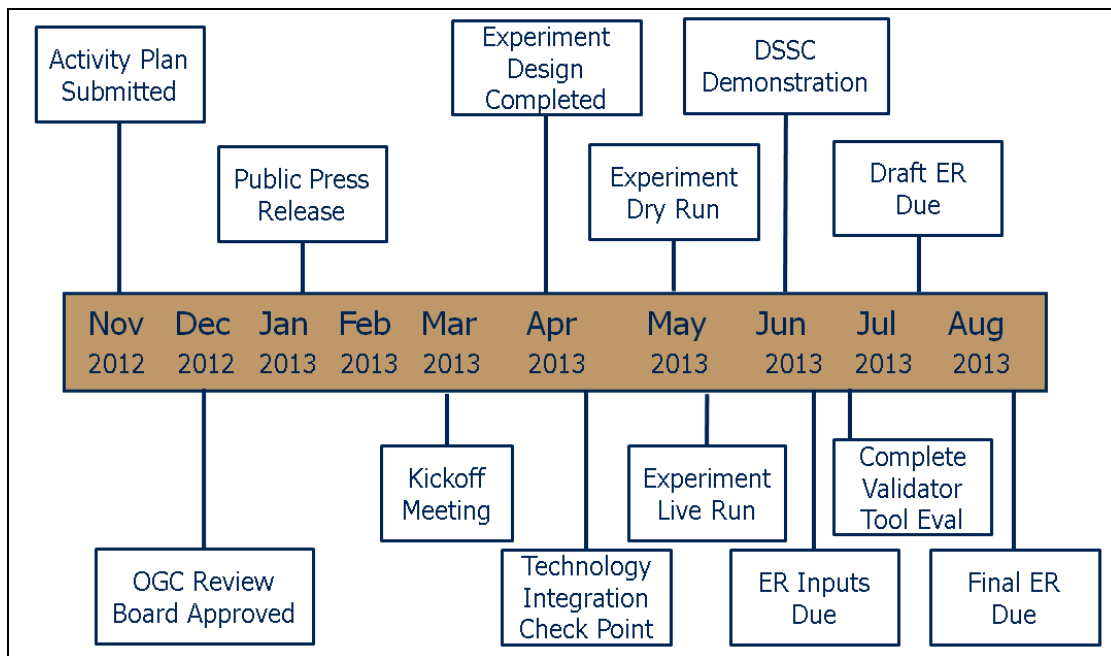


Figure 1: Detailed Execution Timeline

6 Experiment Overview

6.1 Purpose

The purpose of the Military Operations Geospatial Interoperability Experiment (MOGIE) was to test the hypothesis proposed in the “*Geolocation Accuracy and Precision in NIEM*” paper that using the National Information Exchange Model¹ (NIEM) v2.1 and v3.0 technical concepts does not introduce any changes in data related to accuracy and precision (Reference Appendix A).

In addition, MOGIE tested the theory that data may be read by a client without a priori access and knowledge of the data to demonstrate sharing data via OGC web services provides broader community interoperability than data shared without OGC services.

6.2 Background

This experiment came about as a result of community feedback on the Department of Defense (DoD) Command and Control (C2) Core² in 2011. At that time, C2 Core was an emerging data exchange capability within DoD that was 93% aligned with the NIEM v2.1 Naming and Design Rules (NDR). During the evaluation and piloting of C2 Core, community members expressed concern that using the “adapter pattern” prescribed by the C2 Core v2.0 NDR would have an impact on the accuracy and precision of maritime navigation systems.

In response to these concerns, MITRE conducted a desktop analysis which concluded NIEM technical concepts would have no impact on data related to accuracy and precision. This analysis provided the basis for the “*Geolocation Accuracy and Precision in NIEM*” paper in Appendix A of this report. In the end, not all critics were satisfied with the results of the MITRE analysis. This provided the impetus for MOGIE to demonstrate the conclusion of the MITRE analysis was in fact valid and correct.

Two weeks after the MOGIE kickoff meeting in March 2013, the DoD CIO issued a memo announcing the decision to adopt the NIEM for standards based data exchanges. The memo also stated that future enhancements to C2 Core would not be supported, and C2 Core would form the initial content in a DoD sponsored Military Operations (MilOps) domain within NIEM. (Reference Appendix D)

Immediately after the release of the DoD CIO memo, the three MOGIE initiators all agreed to remove “C2 Core” from the experiment’s activity plan and replace it with “NIEM”. Support for this decision was in large part because the C2 Core v2.0, NIEM v2.1 and NIEM v3.0 naming and design rules pertaining to the “adapter pattern” were identical. The only condition placed on this modification was to ensure preservation of the C2 Core lineage in the background section of this report.

A complimentary effort was also conducted in the OGC Geo4NIEM³, which was led OGC members, supported by commercial vendors, and sought to:

- Develop recommendations for the inclusion and standard use of embedded GML with NIEM IEPDs.
- Develop recommendations for the standardized 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 <http://niem.gov>

² <https://c2core.gtri.org>

³ https://portal.opengeospatial.org/?m=projects&a=view&project_id=428

6.3 Description

There are many commercially-available geospatial tools that exploit OGC standards. Using NIEM messages, this experiment demonstrated that GML content can be embedded in NIEM conformant XML and be exploited by commercial and open source tools without loss of precision (e.g., right number of bits) or accuracy (e.g., physical location on a map). Embedding GML in NIEM conformant XML was accomplished in MOGIE using the NIEM adapter.

There were two facets to MOGIE. The primary facet involved demonstrating the NIEM v2.1/v3.0 technical concepts work in combination with OGC's geospatial standards (e.g., GML, WFS, WMS, etc.) in the following contexts:

- NIEM conformant information exchange packages (IEPs) may contain embedded GML elements, in accordance with the GML Encoding Specification and the embedding GML best practice paper in Appendix K. This was demonstrated by embedding GML content in four different NIEM Information Exchange Package Documentations (IEPDs) and resultant IEPs. The IEPDs used in this experiment included:
 - Vessel Position Report developed by the NIEM Maritime Domain Awareness domain.
 - MOGIE Position Report loosely based on the DoD Message Text Format (USMTF) standard and Variable Message Format (VMF).
 - MOGIE Observation Report loosely based on the DoD Message Text Format (USMTF) standard and Variable Message Format (VMF).
 - MOGIE Call for Fire loosely based on the DoD Message Text Format (USMTF) standard and Variable Message Format (VMF).
- The content in NIEM conformant IEPs, including embedded GML content, may be extracted and transformed into an OGC-standard format (e.g. GML, KML), and then displayed on a client through a WFS-conforming interface. Instance data was parsed and inserted into a PostGIS database. The data was then provided through a WFS-conforming interface by GeoServer and displayed on a variety of commercial and open source products including desktop client applications, browser based applications, and handheld devices.
- The content from MilOps IEPs may be transformed to OGC-standard formats and/or made available through a WFS-conforming interface without loss of accuracy or precision. Data in the experiment was captured in raw form prior to transmission, and compared with the data stored in the PostGIS database after transmission, and compared with the data served up by GeoServer in WFS.

The secondary facets of MOGIE were included at the request of the DoD stakeholder community and are of equal importance. They included:

- Evaluate whether use of NIEM required specialty skills. That is to say, evaluate how difficult was it for someone who had never used NIEM before to understand and utilize a NIEM IEPD. The software developer selected for MOGIE had no experience with NIEM, and had 5 years of experience developing with commodity skills such as Java, general programming, XML, databases, etc.
- Evaluate and identify other potential limitations in the architecture outside the scope of the XML data exchange that could potentially affect data accuracy and precision in the experiment.

- Use and evaluate the “GML Validator” tool currently being developed in OWS-9 as appropriate to determine compliance of GML in a MilOps exchange to GML Encoding Specifications, WFS, WMS, etc. (Reference: <http://cite.opengeospatial.org/te2>). This evaluation is provided in Appendix C.
- Demonstrate that, if the NIEM content is modified to be GML application schema compliant delivered via OGC compliant services it can be exploited by a broad range of OGC clients which have no a-priori knowledge of the source..
- Evaluate the commercial implementation and support for the latest draft of the Military Standard 2525D.⁴

6.4 Use cases

The primary use case for MOGIE involved transmitting data in a NIEM conformant IEP, and leveraging GeoServer to deliver the data via WFS/WMS in an OGC conformant GML application schema for display in an OGC conformant mapping tool without loss of precision or accuracy. This use case was applied in two different scenarios: land based military patrol in Jalalabad, Afghanistan, and maritime terrorist attack in San Francisco, CA harbor; both are described in detail in Section 7.

6.5 Experiments

The following experiments were addressed during MOGIE activities:

- Experiment #1: Employ “GML Validator” currently being developed in OWS-9 as appropriate to determine compliance of GML in a MilOps exchange to GML Encoding Specifications, WFS, WMS, etc.
- Experiment #2: Extract IEP content including geospatial data that includes GML and transform it into an OGC Standard format (e.g., WMS, WFS, etc.) with military symbology as appropriate, and then display the data on a client.
- Experiment #3: Extract geospatial data that includes GML content and add additional NIEM attributes (e.g., MilOps content specified as a feature) and then display the data on a client.
- Experiment #4: Demonstrate no loss of precision or accuracy when transforming GML content (e.g., location) embedded in a NIEM conformant IEP. Expose data from a NIEM conformant IEP transformed into a GML application schema compliant form deliverable via a Web Feature Services interface and make the GML content available for vendor tools to display on a client.
- Experiment #5: Demonstrate implementation of the latest draft version of MIL-STD-2525D.

6.6 Methodology and Approach

A considerable amount of time was spent on evaluating and selecting the methodology and approach used in MOGIE. The MOGIE team recognized there were a myriad of technical architectures that could be used, and all supported accomplishment of the MOGIE objectives.

To narrow down the methodology selection, the following criteria were adopted:

- Simple – There was no need to make the experiment overly complex or introduce additional variables that could impact the results. The rational was a simple approach could be later scaled into a more complex architecture and would provide a basis for comparison.

⁴ Evaluate of MIL-STD-2525D was conducted at the request of the SSMC (Reference Section 6.6/7). Exercising the full OGC portrayal capabilities was beyond the scope of MOGIE.

- Robustness – Since the two scenarios involved emergency responses (e.g., one in a war zone and the second a national terrorist attack), the design of the architecture should be sufficiently robust to support multiple simultaneous data input sources and multiple simultaneous consumer queries.
- Performance - Given the two scenarios involved emergency responses, the design of the architecture should be optimized for performance.
- Realistic – The scenario and architecture should align with what one might expect to see in the real world and not in a laboratory test conducted in isolation.

Given the criteria above, the overall methodology chosen was to host a GIS server in single location (e.g., GTRI in Atlanta, GA, USA) and allow the participants in Canada, the United Kingdom and Belgium to participate remotely via the public Internet. All connections to the GIS server were secured with a username/password with unsecure http.

All collaboration, including the kickoff meeting, was conducted using the OGC teleconference line in combination with the DoD Defense Connect Online desktop sharing environment.

Although the scenarios used in MOGIE were intended to portray real world situations, the technical architecture and designs were not intended to mirror or replicate existing operational systems. Any similarities or dissimilarities to actual operational systems are purely coincidental.

6.6.1 Base Technical Architecture

The suite of OGC standards provided a robust baseline architecture for MOGIE, and is depicted in the illustration below. The OGC Web Service (OWS), consuming application and steps 1-5 are all prescribed by OGC, implemented by commercial or open source products, and are therefore outside the scope of MOGIE. The primary focus of the MOGIE architecture is source information and how it is inserted into the OWS.

In the example of the OGC architecture provided below in Figure 2: Base OWS Architecture, the end user recognizes a need for geospatial data in step 1. In step 2, he/she uses an OGC conformant application to search and discover data to satisfy his/her need. In step 3, the application queries one or multiple preconfigured OWSs that respond in step 4 with a capabilities document. This document lists the services that are available from a particular OWS and provides request syntax. With this information, the consuming application can now request the data needed by the user and display it.

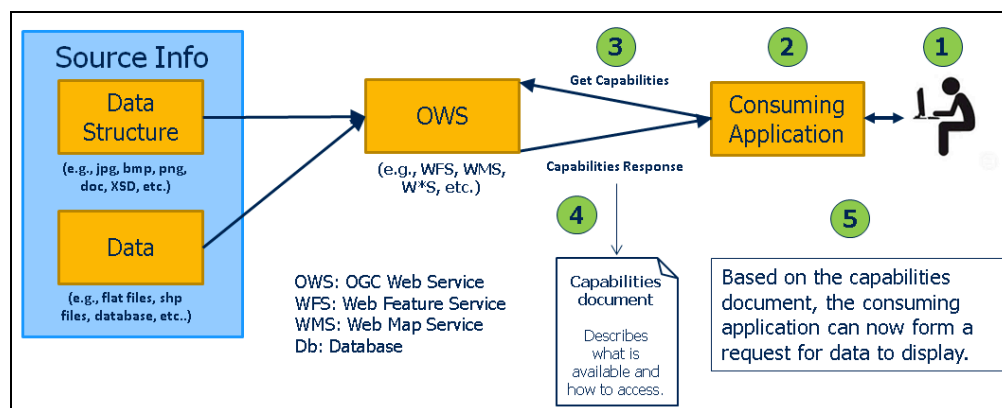


Figure 2: Base OWS Architecture

The first major architecture decision for MOGIE was to select a data source for the OWS. The data source can be provided by any source of data (e.g., stream of sensor data, flat files, relational database, etc.). Given the methodology criteria and experiment use cases, an open source relational database was selected (e.g.,

PostgreSQL). This selection supported recording and playback of the scenarios for the demonstration purposes, as well as the ability to combine (e.g., mashup) commercial and simulated DoD data for the maritime scenario.

The next major decision point was to select the method for inserting NIEM conformant XML into the MOGIE database. The team evaluated two options for inserting data into the OWS. One involved using an XSLT to transform the NIEM conformant XML into WFS transaction XML. The other was to use Java code to parse the XML and populate the database with standard SQL statements. Of the two, the Java option was chosen because it could be optimized to provide better performance over an XSLT transform, and supported the evaluation of a commodity software language as a possible limitation of accuracy and precision.

It is worth noting that when using the WFS insert transaction it is possible for the NIEM conformant XML content to be inserted into a database via Java using WFS transaction. This approach was not evaluated in MOGIE. However, this approach was evaluated in Geo4NIEM.⁵

A third possibility was to use WFS Transactions to write the results in via the WFS interface (implemented by GeoServer). This is simply an extension of option 2 above and was not evaluated within MOGIE; however, it could be undertaken in a future experiment.

The architecture below is the result of the decision to use a relational database as a data source for the OWS, and Java code to parse the NIEM XML and insert it into the database. A more technical diagram of the land and maritime architectures is provided in Section 7 in Figure 9: MOGIE Land **Scenario** and Figure 11: Maritime Terrorist Scenario.

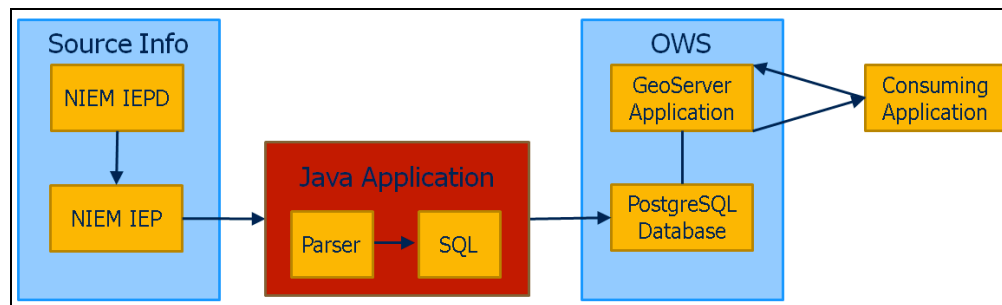


Figure 3: Base MOGIE Architecture

Note the “Java Application” in figure 3 is represented by the “MOGIE Transform Services” in Figure 9 and Figure 11.

6.6.2 Location Data Sets

Three sets of precision data for latitude and longitude were used in each iteration of the Data Collection portion of the experiment. For the purposes of this experiment, precision is measured by the number of fractional significant digits. The precisions chosen were six, ten and thirteen. For example, a longitude value 179.123456 has a precision of six and a value of 179.0123456789 has a precision of ten.

The three sets of inputs were generated from the original (e.g., raw) input data in comma separated values (CSV) format. The raw input data had longitude and latitude values with varying precisions, encoded in Well Known Text (WKT) format, i.e.: “POINT (33.7772145, -84.3961445)”. For each precision, the raw CSV coordinates were the starting point, if the precision of the raw values was greater than the target precision, the trailing digits were truncated until the precision matched. If the precision of the raw values was less than the target precision,

⁵ https://portal.opengeospatial.org/?m=projects&a=view&project_id=428

single random digits were generated and appended until the precision matched the target. New CSV files were then written with the corrected precisions. All other data fields were left unchanged and written in the new CSV file.

In all cases, the longitudes and latitudes were parsed into doubles using the method described in Appendix J.

6.6.3 Data Process and Collection Methodology

Three points for data collection were identified in the MOGIE data flow diagrams in Figure 4 and Figure 5 below. The first is the base CSV data file. The second is the PostGIS database, and the third is the output of the OWS.

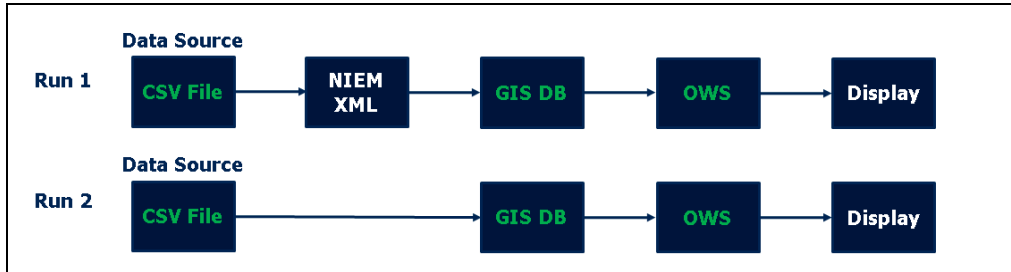


Figure 4: Land scenario data collection points

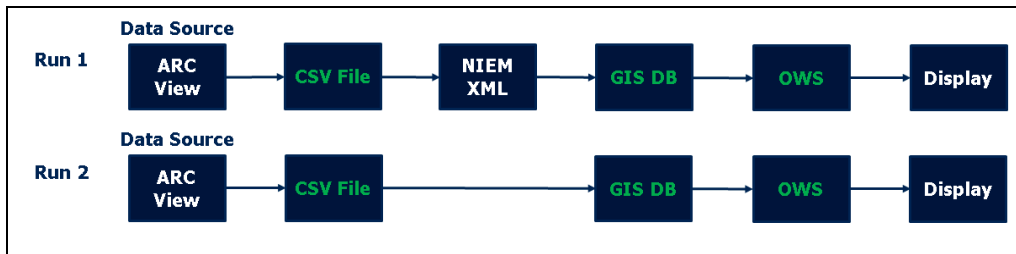


Figure 5: Maritime scenario data collection points

In run #1 of both scenarios, the CSV files were parsed and written to NIEM XML instances. This process involved starting with a NIEM XML *template* and a CSV file. The template is an instance of the Position IEP with placeholder variables mapped to the corresponding CSV values. The CSV stores the values for the XML with field names that match the placeholder variable name. The lines of the CSV file are iterated, each line populating one XML instance. For each field in a single CSV line, the field value replaces the position of the placeholder that matches the field name.

In run #2 of both scenarios, the CSV files were parsed and data inserted directly into the PostGIS database using standard SQL insert procedures.

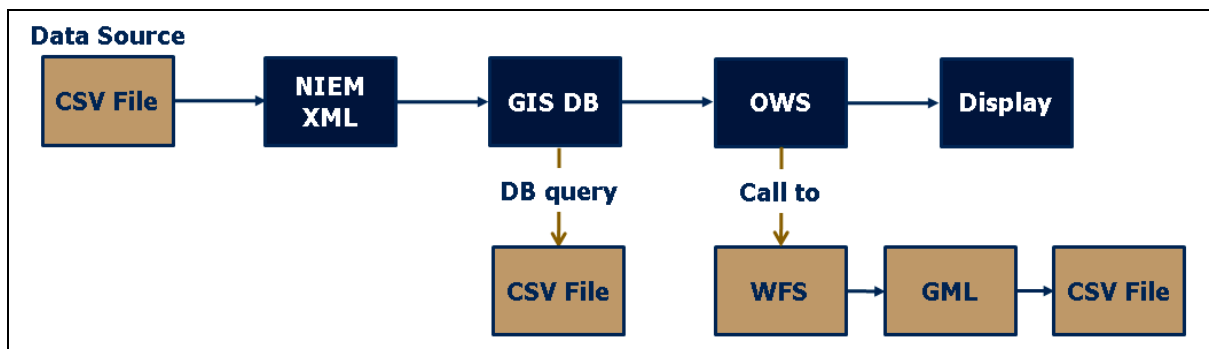


Figure 6: Land scenario data capture methodology

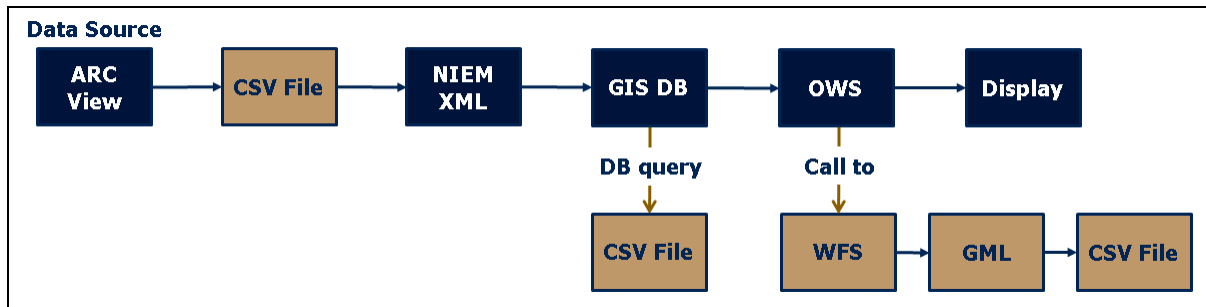


Figure 7: Maritime scenario data capture methodology

Figure 6 and Figure 7 above show the process used to extract data from the previously identified data collection points. There are two points where data is extracted from storage. The first is directly exported from the PostGIS database using standard SQL queries and formatting the output into CSV. The second used standard WFS calls to get the data from the PostGIS database in a GML application format and convert that content into a CSV format.

The result of the data capture is a set of three CSV files: base data, PostGIS data, and WFS data, which was used for comparison to determine if accuracy and precision errors were introduced by the NIEM XML or the processing of the data.

Additional details on the experiment process and data flow are included in APPENDIX B: Experiment Data Flow Description.

6.6.4 Data Comparison Methodology

The results of the position data comparison are provided in Appendix G, while Figure 8 provides a graphic illustration of the process.

During the position data comparison, three sets of data (e.g., P6, P10, P13) were compared from four different message constructs (e.g., position report, call for fire, observation report, vessel position report) from three data collection points (e.g., base, database, OWS) for two runs (e.g., w/ NIEM XML, w/o NIEM XML) in two scenarios (e.g., land, maritime). The data sets are described in Section 6.6.2; the data collection points are described in Section 6.6.3, a sample vessel position report instance is provided in Appendix H and I, and the scenarios are described in Section 7.

The land based position reports, call for fire and observation reports contain material subject to the U.S. International Traffic in Arms Regulations and are therefore not included in this report.

The data analysis included the following comparisons sequenced in the order presented:

1. RUN #1: Base CSV data compared to Database Export CSV of NIEM XML data
2. RUN #2: Base CSV data compared to database export CSV of directly inserted base CSV data
3. Database Export CSV compared to OWS exported GML data converted to CSV

During the data comparison, the sum of the absolute value of the differences in both latitude and longitude were calculated independently and reported.

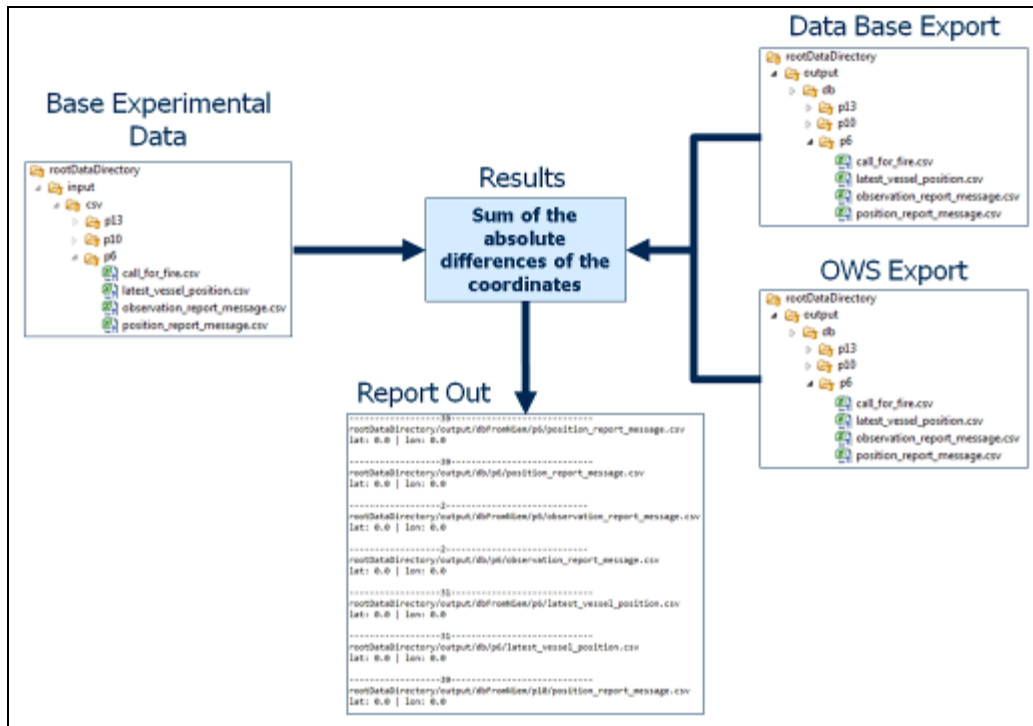


Figure 8: Data Comparison Methodology

6.6.5 Developer Skills Required

In order to evaluate the developer skills necessary to work with and implement NIEM based XML exchanges careful consideration was given to the selection of the lead developer to support MOGIE.

The individual chosen had no prior experience with NIEM. In fact, he had not even heard of NIEM before being asked to support MOGIE. His background included:

- Undergraduate degree in Computer Science
- 5 years software development experience; primarily Java
- 5 years GIS experience
- 1 year experience writing a WMS, WMTS and KML server for FalconView in C#
- Experience working with WorldWind (Java) source code during client testing

After agreeing to work on MOGIE, the developer was provided with IEPDs and sample IEPs supporting the land scenario. Although, these artifacts were derived from previous experimentation and evaluation work with Joint Staff J6 and Esri, they still required modification to be used in MOGIE.

The developer was directed to the NIEM.gov website for technical specifications, tools, and training materials. He was also directed to the NIEM IEPD clearing house as a source for IEPDs that could support the maritime scenario.

No additional training or technical support was provided.

At the conclusion of MOGIE, the developer reported he required 4 days to review NIEM technical specifications and training before he was ready to begin working with NIEM. Although he reviewed the technical specifications numerous times throughout MOGIE, he reportedly did not require any specialty skills and reported implementing NIEM XML was no more difficult than implementing any other instantiation of XML in his

experience.

6.6.6 MOGIE Limits of Precision

The Institute of Electrical and Electronics Engineers (IEEE) standard for Floating-Point Arithmetic (IEEE 754) is a technical standard for floating-point computation that is widely implemented and supported in computer systems and software today. The current version, IEEE 754-2008 is published in August 2008 and defines:

- Arithmetic formats: sets of binary and decimal floating-point data, which consist of finite numbers (including signed zeros and subnormal numbers), infinities, and special "not a number" values
- Interchange formats: encodings (bit strings) that may be used to exchange floating-point data in an efficient and compact form
- Rounding rules: properties to be satisfied when rounding numbers during arithmetic and conversions
- Operations: arithmetic and other operations on arithmetic formats
- Exception handling: indications of exceptional conditions (such as division by zero, overflow, etc.)

The standard also includes extensive recommendations for advanced exception handling, additional operations (such as trigonometric functions), expression evaluation, and for achieving reproducible results.⁶

In the IEEE 754 standard a single precision floating point number is stored in 32 bits and a double precision floating point number is stored in 64 bits. These bit-level storage requirements limit the preservation of floating point number precision. For example, when a string representing the MOGIE data value: 179.9999999999999 (e.g., 13 decimal precision) is parsed into a double and then printed back as a string it will yield an output of 179.9999999999999. However, when a string representing the value 179.99999999999999 (e.g., 14 decimal precision) is parsed into a double and then printed back as a string it will yield an output of 180.0. This rounding error is introduced as a result of the limitation of 64 bit storage and the numerical range of position data used in MOGIE.

A limitation feature identified in MOGIE was that the binary structure defined by IEEE 754 for storing floating point numbers does not preserve trailing zeros. For example, IEEE 754 binary cannot distinguish between 179.990000 (e.g., P6) and 179.99000000000000 (e.g., P13). IEEE 754 interprets both the P6 and P13 floating point numbers as: 179.99. The net result of this feature is that in MOGIE P6 and P10 data all trailing "0"s were omitted by software (e.g., Java, PostGIS, etc.) supporting IEEE 754. Refer to Appendix E for examples.

In response to these IEEE 754 limitations, the decision was made to prescribe precision level for MOGIE as P6, P10 and P13. In addition, trailing "0" were not replaced in the data because it did not affect the outcome of the experiment in any way.

Finally, to give readers an idea of what 13 decimal precision latitude and longitude position means in practical terms, P13 latitude and longitude precision identifies a specific location to $111,111/10^{13}$ (at the equator), or approximately 1 angstrom (e.g., the size of a chlorine atom). Although this level of precision is impossible to discern on a computer monitor, it has the potential to make a difference in some DoD targeting and tracking systems.

⁶ https://en.wikipedia.org/wiki/IEEE_floating_point

6.6.7 MIL-STD-2525D

At the request of the Defense Information Systems Agency (DISA) as the custodian for MIL-STD-2525, and with the approval of the Symbology Standards Management Committee (SSMC), commercial vendors participating in MOGIE were asked to use and evaluate the January 2013 draft version of MIL-STD-2525D. Both Envitia and Luciad fulfilled the request and reported no issues or recommended changes to the draft standard.⁷

MIL-STD-2525 provides a standardized, structured set of graphical symbols for the display of information in command and control (C2) systems and applications. As such, it represents a standard method for symbol construction using common building blocks which can be used to create current symbol sets. This includes frame, icon, modifier, and amplifier using color, graphic, and alphanumeric representations. It provides requirements for symbol construction and composition with flexibility for special user needs. This standard is approved for use by all departments and agencies of the Department of Defense (DOD) and available for use by non-DOD entities (e.g., first responders, United Nations, and multinational partners).

⁷ Exercising the full OGC portrayal capabilities was beyond the scope of MOGIE.

7 Scenario Overview

MOGIE involved two scenarios: an Army land based scenario, and a maritime terrorist scenario. This section describes both in detail.

7.1 Land scenario

The land based scenario takes place in Jalalabad, Afghanistan and involved simulated DoD data sources.

7.1.1 Scenario Description

In the land-based scenario the 25 Infantry Division, 2nd Stryker Brigade Combat Team is set up to the west of Jalalabad. The brigade set up a field artillery battery to their east and west of the city in support of troops on patrol inside the city. A tactical operations center was also set up south of the city to improve communications with troops inside the city.

A company composed of five Infantry Squads was patrolling a section of the city beginning in the south and working northward, with two lead squads serving as scouts patrolling north of the main body of the company on the east and west boundaries of the patrol area. The remaining three squads are working up the center of the patrol area.

One of the scouts reports observing an enemy squad to the north of Alpha Company and requests fire support to destroy the enemy. A call for fire message is sent to the artillery battery and the enemy is fired upon. The scout squad observes the results of the fire and reports the enemy destroyed.

Alpha Company advances forward, verifies the kill, and continues on their patrol.



Figure 9: MOGIE Land Scenario

7.1.2 Technical architecture

The illustration below provides a view of the land scenario technical architecture.

NIEM IEPDs simulating military position reports, call for fire and observation reports were used to generate the NIEM conformant XML. The MOGIE Transform Service includes the XML parser and Java code written to insert the data into the PostGIS database.

GeoServer⁸ is an open source product that served the data in OGC conformant WFS to a desktop, browser or smart phone applications as appropriate. The WFS Transactions in this diagram were present but not used.

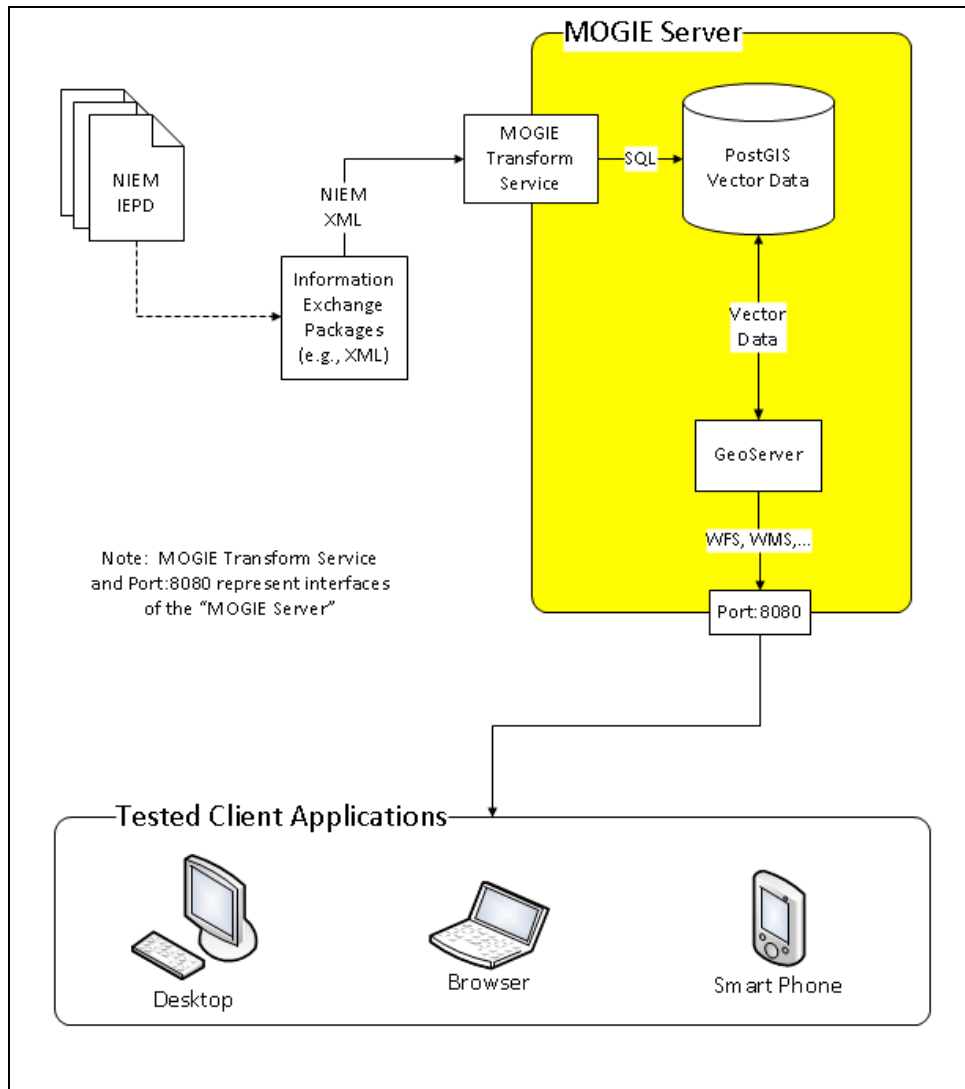


Figure 10: MOGIE Land Scenario Technical Architecture

Note the “MOGIE Transform Service” in Figure 10 is represented as the “Java Application” in Figure 3: Base MOGIE Architecture.

⁸ <http://geoserver.org/display/GEOS/Welcome>

7.1.3 Data overview

Source data was produced from NIEM conformant XML based information exchanges. Three different data exchanges were used:

- Position Report
- Enemy Observation Report
- Call For Fire

A data overview for the land-based scenario is provided in Section 8.1.

7.2 Maritime scenario

The maritime scenario involves a suspected terrorist attack in the San Francisco harbor area. Data sources include civilian vessel data as well as simulated DoD data sources.

7.2.1 Scenario Description

In the maritime scenario, a terrorist attack is reported as an explosion at a pier in San Francisco Harbor in California. Coast Guard Vessel Traffic Service (VTS) is monitoring vessels in the area via radar and AIS data. Coast Guard Pacific Area Command Center manages response to the incident and monitors vessel traffic on a common operational picture (COP) displaying public AIS data as well as data from DoD and Coast Guard vessel position reporting systems.

The USS JOHN PAUL JONES, Coast Guard Cutter ACTIVE, and Coast Guard Cutter WAESCHE are operating just outside San Francisco Bay and are directed to check vessel traffic exiting the bay. The Coast Guard Cutter SOCKEYE and Coast Guard Cutter TERN are operating in San Francisco Bay and are directed to the scene and check vessel traffic coming from the direction of the explosion.

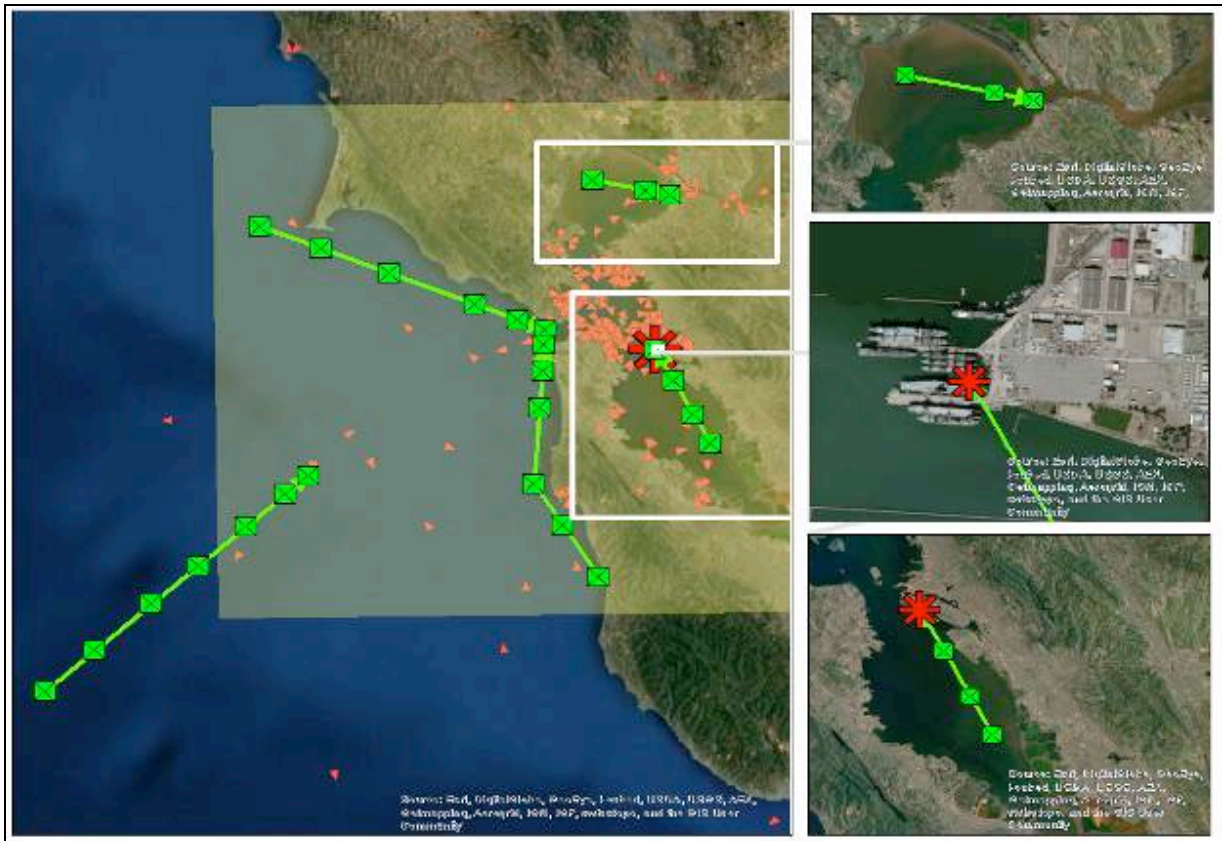


Figure 11: Maritime Terrorist Scenario

7.2.2 Technical architecture

Figure 12 provides a view of the maritime scenario technical architecture.

NIEM Maritime Domain Awareness IEPDs for vessel position reports were used to generate the NIEM conformant XML. The MOGIE Transform Service includes the XML parser and Java code written to insert the data into the PostGIS database.

Additional commercial vessel Automatic Identification System (AIS) data was provided by exactEarth in support of the maritime scenario. This data was stored in the PostGIS database through cascaded WMS and WFS and served up to clients as requested.

GeoServer⁹ is an open source product that served the data in OGC conformant WFS to a desktop, browser or smart phone applications as appropriate. The WFS Transactions in this diagram were present but not used.

⁹ <http://geoserver.org/display/GEOS/Welcome>

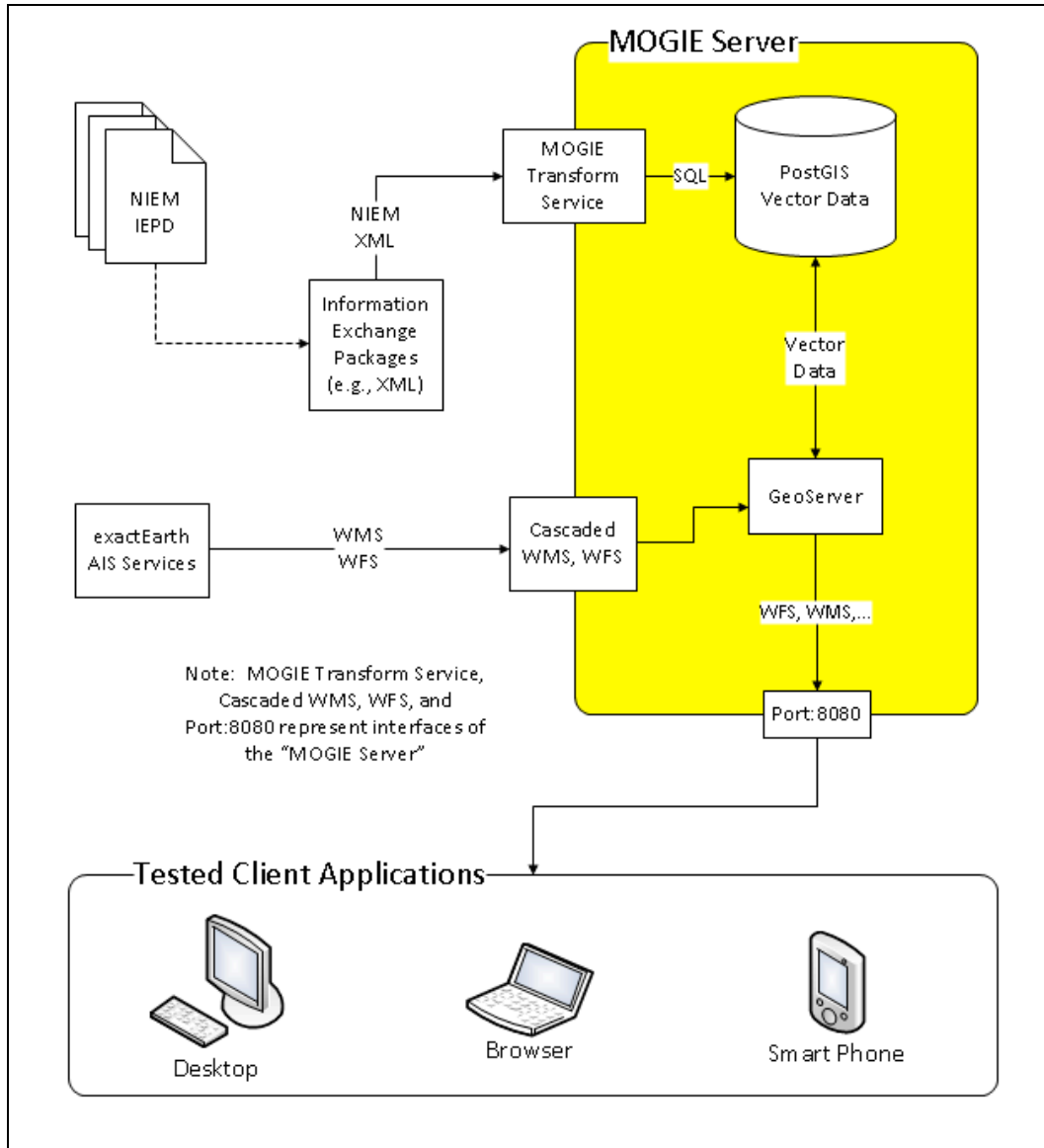


Figure 12: Maritime Scenario Technical Architecture

Note the “MOGIE Transform Service” in Figure 12 is represented as the “Java Application” in Figure 3: Base MOGIE Architecture.

7.2.3 Data overview

A data overview for the maritime-based scenario is provided in Section 8.2. Example NIEM IEP and corresponding GeoServer output XML are provided in Appendix H and I respectively.

8 Components

8.1 Data Source and Data Storage

This section discusses the source of the data used in MOGIE and how it was ultimately stored in the database tables. In both scenarios, the data used to support the MOGIE experiment was stored in a PostGIS enabled PostgreSQL database, which acted as the backend data source to the WFS, as shown in Figure 10 and Figure 12.

8.1.1 Land Scenario Data

Section 7.1 describes the land scenario which simulates a military use case where an enemy is spotted, targeted and destroyed. The observations and resulting attack information is collected, organized and distributed using NIEM and GML conformant XML exchanges processed through MOGIE services.

The three NIEM IEPDs used in this scenario were: Position Report Message (POSREP), Observation Report Message (OBSREP) and Call For Fire (CFF). The NIEM XML instances generated from these IEPDs were parsed with Java code and inserted in the PostGIS database using simple SQL statements. The IEPDs used in the land scenario were derived from existing IEPDs previously used in multiple venues during the development of C2 Core. For MOGIE, the IEPDs were modified in that all CUI material was removed and they were made to fully conform to NIEM v3.0 Alpha 2 release.

8.1.1.1 POSREP IEPD

The POSREP IEPD conveys the location of a reporting unit at a certain point in time. It also includes additional unit status information such as unit strength, unit identifier, unit name, etc. Additional attributes included in the database but not in the IEPD were `message_time_text`, `time_stale`, and `time_stale_text`. These were added to support the client development and configuration.

The `time_stale` attribute indicates the time when this message is no longer valid because the unit issued an updated location message. Because simulating an actual real-time data feed was outside the scope of this experiment, it was decided to playback the position reports from a pre-populated set of reports. The client developer only had to query the WFS service for the entire set and then replay them. Because the KML timespan feature was being leveraged, a start and end time was needed for each message report. For more information see Section 8.2.2.

The `message_time_text` and `time_stale_text` attributes are string type versions of the Timestamp attributes. This was necessary to facilitate the temporal field features in ArcMap. GeoServer currently has no feature that allows a timestamp format string to be passed with a WFS query. This means the WFS clients were forced to work with the default time string format. (ie: "2001-12-17T09:31:16Z") Since this was not one of the immediately supported formats in ArcMap, a text version of the timestamp attributes was added with the timestamps pre-formatted as a string that ArcMap understood. (ie: "2001-12-17 09:31:16") This modification was made to all timestamp types found in both scenarios, not just Position Report Message. They are all suffixed with "_text".

Figure 13 shows the data fields and definitions in the PostGIS database. The field "From IEPD" indicates whether that field originated from the NIEM source. The "Client Support" field indicates whether that field was added to support client development.

Position Report Message			
Attribute Name	Definition	From IEPD	Client Support
exercise_nickname	Text identifying this exercise	Yes	No
message_time	Timestamp when the message was created	Yes	No
unit_location	Point representing unit location in epsg:4326	Yes	No
originator_id	An identifier of the originator of the message	Yes	No
unit_name	Nomenclature authoritatively assigned to specifically identify a military element	Yes	No
unit_id	Unique identifier for the reporting organization	Yes	No
unit_strength	unit strength in percent (0 <= unit_strength <= 1)	Yes	No
unit_sidc_2525b	A unit SIDC symbol code as defined in MIL-STD-2525B	Yes	No
unit_sidc_2525c	A unit SIDC symbol code as defined in MIL-STD-2525C	Yes	No
unit_sidc_2525d	A unit SIDC symbol code as defined in MIL-STD-2525D	Yes	No
message_id	Text identifying this message	Yes	No
message_time_text	Alternate text format time representation for message_time ex: "2001-12-17 09:31:16"	No	Yes
time_stale	Timestamp representing the time at which this message is no longer valid A null value indicates that this is the most recent message	No	Yes
time_stale_text	Alternate text format time representation for time_stale ex: "2001-12-17 09:31:16"	No	Yes

Figure 13: Position Report Database Attributes

8.1.1.2 OBSREP IEPD

The OBSREP IEPD reports an observation made from a field unit. Data in this messages includes the locations and id’s of both the target and observer, as well as the time of the observation. The figure below shows the fields as they are stored in the database.

It was not necessary to add the time_stale field to the OBSREP table because this IEPD represents a single observation and not a position update as with POSREP.

Figure 14 below shows the data fields and definitions in the PostGIS database.

Observation Report Message			
Attribute Name	Definition	From IEPD	Client Support
exercise_nickname	Text identifying this exercise	Yes	No
observation_time	A date and time group when the observation occurred	Yes	No
originator_id	An identifier of the originator of the message	Yes	No
message_id	An identifier of a message or report	Yes	No
message_time	Timestamp when the message was created	Yes	No
unit_sidc_2525b	A unit SIDC symbol code as defined in MIL-STD-2525B	Yes	No
unit_sidc_2525c	A unit SIDC symbol code as defined in MIL-STD-2525C	Yes	No
unit_sidc_2525d	A unit SIDC symbol code as defined in MIL-STD-2525D	Yes	No
target_position	Point object storing the location of the target in epsg:4326	Yes	No
observer_position	Point object storing the location of the observer in epsg:4326	Yes	No
message_time_text	Alternate text format time representation for message_time ex: "2001-12-17 09:31:16"	No	Yes

Figure 14: Observation Report Database Attributes

One interesting challenge with processing this message was the existance of two locations within the same object. This approach, while valid and relevant to the design, made it more difficult to configure the labeling and general behavior of the visual representation of the message in GeoServer and some client applications. For example, by default, ArcMap will not display both of the locations when imported from WFS as GML. Additional modifications have to be made to the WFS import settings to convert the two points to a single multipoint type. Then more work must be done to correctly style the points independently.

8.1.1.3 CFF IEPD

The Call For Fire IEPD conveys an order to attack. It includes information about the target, performing unit, observing unit and the message originator.

Figure 15 shows the fields as they are stored in the database.

Call for Fire			
Attribute Name	Definition	From IEPD	Client Support
exercise_nickname	Text identifying this exercise	Yes	No
target_id	An identifier of the entity or object considered for possible engagement or other action like a target	Yes	No
message_id	An identifier of the message or report	Yes	No
unit_performing_fire_mission	Name of unit performing fire mission	Yes	No
unit_performing_fire_mission_id	Organization unique id for unit performing fire mission	Yes	No
message_time	Timestamp when the message was created	Yes	No
originator_id	An identifier of the person or organization who created the original message	Yes	No
fire_mission_observer_id	Identifier for the forward observer for this fire mission	Yes	No
fire_mission_observer_position	Point object storing the location of the observer in epsg:4326	Yes	No
target_position	Point object storing the location of the target in epsg:4326	Yes	No
message_time_text	Alternate text format time representation for message_time ex: "2001-12-17 09:31:16"	No	Yes

Figure 15: Call for Fire Database Attributes

8.1.2 Maritime Scenario Data

Section 7.2 describes the maritime scenario which simulates an emergency response event in San Francisco, CA. In the scenario, a terrorist attack is reported, and appropriate action is taken using military vessels that are in the area at the time of the event. In addition, relevant civilian vessel data was included from a commercial source to enable real time vessel tracking as well as supporting post incident investigation.

8.1.2.1 DoD Vessel Data

The basis for the simulated DoD vessel data was the Maritime Domain Awareness Vessel Information IEPD 3.2 (VEPOR) found in the public NIEM Clearinghouse.¹⁰ Supporting data was generated using Esri ArcMap v10.1. An empty point shapefile was created with the fields extracted from the IEP. The points were then manually placed on the scenario map, using the editor features in ArcMap. Finally, the vessel info, speed and heading attributes were calculated based on the vessel positions. Once the fields are all populated, the data can be extracted as a CSV file. This file was used as the input data for the experiments.

The simulated military vessels were stored in the database with the table named LatestVesselPosition. The attributes include information about the vessel’s identity, owner, current position and movement et al.

Figure 16 shows the data fields and definitions in the PostGIS database.

¹⁰ <https://it.ojp.gov/framesets/iepd-clearinghouse-noClose.htm>

Latest Vessel Position			
Attribute Name	Definition	From IEPD	Client Support
call_sign	The call sign for the vessel	Yes	No
hull_number	The hull number of the vessel	Yes	No
imo_number	The International Maritime Organization Number (IMO number) of the vessel	Yes	No
mmsi	The Maritime Mobile Service Identity (MMSI) of the vessel	Yes	No
vessel_name	The name of the vessel	Yes	No
national_flag_code	The national flag under which the vessel sails as defined in IOS3166A3 (ie:USA)	Yes	No
owner_full_name	The full name of a person, organization, or thing capable of bearing legal rights and responsibilities for the vessel	Yes	No
owner_nationality_code	The country in which the owner was born	Yes	No
sco_num	The Ship Control Number (SCONUM) of the vessel	Yes	No
speed	The measure of the speed of the vessel or other conveyance	Yes	No
speed_units	The speed units (ie: kt)	Yes	No
course	The measure of the angular course of the vessel or other conveyance	Yes	No
course_units	The angular units of the course (ie: deg)	Yes	No
heading	The measure of the angular heading of the vessel or other conveyance	Yes	No
heading_units	The angular units of the heading (ie: deg)	Yes	No
navigation_status	The navigation status of the vessel	Yes	No
latest_position_time	The timestamp of the latest_position sample	Yes	No
latest_position	The last reported location of the vessel in EPSG:4326	Yes	No
latest_position_time_text	Alternate text format time representation for latest_position_time ex: "2001-12-17 09:31:16"	No	Yes
unit_sidc_2525d	A unit SIDC symbol code as defined in MIL-STD-2525D	No	Yes
time_stale	Timestamp representing the time at which this message is no longer valid A null value indicates that this is the most recent message	No	Yes
time_stale_text	Alternate text format time representation for time_stale ex: "2001-12-17 09:31:16"	No	Yes

Figure 16: Latest Vessel Position Database Attributes

8.1.2.2 Commercial Vessel Data

The source of the commercial data used in the maritime scenario was provided by exactEarth¹¹ AIS Services through a WFS input directly into the MOGIE server. This section will describe how the data provided by exactEarth was recorded for the experiment. For more general information about exactEarth see Section 8.4.2.

MOGIE used two sources of data from exactEarth which included the: Latest Vessel Information (LVI) and Historical Track 30. (HT30) LVI is a real-time source of the vessels being tracked by exactEarth. HT30 is a 30 day track history for the tracked vessels, which supported playback of the historical movement of vessels in the area at the time of the simulated incident. exactEarth data was recorded for a window in time supporting the scenario and stored on the PostGIS database.

The LVI WFS provides the most recent positions for vessels over the entire world. To record this for MOGIE, the service was queried every minute for four hours. Each query filtered the results to the San Francisco Bay area, the exact bounding box is shown in Figure 11. The returned information was stored in the database at each request. If a location-time pair already existed for a returned vessel, that record was discarded. The end result is a four hour cache of the LVI data.

When played back, the raw LVI data had numerous ships that either did not move or moved very little over the four hour time span. This made it difficult to view the ships that did move. To deal with this, vessels were filtered to those with adequate distances between location samples. The LVI service returns the most recent location of the vessel even if it is up to 30 days old. Because this contributed to the clutter and data size, these older location samples were also thrown out.

The result was 163 unique vessels tracked from 04/13/2013 18:13:46 to 04/14/2013 06:53:25 UTC. The HT30 track for each remaining vessel was then retrieved via exactEarth's WFS service. The 30 day tracks were to be used as an investigative tool to find out where a given vessel had been in the recent past.

Figure 17 shows LVI and HT30 data fields and definitions in the PostGIS database.

¹¹ <http://www.exactearth.com/>

Latest Vessel Information		30 Day Track History	
Attribute Name	Definition	Attribute Name	Definition
mmsi	A Maritime Mobile Service Identity (MMSI) of a vessel	mmsi	A Maritime Mobile Service Identity (MMSI) of a vessel
imo	An International Maritime Organization Number (IMO number) of a vessel	imo	An International Maritime Organization Number (IMO number) of a vessel
vessel_name	The name of the vessel	vessel_name	The name of the vessel
callsign	The call sign for a vessel	callsign	The call sign for a vessel
vessel_type	A description of a type of a vessel based upon the purpose for which the vessel was designed or built.	vessel_type	A description of a type of a vessel based upon the purpose for which the vessel was designed or built.
vessel_type_code	unique code assigned to corresponding vessel type	vessel_type_code	unique code assigned to corresponding vessel type
vessel_type_cargo	A description of the kind of cargo that a vessel is carrying. Cargo types may be categorized as DG=Dangerous Goods, HS=Harmful Substances, or MP=Marine Pollutants.	vessel_type_cargo	A description of the kind of cargo that a vessel is carrying. Cargo types may be categorized as DG=Dangerous Goods, HS=Harmful Substances, or MP=Marine Pollutants.
length	Length of Bow to Main Tower and Main Tower to Stern	length	Length of Bow to Main Tower and Main Tower to Stern
width	Length of Port to Main Tower and Main Tower to Starboard	width	Length of Port to Main Tower and Main Tower to Starboard
flag_country	Name of country's flag (ie:USA,Japan)	flag_country	Name of country's flag (ie:USA,Japan)
flag_code	A country under which a vessel sails, represented as an ISO 3166 three letter code	flag_code	A country under which a vessel sails, represented as an ISO 3166 three letter code
destination	Port of Destination	destination	Port of Destination
draught	Vessel Draught	draught	Vessel Draught
longitude	WGS84 longitude in decimal degrees	from_longitude	WGS84 longitude in decimal degrees
latitude	WGS84 latitude in decimal degrees	from_latitude	WGS84 latitude in decimal degrees
sog	Speed Over Ground A measure of the speed of a vessel or other conveyance	longitude	WGS84 longitude in decimal degrees
cog	Course over Ground A measure of the angular course of a vessel or other conveyance	latitude	WGS84 latitude in decimal degrees
rot	Rate of Turn	sog	Speed Over Ground A measure of the speed of a vessel or other conveyance
heading	A measure of the angular heading of a vessel or other conveyance	cog	Course over Ground A measure of the angular course of a vessel or other conveyance
nav_status	A navigation status of a vessel or other conveyance at a particular position	rot	Rate of Turn
nav_status_code	Navigational Status Code	heading	A measure of the angular heading of a vessel or other conveyance
position	A location specified by a 2D or 3D geometric point	nav_status	A navigation status of a vessel or other conveyance at a particular position
ts_position_utc	The date and time that a position of a vessel or other conveyance was recorded or measured	nav_status_code	Navigational Status Code
ts_static_utc	Date and Time of Last Static AIS Message in UTC	segment	A line segment from the previous position to the next position.
eta	Month, Day, Hour, and Minute of Estimated Time of Arrival in UTC	ts_position_utc	The date and time that a position of a vessel or other conveyance was recorded or measured
time_stale	Timestamp indicating that his location sample is out of date. If null, this is the most recent sample.	ts_static_utc	Date and Time of Last Static AIS Message in UTC
		eta	Month, Day, Hour, and Minute of Estimated Time of Arrival in UTC

Figure 17: LVI and HT30 Database Attributes

The primary difference between the LVI and HT30 fields is the addition of from_longitude and from_latitude. These represent the previous location of the vessel. It appeared that exactEarth limits the LVI dataset to the most recent position update of each tracked vessel to improve performance. This approach allows a simple query to download the latest vessel positions for the entire planet without special filters. HT30, on the other hand, supports numerous position updates for each vessel. If the same simple query, used with LVI, were allowed with HT30 a huge amount of data would have to be packaged and transmitted to the client, seriously effecting performance on both the client and server. exactEarth forces queries to the HT30 layer to filter the results to a single vessel, using the mmsi field.

8.1.3 Data Parsing and Processing

This section explains the process used to parse the NIEM XML and insert it into the PostGIS database in both the land and maritime scenarios. During the experiment, four IEPDs were processed including the: POSREP, OBSREP, CFF and VEPOR. All four followed the same process described in this section.

8.1.3.1 Overview

Figure 3: Base MOGIE Architecture illustrates the high-level architecture to which this section will refer. The Java Application component is the custom Java source code written for the MOGIE experiment. This is also depicted in Figure 10: MOGIE Land Scenario Technical Architecture and Figure 12: Maritime Scenario Technical Architecture as the MOGIE Transform Service. This application was written using Eclipse Indigo SR1

in Java version 1.6. The database was hosted using PostgreSQL¹² with the PostGIS¹³ extension installed. PostGIS is the spatial extension to PostgreSQL. It extends PostgreSQL by adding advanced geo-spatial capabilities such as spatial query filters, multiple coordinate system support, common spatial types, etc. GeoServer is the actual server component that manages the OGC services. With a PostGIS table in place, one can use the GeoServer web interface to create, configure and manage the service that delivers the spatial data in the table.

OpenGeo version 3.0.2 was used to install and configure GeoServer, PostGIS, PostgreSQL and the various dependencies included in the OpenGeo stack.¹⁴ Referencing Figure 3, all interaction between the Parser and SQL components, that is, all interaction between the MOGIE Java application and the database used the official JDBC4 version 9.2-1001 driver provided by PostgreSQL.¹⁵

8.1.3.2 XML Parsing

The conversion from XML to the database was a standard process of parsing the XML to extract the data and then constructing SQL statements to insert that data into the database. The standard Java DOM parser was used to parse the xml. The elements of each IEP were mapped to the column names in the database.

The initial input is a NIEM XML instance, an example of one for VEPOR is provided in appendix H. The part most relevant to this section is the GML Point element:

```
<gml:Point srsName="http://metadata.ces.mil/mdr/ns/GISP/crs/WGS84E_2D">
  <gml:pos>37.1498819680237 -123.5583574667275</gml:pos>
</gml:Point>
```

The above GML Point has two pieces of important information, the srsName attribute and the gml:pos element. The srsName attribute indicates the coordinate system used for the coordinate values. Among other things, the coordinate system defines the units and axis ordering for the coordinates. In this case, the referenced system uses angular units, latitude and longitude, with latitude first. The gml:pos element holds the actual coordinates as a GML doubleList type. (GML3.2.1, Section 10.1.4.1)

Once the XML file has been loaded into a Java Document object, all of the contained XML is organized into a tree structure. Each element is a node with attributes and child and/or parent elements. When the Point element has been retrieved as a Node object, getAttributes() can be called to retrieve the attributes for the Point element and getChildNodes() can be called to retrieve the child element gml:pos. Finally, getNodeValue() is called from the gml:pos Node object which returns the coordinates as a Java String, in this example the String would print as: "37.1498819680237 -123.5583574667275".

¹² <http://www.postgresql.org>

¹³ <http://postgis.net>

¹⁴ <http://opengeo.org/publications/opengeo-architecture>

¹⁵ <http://jdbc.postgresql.org>

GML Data Types

Because the conversion of the actual values from one type to another is the true source of any loss of data, here the GML Point type definition is discussed.

GML PointType is defined in OGC 07-036 (GML3.2.1) Section 10.3.1. The section states:

A gml:Point is defined by a single coordinate tuple. The direct position of a point is specified by the gml:pos element which is of type gml:DirectPositionType.

gml:Point implements ISO 19107 GM_Point (see D.2.3.3 and ISO 19107:2003, 6.3.11).

The use of the element “coordinates” is deprecated. Use “pos” instead.

```
<complexType name="PointType">
  <complexContent>
    <extension base="gml:AbstractGeometricPrimitiveType">
      <sequence>
        <choice>
          <element ref="gml:pos" />
          <element ref="gml:coordinates" />
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<element name="Point"
  type="gml:PointType"
  substitutionGroup="gml:AbstractGeometricPrimitive"
/>
```

Figure 18: GML PointType Definition

Figure 19 shows the defining XML. From here, the choice of how to actually store the coordinate values comes down to gml:pos or gml:coordinates, but as stated by the standard, gml:coordinates is deprecated so the gml:pos element will actually hold the values.

GML DirectPositionType (gml:pos) is defined in Section 10.1.4.1 of GML3.2.1. Figure 19 shows the defining XML, and indicates that the values will be stored as a gml:doubleList.

GML doubleList is defined in Section 8.2.4.1 of GML3.2.1. Figure 20 shows the defining XML. The sections states:

NOTE 3 An element which uses one of these types will contain a whitespace-separated list of members of the relevant type

(see <http://www.w3.org/TR/xmlschema-2/#atomic-vs-list> for more details of the XML list structure).

```
<complexType name="DirectPositionType">
  <simpleContent>
    <extension base="gml:doubleList">
      <attributeGroup ref="gml:SRSReferenceGroup"/>
    </extension>
  </simpleContent>
</complexType>
<element name="pos" type="gml:DirectPositionType"/>
```

Figure 19 GML DirectPositionType

```
<simpleType name="doubleList">
  <list itemType="double"/>
</simpleType>
```

Figure 20 GML doubleList

XML double is defined in XML Schema Part 2: Datatypes Second Edition¹⁶ Section 3.2.5. The section states:

[Definition:] The double datatype is patterned after the IEEE double-precision 64-bit floating point type [IEEE 754-1985]. The basic value space of double consists of the values $m \times 2^e$, where m is an integer whose absolute value is less than 2^{53} , and e is an integer between -1075 and 970, inclusive...

¹⁶ <http://www.w3.org/TR/xmlschema-2/#double>

...Leading and trailing zeroes are prohibited subject to the following: number representations must be normalized such that there is a single digit which is non-zero to the left of the decimal point and at least a single digit to the right of the decimal point unless the value being represented is zero. The canonical representation for zero is 0.0E0.

The following section discusses the next step which is to parse the text representation of the coordinates, convert them into IEEE 754 doubles and insert them into the database.

8.1.3.3 Database Insertion

The standard PostgreSQL JDBC driver was used to populate the database from within Java. This step is where the longitude and latitude are converted to doubles in Java and then to a Point type in the database. This section will follow a high level discussion of this process, for a detailed example of the process including source code see appendix J.

Essentially, the input to the database insertion code is a string with a latitude followed by a longitude. From the GML Point definition, it is assured that this string should be two valid IEEE 754 doubles with a single space between them. The single string can now be split into two separate strings, without the space, by calling the `String.split()` function. Each of these can then be converted to Java doubles with the function `Double.valueOf(String)`.¹⁷ Now the coordinates are actually stored as IEEE 754 doubles in memory.

Next the doubles must be prepared to be inserted into the PostGIS database. PostGIS will store the coordinates as a Point object, or column, in the spatially enabled database table. To achieve this, the coordinates will first be converted to Well Known Text (WKT) format and then passed into the SQL function `ST_GeomFromText(WKT,EPG)`, provided by the PostGIS extension. The `ST_GeomFromText` function converts a WKT representation of an OGC Point, into the internal binary structure. Again, detailed steps are shown in annex J. It's worth noting here that there are obviously multiple ways to insert GML into a PostGIS database. This approach was chosen to make the actual coordinate conversion process as explicit and visible as possible. Also, while it is not necessary to convert the coordinate values from text into doubles for this simple example, common use-cases might require including this conversion for error-checking, pre-processing, additional calculations on the coordinates before inserting them into the database.

Once the data is in the database, GeoServer takes the conversion from there. However, reading from the database is not overly complicated. This can be accomplished with PostGIS functions. For example, `ST_AsText(geometry_column)` will return the value of a spatial column as WKT. To ensure the returned value has not lost precision, a more direct function is `ST_AsBinary(geometry_column)` which will return the Point as Well Known Binary. (WKB) WKB encodes the actual coordinate values as binary IEEE 754 doubles.

Data Types

The **Java double** is explained in the Java Language Specification Section 4.2.3. (<http://docs.oracle.com/javase/specs/jls/se7/html/jls-4.html#jls-4.2.3>) The section states:

The floating-point types are float and double, which are conceptually associated with the single-precision 32-bit and double-precision 64-bit format IEEE 754 values and operations as specified in IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985.

Point is defined in OGC 06-103r4 Simple Feature Access - Part 1: Common Architecture (SFA-CA) Section 6.1.4.

Well Known Text (WKT) is defined in SFA-CA Section 7. An example can be viewed in annex J.

Well Known Binary (WKB) is defined in SFA-CA Section 8. The section states:

¹⁷ [http://docs.oracle.com/javase/6/docs/api/java/lang/Double.html#valueOf\(java.lang.String](http://docs.oracle.com/javase/6/docs/api/java/lang/Double.html#valueOf(java.lang.String)

The Well-known Binary Representation for Geometry is obtained by serializing a geometric object as a sequence of numeric types drawn from the set {Unsigned Integer, Double} and then serializing each numeric type as a sequence of bytes using one of two well defined, standard, binary representations for numeric types (NDR, XDR)...

... A Double is a 64-bit (8-byte) double precision datatype that encodes a double precision number using the IEEE754 double precision format.

The above definitions are common to both XDR and NDR.

More information on the functions ST_GeomFromText, ST_AsText and ST_AsBinary can be found in OGC 05-134 Simple feature access - Part 2: SQL option (SFA-SQL) and ISO/IEC CD 13249-3:2006(E) – Text for FDIS Ballot Information technology – Database languages – SQL Multimedia and Application Packages — Part 3: Spatial, May 15, 2006. (SQL/MM)

8.2 Data Delivery Services

The two primary outputs used in MOGIE were GML via WFS and KML via WMS. This section explains the interaction between the client and server for both output types.

8.2.1 WFS and GML

WFS has many components and supports multiple formats. It is outside the scope of this document to thoroughly describe them all. Therefore, this section describes the components of WFS used by MOGIE and the resulting GML output.

As with all OWS compliant services, there is an initial negotiation phase that begins with the client application requesting a capabilities document. While the structure of these XML documents varies from service to service, their general function is the same, they provide the client with the necessary knowledge to query the services provided. Figure 21 illustrates this process.

- The first arrow shows the client issuing a GetCapabilities request to the server. This initial request typically takes the form of a HTTP GET call. This is the type of call made by a web browser when a URL is entered into the *browser address bar*. To construct the GetCapabilities request, one starts with a base address, in the case of MOGIE: <http://falconservices.icl.gtri.org:8080/geoserver/ows?>. The question mark at the end indicates the beginning of parameters to follow. The two required parameters, in this case, are service and request. Setting service=WFS and request=GetCapabilities tells the server to return the capabilities XML document for its WFS service. The final request URL looks like this: <http://falconservices.icl.gtri.org:8080/geoserver/ows?service=WFS&request=GetCapabilities>.
- The second arrow shows the server responding with the capabilities document. This document can be quite complex, depending on the server's capabilities. The figure shows parts we will focus on for the purposes of MOGIE.
 - The first and second elements shown, collapsed, are GetFeature and DescribeFeatureType Operations. They will be discussed later but generally tell the client how to actually make those requests.
 - The third element shown, expanded, is the FeatureType of interest, latest_vessel_position. This tells the client only basic information about the feature, including its name, prefixed with its namespace, and the geographic bounding box to which the spatial data is constrained. Now the client has enough information to query the service further, as shown next.
- The third arrow shows the client requesting more detailed information about the latest_vessel_position FeatureType listed in the capabilities document. It does this using the DescribeFeatureType request. By setting the request=DescribeFeatureType and the typeName=mogiemaritime:latest_vessel_position, the client is asking the server for the data structure of latest_vessel_position.

- The final arrow shows the server responding with the GML application schema of latest_vessel_position. The client now knows exactly what to expect when it actually receives the requested data. (Note: part of the XML is collapsed to save space) The information contained here includes the field names and types, which will be used, among other things, for filtering the data in subsequent requests. The schema itself can also be used to validate the actual Features when retrieved.

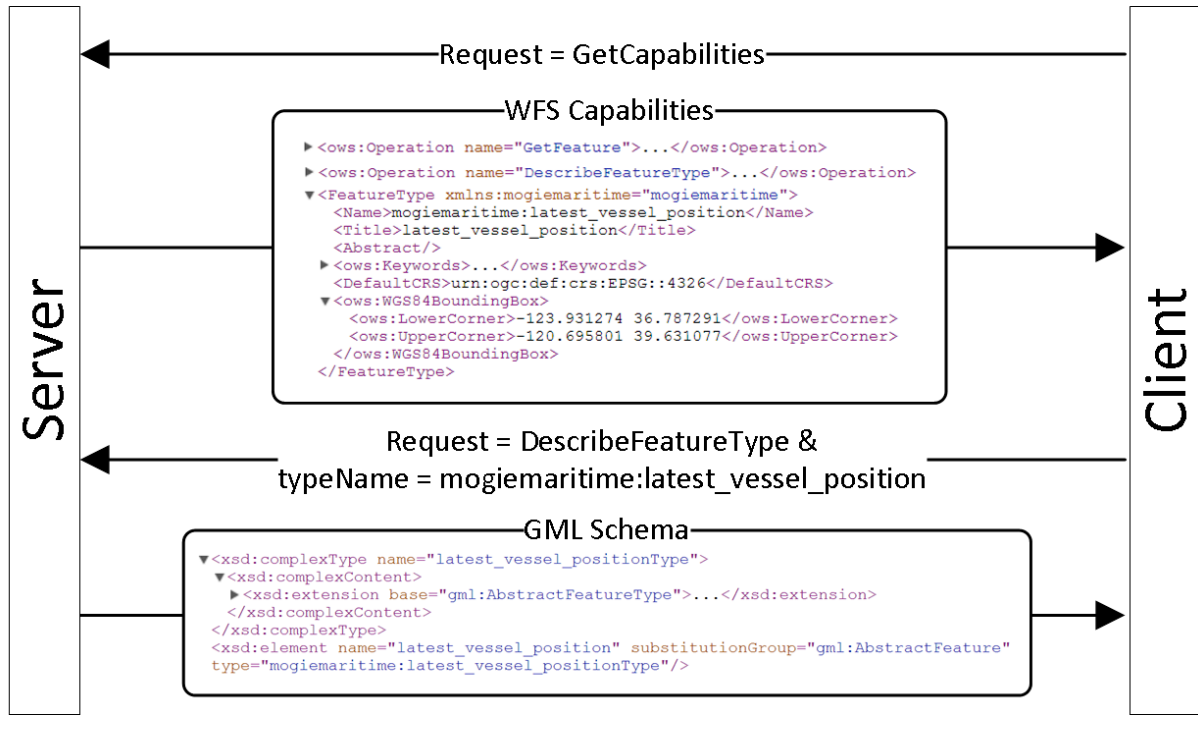


Figure 21: Data Retrieval Process from WFS

Figure 22 shows the process to retrieve the actual data from the WFS.

- The first arrow shows the client issuing a GetFeature request for the latest_vessel_position GML data. This is the simplest form of the GetFeature request. A more complex GetFeature request could include spatial, temporal, numeric, etc. filters to limit the returned data.
- The second arrow shows the server responding with the GML Feature data. It should be noted, GeoServer supports multiple output data types, not just GML. If one preferred the returned data to be in Comma Separated Values (CSV) format, they could set outputFormat=csv in the GetFeature request. However, by not including that parameter, GML is returned by default.



Figure 22: Process to retrieve data from the WFS

The basic process of discovering and retrieving data via WFS has been shown. However, WFS has many capabilities, some of which were enabled but not used in the end. For example, data can be inserted, modified and deleted all with WFS operations similar to those shown above. The developer tested these operations, initially, in case they found a use as the experiment evolved. While they did not, it should be noted to GeoServer's credit, that they did function correctly.

8.2.2 WMS and KML

MOGIE also supported the use of KML (formerly known as Keyhole Markup Language) as an output format. Originally designed for Google Earth, KML 2.2 was adopted as an OGC standard in 2008. Essentially, KML is capable of embedding vector data and styling instructions in the same XML document. GeoServer uses WMS to deliver KML, which allows the client to specify which style they would like embedded in the KML document. As with WFS, both WMS and KML can become quite complex. This section only focuses on the very basic use of both, as they were used by MOGIE.

Figure 23 illustrates the process of retrieving KML via WMS, as supported by GeoServer.

- The first arrow shows the familiar GetCapabilities request from the client.
- The second arrow shows the server responding with the WMS capabilities document. Here, only the two important parts of the document are shown.
 - The left shows the GetMap request. This has a function similar to the GetFeature request in WFS, except the GetMap parameters are more focused on the styling of the features and dimensions for image output formats. The part most interesting to MOGIE is the Format element list. This lists the output formats that the GetMap request supports. In the list we see "application/vnd.google-earth.kml+xml", which is discussed shortly.

- The right shows the latest_vessel_position Layer element. This tells us, among other things, the bounding box to which the spatial data is constrained, the supported coordinate systems and the accepted styles.
- The third arrow shows the client making the GetMap request. The layers parameter tells the server that the latest_vessel_position layer is what should be returned. The format parameter tells the server that the returned data should be encoded in KML. It should be noted that the WMS GetMap request has more required parameters than shown, however, these aren't immediately useful in this explanation, so are left out for clarity.
- The final arrow shows the server responding with the requested KML document. The KML encodes each vessel position report as a Placemark. The image shows just one as an example.
 - The name and description elements tell the client what information the user might find relevant about this particular placemark. If the description element were expanded, it would show all of the vessel position attributes and their values as an HTML table.
 - The Point element simply shows the latitude and longitude of the vessel position.
 - The TimeSpan element shows the time the position report was received, <begin> and the time the next position report was received, making this one no longer valid, <end>.
 - In the context of the client rendering this Placemark, the TimeSpan indicates the window of time that the position should be shown in a replay of all of the Placemarks. By setting these correctly, a replay of all of the position reports, for a single vessel, visualizes old positions disappearing as the next one appears. This is how the replay requirement was achieved with KML.
 - On the server side, the TimeSpan start element is mapped to the latest_position_time field and the end element is mapped to the time_stale field, discussed in Section 8.1.1.1. In GeoServer, this is done by creating a file called time.ftl in the same directory as the layer definition file. The file contains a single line that defines the mapping: “\${latest_position_time.value}||\${time_stale.value}”.

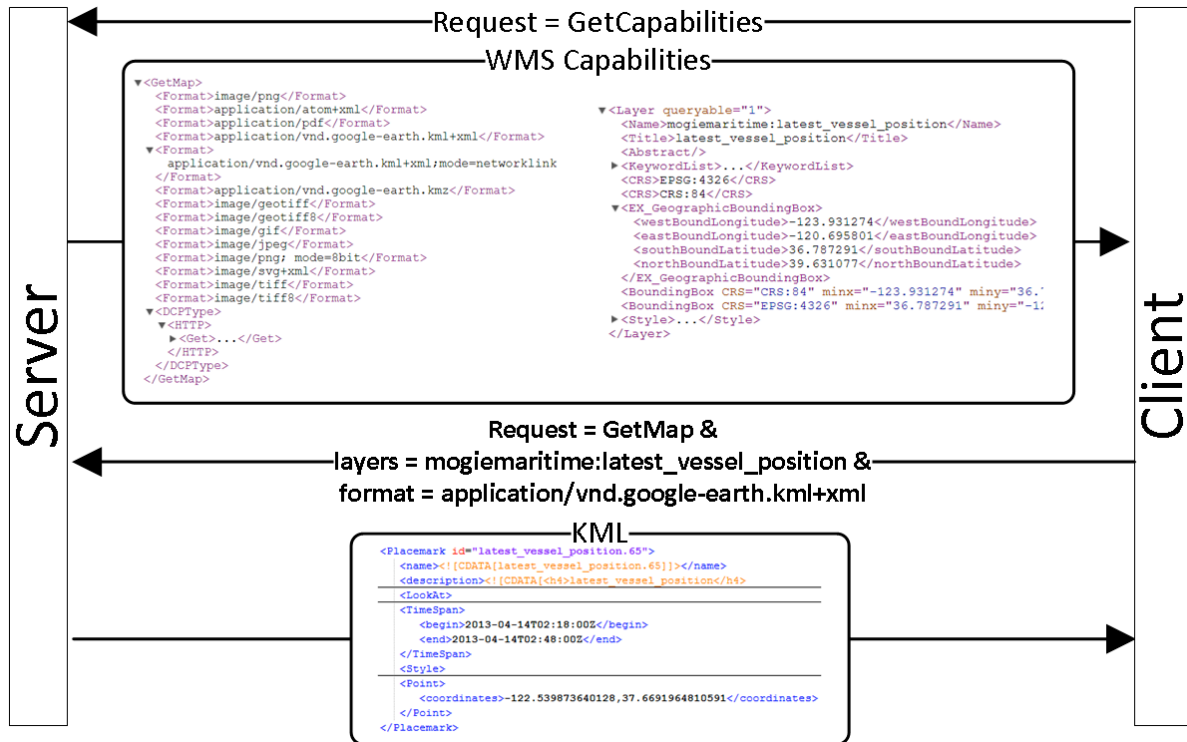


Figure 23: Process of retrieving KML via WMS

8.3 Data Consumption

The following chart provides an overview of each of the participant and what they contributed to MOGIE.

Organization	Contribution							
	Desktop Solution	Mobile Solution	Browser Solution	Maps	GIS Server	Data Source	Land	Maritime
Envitia		X	X	X			X	X
ESRI	X			X			X	X
exactEarth						X		X
FalconView	X			X			X	X
GTRI					X	X	X	X
Luciad	X	X		X			X	X

Figure 24: Participant Contribution to MOGIE

The following is a more detailed description of each participant’s involvement and overview of their consuming applications.

8.3.1 Envitia Services Browser and Mobile Client

Envitia is a specialist geospatial company which delivers technology into military and government systems worldwide. This includes providing key geospatial components into systems such as the USAF Rivet Joint SIGINT Aircraft (via L3 Communications), DSGS Aeronautical Segment (via Lockheed Martin) as well as into many other US military programs. Envitia software support OGC interfaces as well as US Military Symbology and so provides domain specific client implementations which are directly relevant to the MOGIE and NIEM target audience.

In support of the MOGIE experiments Envitia demonstrated how its COTS components, already deployed in an existing Spatial Data Infrastructure (the UK Hydrographic Office Defense Maritime Geospatial Services) could exploit the MOGIE experiment services and how this allowed delivery through both light clients (browser solutions as well as via mobile).

The specific benefit of exploitation of the MOGIE services demonstrated by Envitia related to the ability to coordinate delivery of a wide range of situational awareness information to a range of user types in a crisis situation. A high level coordinator, using an Envitia browser application, pulled together data from different OGC compliant services and assembled a situational awareness picture that could be distributed to a wide range of users including those on mobile.

8.3.2 Overall Architecture of the Exploitation Subsystem

Envitia has provided a number of components which deliver MOGIE Services visualization (shown below).

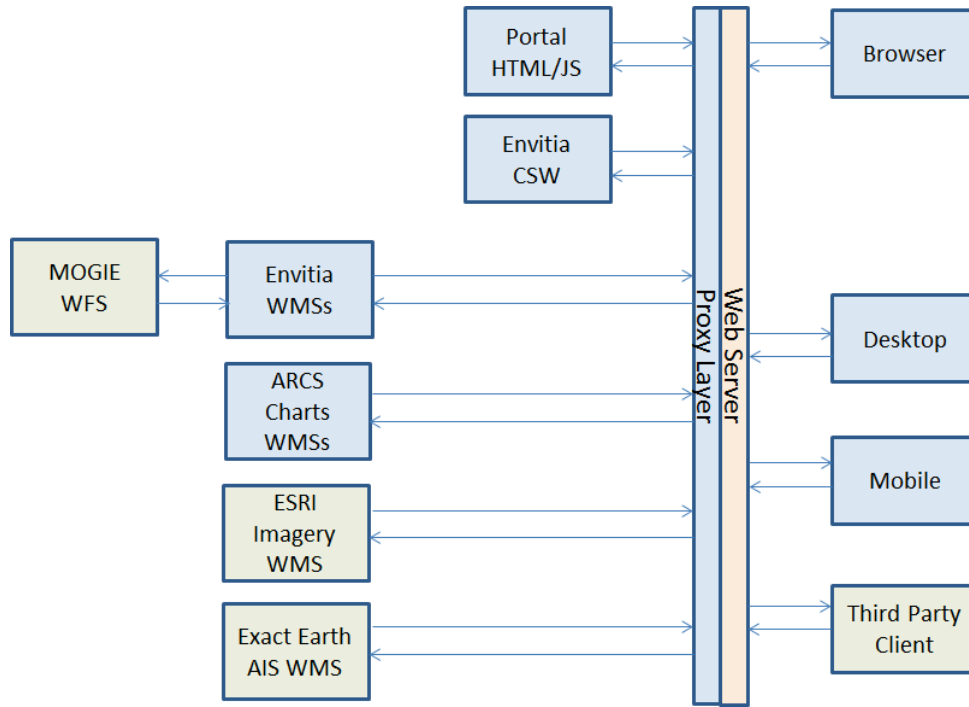


Figure 25 - Envitia Exploitation Sub-system

8.3.3 Envitia Mediating Web Map Service

The Envitia Web Map Service (WMS) is deployed as a chained service accessing the GeoServer Web Feature Service deployed by Georgia Tech Research Institute as part of the MOGIE Experiment. The Envitia WMS retrieves track information from the WFS and applies 2525D/APP6C¹⁸ symbolization. It then offers the various feature types present in the WFS as WMS Layers in the WMS Capabilities document (retrieved by a GetCapabilities request). The overall architecture/request structure looks as shown in the figure below.

¹⁸ 2525D/APP6C is a draft specification at present. Envitia fully support 2525B/APP6A but for the purposes of the MOGIE project have implemented support for the subset of 2525D required to support the experiment. Our intention is to fully support 2525D within our MapLink Pro/WMS product line when it is formally released.

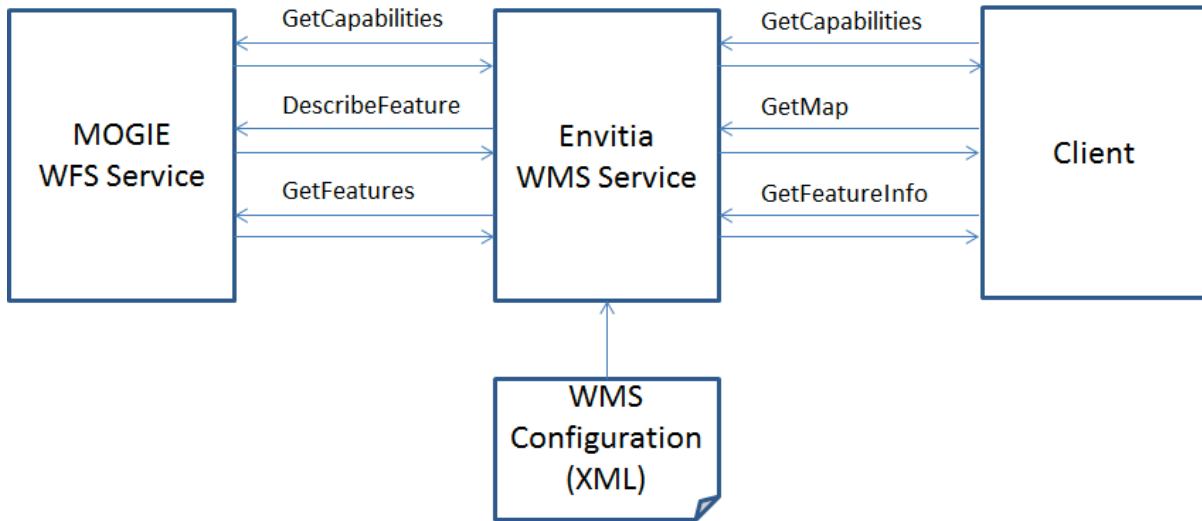


Figure 26 – Envitia Mediating WMS Architecture

The WMS takes an XML Configuration document as input which determines which Web Feature Services it will visualize as well as what symbolization scheme to use (in this case 2525D but S52 and Generic SLD are also supported).

The WMS service also supports result caching of the WFS results which improves performance (particularly response time) by up to an order of magnitude and also reduces the load on the WFS. This can introduce extra latency of a few tens of seconds if an event service is not available alongside the WFS as otherwise the WMS needs to poll for changes. It has been suggested in future work that an event based delivery alongside the WFS is considered which would allow caching without any latency issues.

In order to animate data retrieved via WMS it is necessary to be able to specify a time or time range for a layer. The MOGIE FeatureTypes do include time properties but one issue is how to determine exactly which one to use. So the assumption is that a-priori knowledge of which property to animate against is possible, and as a result the property to use for each feature is stored in the configuration file of the WMS.

In terms of the mechanism used to access time dependent WMS data, this is fairly well defined at one level but there are some additional semantics which need to be considered. Time axes are a specific case of the general ‘Dimensions’ concept present in the WMS Specification. But a dimension is a relatively open concept.

Envitia has used two specific methods of specifying time in WMS Dimension statements:

- Ordinal time Dimension. In this case the WMS Capabilities provides a list of valid supported times (in our case these are monthly timeslots used to select climatological data).
- Continuous Time Dimension. In this case the WMS Capabilities provides a start, end and interval value for the time dimension. If the interval is 0 any time between the upper and lower time extents is valid.

For the MOGIE work the continuous time dimension was adopted. The dimension definition for a layer in the WMS Capabilities document is typically as follows:

```
<Dimension name="time" units="ISO8601" default="2013-04-13T18:00:00Z">2013-04-13T18:00:00Z/2013-04-14T03:48:00Z/0</Dimension>
```

(Note the interval here is zero).

8.3.4 Envitia Catalogue Service and Supporting Services

The Envitia OGC Catalogue Services for the Web (CSW) services provided a supporting capability in the exploitation subsystem. Envitia’s catalogue can accept both services delivered in the sub-system (these were for example web map services using data from the UK Hydrographic Office) and external services (in this case the MOGIE WFS via the Mediating WMS, the Esri Imagery WMS and the Exact Earth AIS WMS) and provide the user with an extensive search capability to discover and evaluate the various services together. As well as the CSW and the foundation chart WMS Services, DMarGS also provided a Tidal Data WMS, a Climatology Data WMS/WPS and a Gazetteer service.

8.3.5 Envitia HTML/Javascript Portal

The Envitia JavaScript/HTML portal is designed to operate on a wide range of browsers.

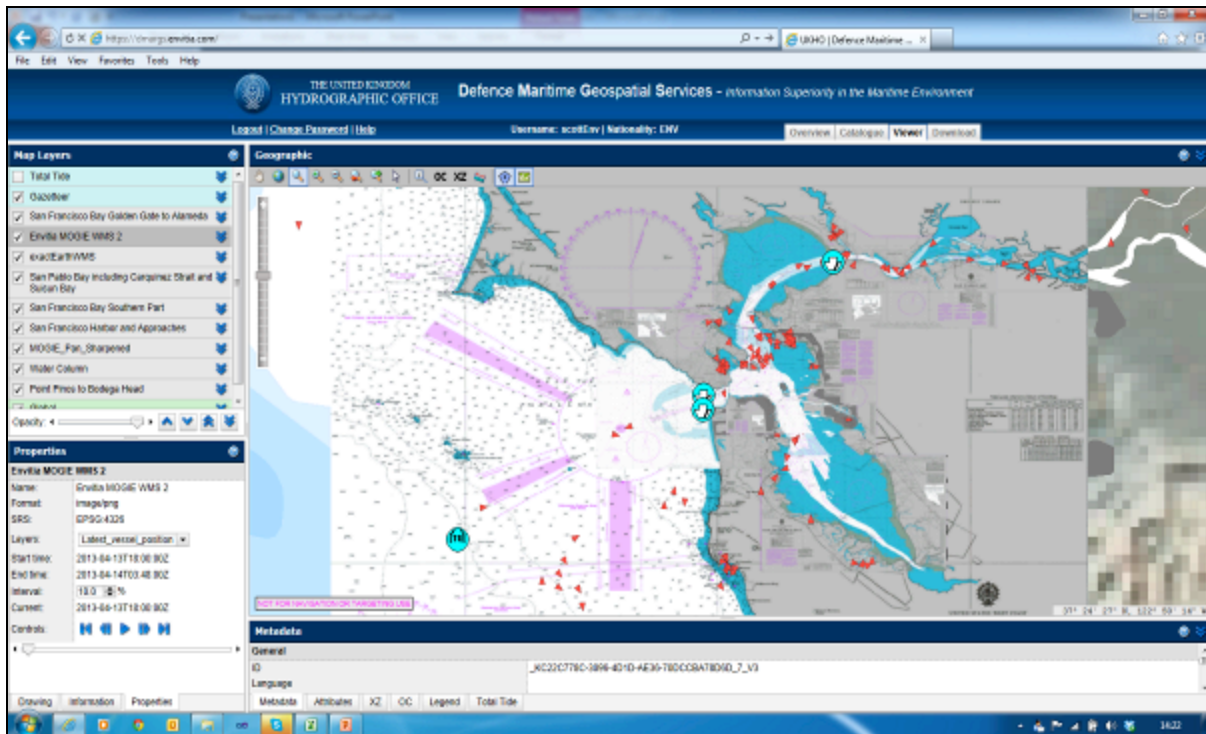


Figure 27 - Envitia Portal showing MOGIE data

The portal interacts with the catalogue service to identify resources it can exploit and then accesses those services to provide a wide range of functionality. The Envitia Portal does not require any form of plug-in, and is implemented completely in HTML/Javascript.

8.3.6 Envitia Browser Portal

Envitia also exploited its MapLink Pro Mobile developer kit to develop an Android based app capable of exploiting both the MOGIE and other DMarGS services and deliver a common operating picture both on-line (using the OGC OWS Context Standard) and Off-line (using the OGC Geopackage standard).

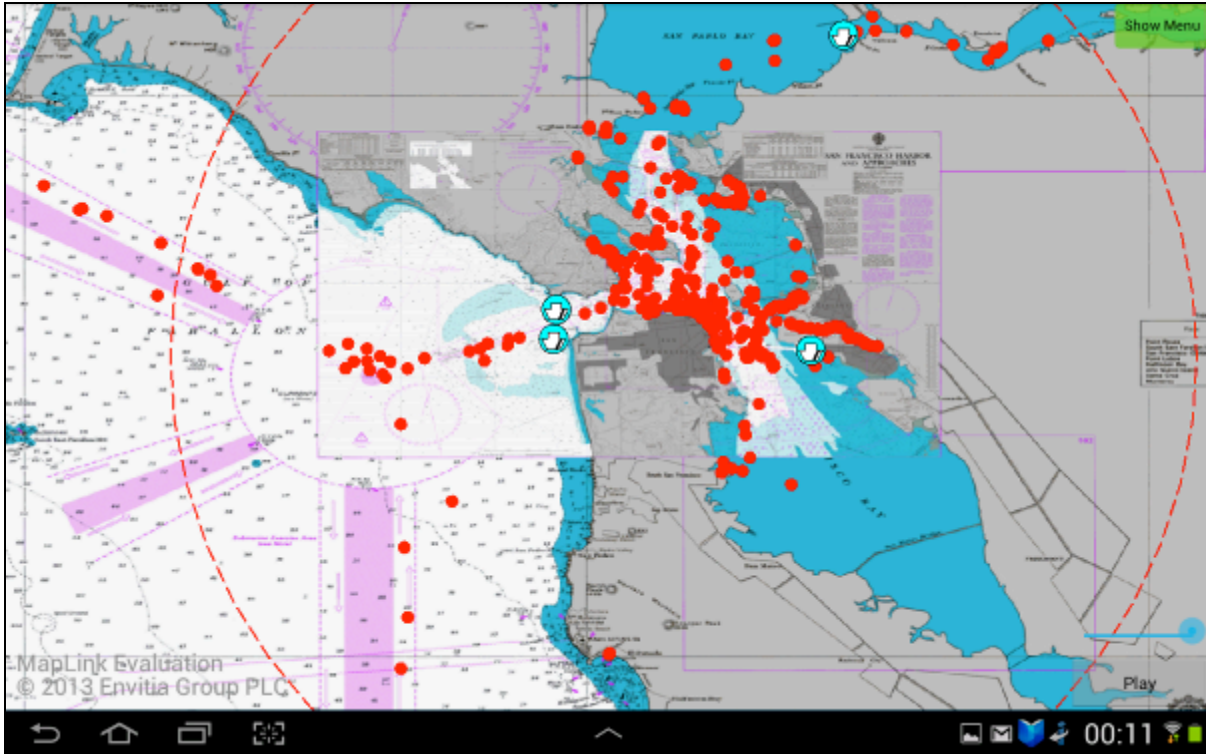


Figure 28 - Envitia MapLink Pro Android Application

8.3.7 Esri ArcGIS Desktop

Esri is an international supplier of GIS software with many government users. Their ArcGIS software suite is heavily focused on interoperability and supports numerous OGC standards as a result. ArcGIS was used to demonstrate the land scenario from an existing client, namely ArcMap, which required no additional software development. The interoperability extension was used to configure and connect to the WFS service and interpret the returned GML. The tracking analyst extension's playback feature was used to replay the scenario data.

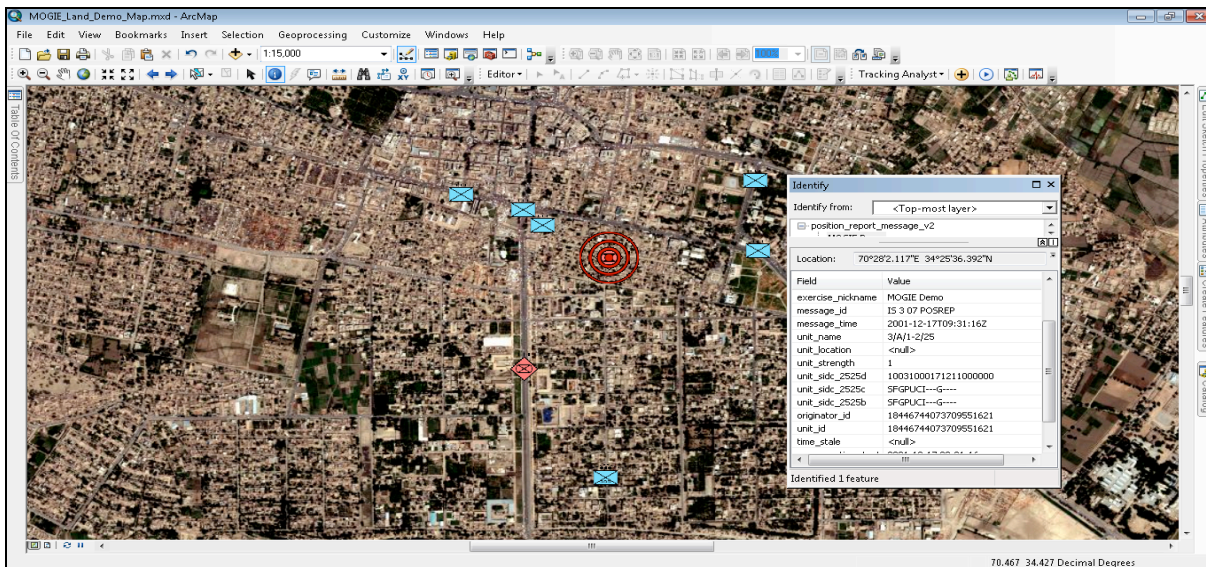


Figure 29: Land scenario in ArcGIS

Base Map

The background imagery was accessed via the WMTS service described in Section 8.4.1. ArcGIS Desktop can connect to WMS, WMTS and WCS services natively via ArcCatalog. The imagery layers can then be viewed in ArcMap.

WFS – Interoperability extension

ArcGIS Data Interoperability enables data sharing by providing direct data access; data translation tools; and the ability to build complex spatial extraction, transformation, and loading (ETL) processes. ArcGIS Data Interoperability allows one to use any standard GIS data within the ArcGIS for Desktop environment for mapping, visualization, and analysis. The Workbench application, included with the extension, enables one to build complex spatial ETL tools for data validation, migration, and distribution.

Using ArcGIS Data Interoperability, one can directly read more than 100 spatial data formats, including GML, XML, WFS, Autodesk, DWG/DXF, MicroStation Design, MapInfo, MID/MIF and TAB, Oracle and Oracle Spatial, and Intergraph GeoMedia Warehouse, and export to more than 70 spatial data formats.¹⁹

Symbology

The MIL-STD-2525D symbology displayed was manually added using ArcMap's Symbology editing tools found in the layer editor like any other layer in ArcMap. The set of PNG symbols were pre-rendered for the MOGIE experiment.

Playback

ArcGIS Tracking Analyst extends the time-aware capabilities of the ArcGIS system with advanced functions to view, analyze, and understand spatial patterns and trends in the context of time. By providing tools for time-dependent symbolization and time-based analysis, ArcGIS Tracking Analyst automates and enables the tracking and discovery of time-related trends and patterns.²⁰

8.3.8 FalconView Desktop

FalconView is a mapping system created by the Georgia Tech Research Institute.(GTRI) It displays various types of maps and geographically referenced overlays. Many types of maps are supported, including aeronautical charts, satellite images and elevation maps. FalconView also supports a large number of overlay types that can be displayed over any map background including KML, GML, AIS, and NMEA. FalconView 5.1 was used to demonstrate the maritime scenario as an existing client that required no additional software development.

The FalconView KML overlay was used to interpret the KML returned from the server. While FalconView does have WMS client capability, this is designed for raster and imagery output formats. However, the KML overlay supports retrieving data from a URL. So to retrieve the data, a WMS GetMap request was predefined and passed as the URL to the KML overlay. The playback tool was used to replay the scenario data. Figure 30 shows FalconView rendering the maritime vessels while displaying the vessel information of the selected Coast Guard ship.

¹⁹ <http://www.esri.com/software/arcgis/extensions/datainteroperability>

²⁰ <http://www.esri.com/software/arcgis/extensions/trackinganalyst>

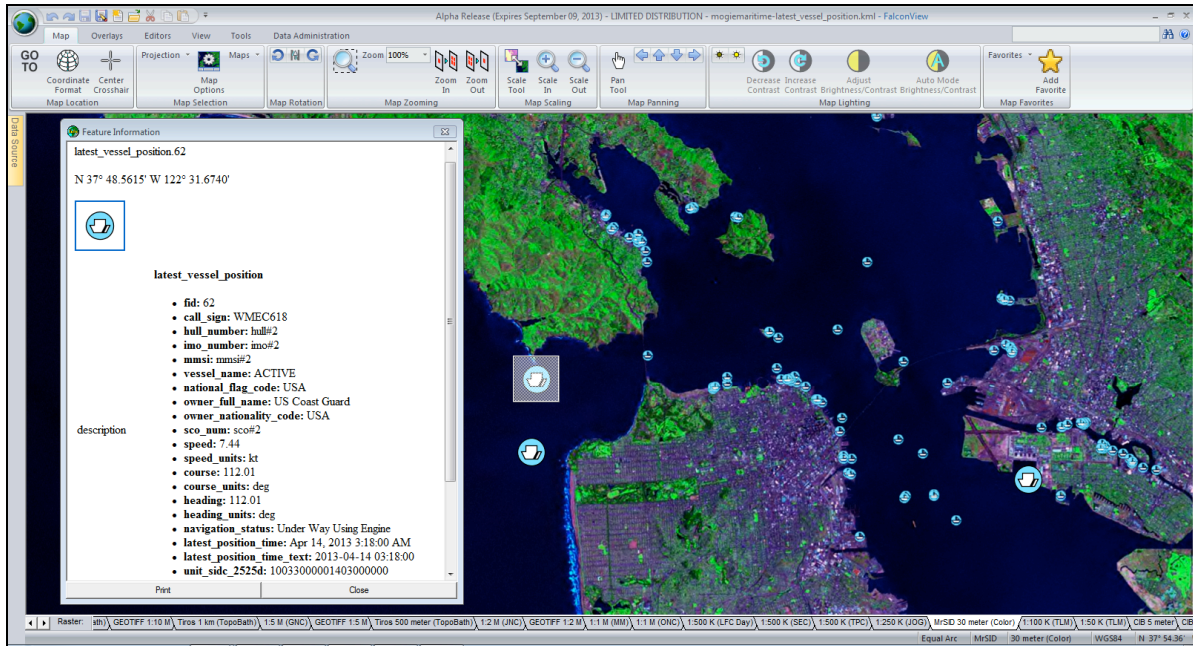


Figure 30: Maritime scenario in FalconView

The base map is Landsat imagery provided by NASA in MrSID format. This and many other types of imagery are typically loaded into FalconView’s data manager to be organized and stored onto the local hard drive. Spatially relevant imagery sources can then be easily swapped to quickly compare the vector data on multiple base maps. As an example, this would allow attack response coordinators to quickly view the vessel data overlaid on satellite imagery as well as nautical charts.

FalconView loads and replays the data as KML, using the method described in Section 8.2.2. The KML overlay supports most KML features including regions, network links, styles, timestamps and timespans. While FalconView does support common military symbology standards, such as MIL-STD-2525, the styles specified in the KML document were used for MOGIE.

Because the KML overlay extends the broader FalconView overlay software architecture, other commonly used overlay tools are available for KML datasets, such as feature playback. An example is shown in Figure 31.

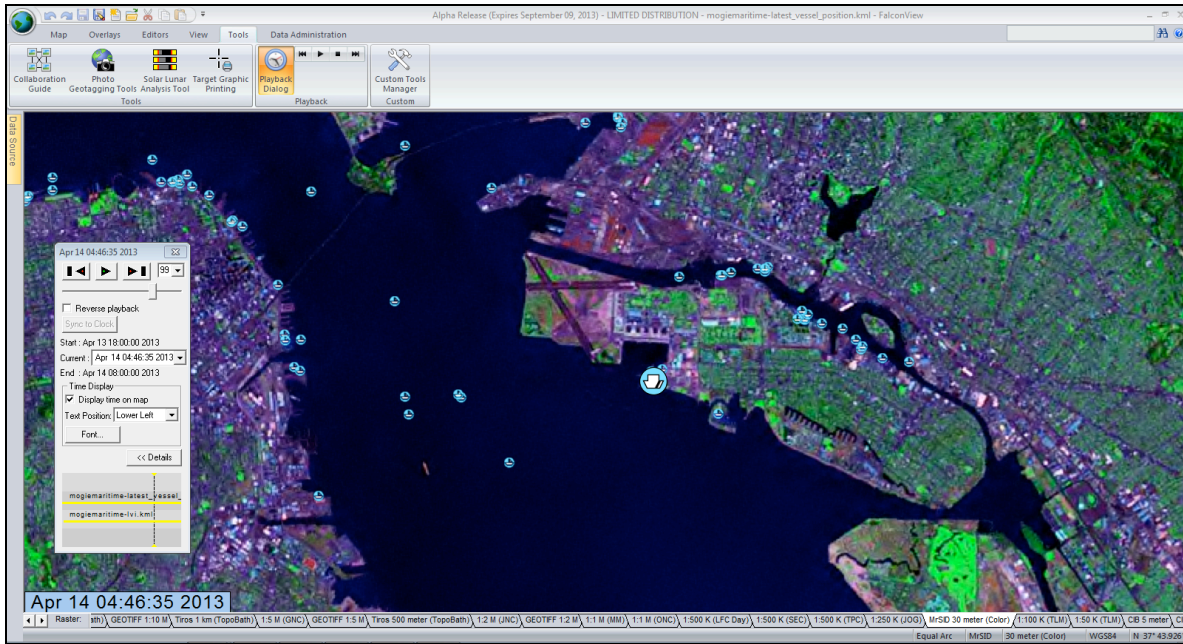


Figure 31: Playback Dialog in FalconView

The playback tool can be used with any supported temporally enabled vector format including AIS, NMEA, GML, and KML. In the case of MOGIE, FalconView uses the timespan and timestamp elements defined in the KML to determine when the appropriate features must be rendered.

8.3.9 Luciad Services Desktop and Mobile Client

To support MOGIE and the scenario experiments and demonstrations, Luciad contributed a set of components based on its SCOTS product suite.

Luciad is a software product company providing standards-based software components to develop situational awareness applications in the Aviation, Defense & Security, and Maritime domains. Luciad has been an active supporter of OGC and OGC standards for many years. Luciad products implement more than a dozen OGC standards and candidate standards.

One important outcome for Luciad’s participation in the MOGIE project is the major reuse of existing components because of the adherence to OGC standards. With almost no development effort, it was possible to setup a situational awareness display for both scenarios on different devices (desktop, mobile), and providing users with a set of analysis tools (time filtering, precision measuring, historic trajectory retrieval ...) to support the scenario actions.

The following sections further discuss the contributed components and their use in the experiment.

8.3.9.1 Luciad Desktop client

For both the land and maritime scenario experiment, Luciad contributed a desktop situational awareness client based on its COTS product LuciadLightspeed. LuciadLightspeed provides software components and functionalities that enable data fusion, visualization and analysis of geospatial information. This can include static and moving data, maps, satellite imagery, and terrain elevation in many different formats and references.

The characteristics of the contributed application include:

- Hybrid 2D & 3D map-centric situational awareness display

- Fusion and visualization of different data sources:
 - MOGIE vector data through GTRI's OGC Web Feature Service
 - SRTM elevation data through Luciad's OGC Web Coverage Service
 - Satellite imagery through Microsoft's Bing Maps
 - OpenStreetMap data through Luciad's OGC Web Feature Service
 - NOAA ENC data through Luciad's OGC Web Map Tile Service
- MIL-STD-2525D styling of the defense-related MOGIE vector data (i.e., the land scenario data + the military vessel data of the maritime scenario)
- Temporal filtering / previewing of time-dependent MOGIE vector data

Highlights of the experiments and demonstrations:

- Analysis of the MOGIE land scenario precision data: the distance between the P6, P9 and P13 data was determined through a distance ruler and visually displayed on the map at a centimeter scale.
- Provision of situational awareness aids to ease analysis of the MOGIE maritime scenario data:
 - Automatic highlighting of all latest vessel positions when hovering over a vessel.
 - Realistic ship icon based on vessel type information.
 - HMTL-based labeling showing all relevant information of a selected vessel:
 - Origin country + flag icon
 - Destination
 - Navigational status
 - Ship type
 - Call sign and MMSI
 - User controls to retrieve and visualize the historical track (30 days) for a selected vessel, by using the historical track data provided by GTRI's OGC Web Feature Service.

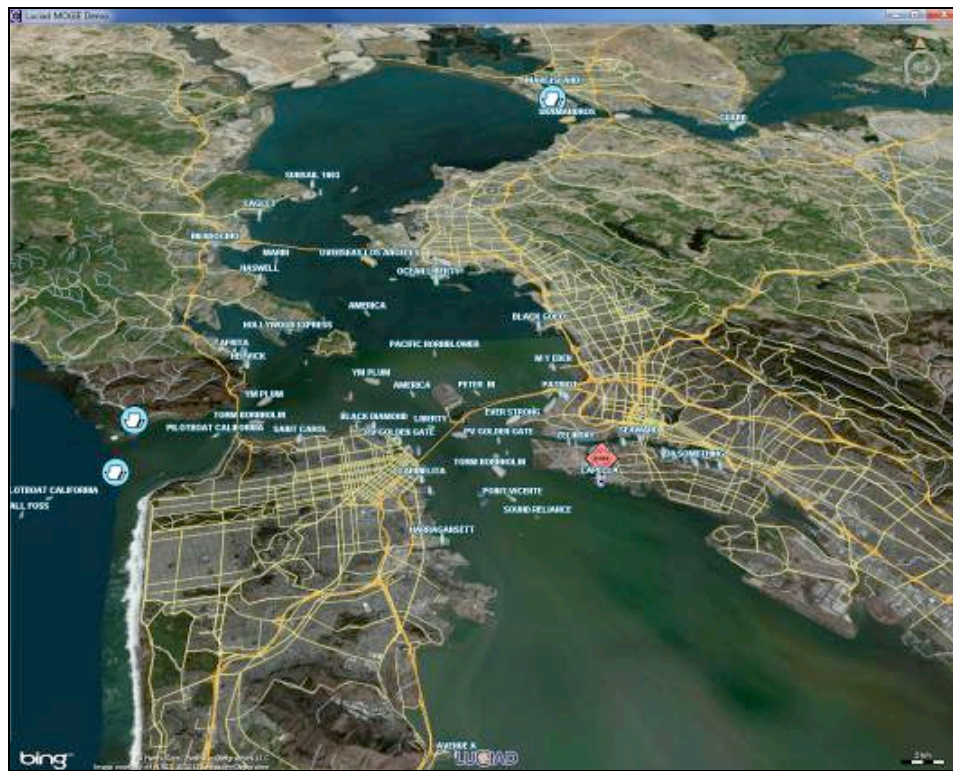


Figure 32: Military and civilian vessels in the Luciad Desktop client

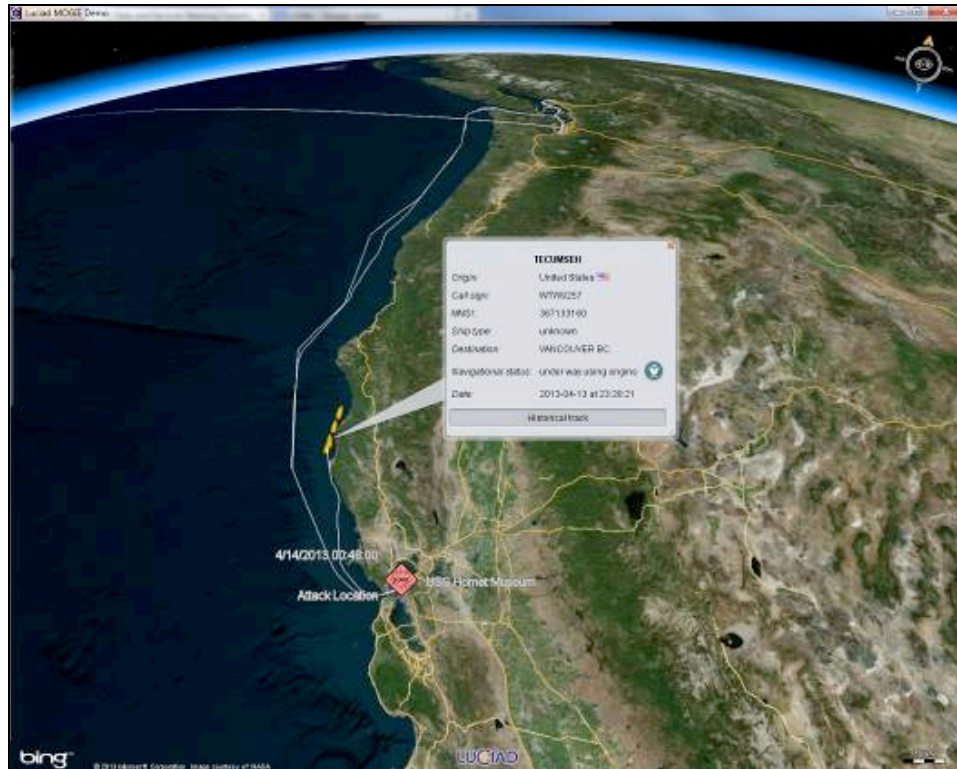


Figure 33: Vessel analysis and querying its historical track (shown by the white line)

8.3.9.2 Luciad Mobile client

For the land scenario experiment, Luciad contributed a mobile application based on its COTS product LuciadMobile. LuciadMobile enables rapid development of geospatial situational awareness applications for mobile devices running on the Android operating system.

The characteristics of the contributed application include:

- Touch-centric situational awareness display for Android-based smartphones and tablets
- Consumption of the MOGIE land scenario data
 - Reading of MOGIE land scenario data from OGC GeoPackage
 - Visualization with MIL-STD-2525D symbology
 - Info panel showing properties of selected MOGIE features
- Visualization of background satellite imagery through Microsoft's Bing Maps
- Map annotation capabilities
 - Draw arrows / areas on the map
 - Store in OGC GeoPackage for persistency and sharing with other users.

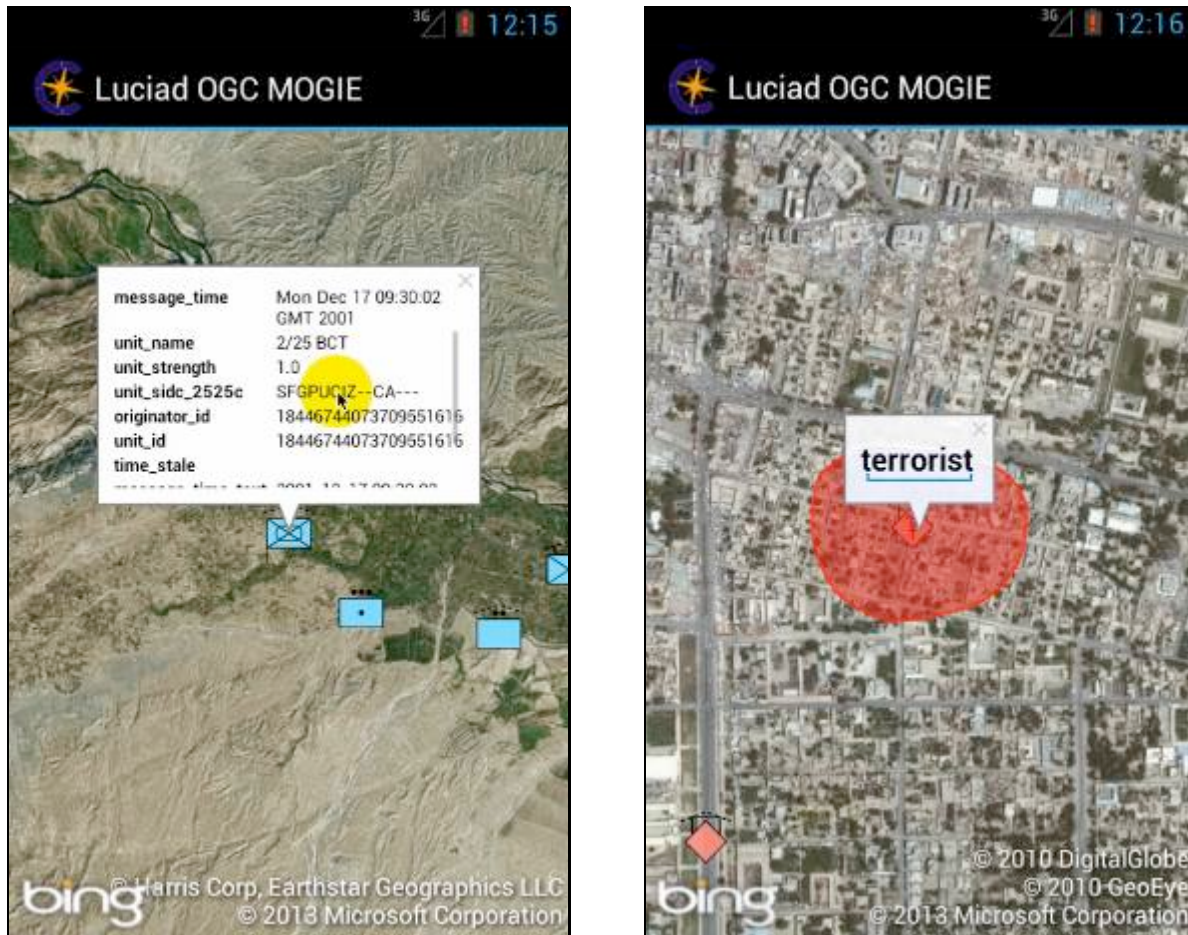


Figure 34: Accessing MOGIE land scenario data and making map annotations on the Luciad Mobile client through OGC GeoPackage

The main highlight of the mobile client experiment and demonstration is the interoperability test with OGC GeoPackage, a new vector and raster storage format being developed by OGC and the Army Geospatial Center with a focus on use by mobile devices.

For the test, the MOGIE land scenario data was stored in an OGC GeoPackage file; this OGC GeoPackage file was read by the Luciad Mobile client and the MOGIE data was made accessible to the user: by visualize the features on the map using MIL-STD-2525D symbology and by making the properties of each feature accessible in an info panel. One interesting benefit of the integration with the OGC GeoPackage format in this test was the ability for the user to annotate the data: the Luciad Mobile client includes capabilities to create map annotations (arrows, areas), which were stored in the same OGC GeoPackage file, next to the MOGIE data. Being an interoperable, standardized data format, this proves to be an effective way to share and annotate operational land data in a mobile environment.

8.3.9.3 Luciad OGC services

Luciad internally deployed a number of OGC services to ingest external background data sources into the contributed applications. This includes:

- An OGC Web Map Tile Service, providing access to the NOAA ENC data for use as background data in the maritime desktop client. The source NOAA ENC data was provided in the IHO S-57 format. This

data was fed into Luciad's COTS product LuciadFusion, which processed it into a tiled, multileveled raster pyramid and made the resulting data accessible through an OGC Web Map Tile Service in the PNG format.

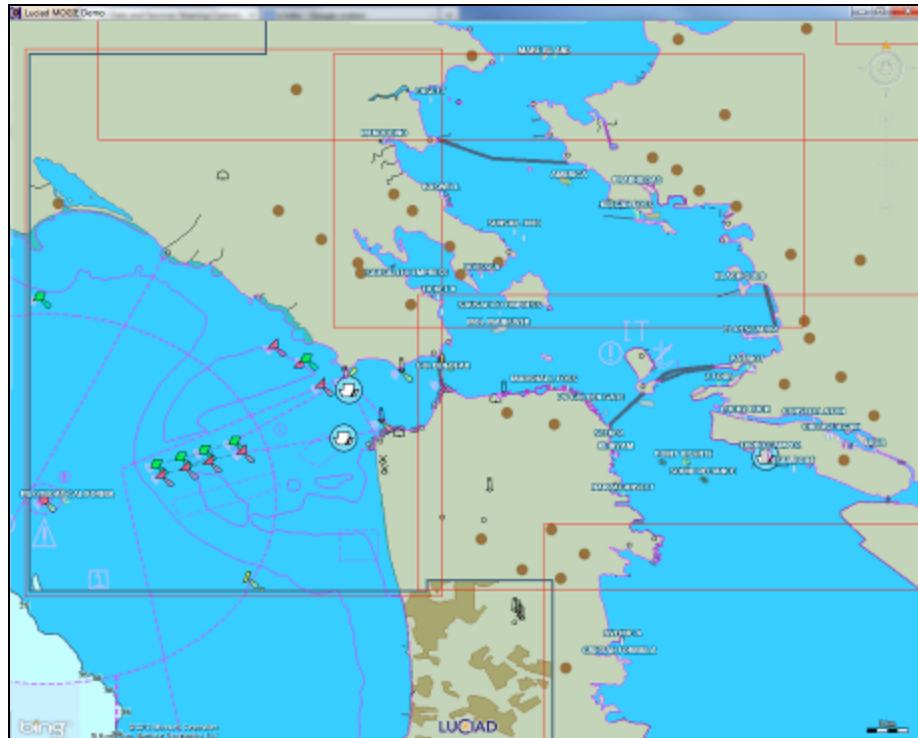


Figure 35: NOAA ENC data accessed in the Luciad Desktop client

- An OGC Web Coverage Service, providing access to SRTM elevation data for use as terrain data in the land and maritime desktop clients. The source data was provided in the DTED format. Through LuciadFusion, the data was made accessible through an OGC Web Coverage Service in the GeoTIFF format.
- An OGC Web Feature Service, providing access to OpenStreetMap data for use as background data in the land and maritime desktop clients. The source data was provided in the SHP format. Through LuciadFusion, the data was made accessible through an OGC Web Feature Service in a GML 3.2 based format.

8.4 Supporting Services

8.4.1 Esri ArcGIS Imagery Server

In support of MOGIE, Esri supplied imagery services for the land scenario. ArcGIS Server 10.1 was used to host high resolution pansharpended true color imagery of the Jalalabad area in WMS and WMTS service formats. Examples of the imagery can be seen in Figure 29: Land scenario in ArcGIS.

8.4.2 exactEarth AIS WFS / WMS

exactEarth provided MOGIE with access to their real-time global maritime data via exactAIS Geospatial Web Services.

exactAIS Geospatial Web Services delivers maritime vessel information, derived from Satellite and Terrestrial AIS sources, directly into your geospatial platform of choice. Certified to OGC standards, we now offer a web services solution that allows instant access to our exactAIS® data in an on-demand environment.²¹

Specifically, all data described in Section 8.1.2.2 was recorded directly from the WFS service provided by exactEarth.

²¹ <http://www.exactearth.com/products/exactais-geospatial-web-services/>

9 Findings

- **Experiment #1:** Employ “GML Validator” currently being developed in OWS-9 as appropriate to determine compliance of GML in a MilOps exchange to GML Encoding Specifications, WFS, WMS, etc.

Finding #1: The OGC GML Validator TEAM Engine and ETS-GML were employed to verify MOGIE produced a valid GML Application schema. Appendix H and I provide XML examples to show GML can be embedded in a NIEM conformant data exchange and transformed into an OGC conformant format (e.g., WFS, WMS, etc.). Other examples include the land-based position report, call for fire, and observation report and are available upon request within the limits of ITAR and CUI restrictions.

- **Experiment #2:** Extract IEP content including geospatial data that includes GML and transform it into an OGC Standard format (e.g., WMS, WFS, etc.) with military symbology as appropriate, and then display the data on a client.

Finding #2: GML and other data was transmitted in NIEM conformant XML, stored in a database, and delivered by OGC conformant web services in WFS/WMS for display on several clients. Four vendor products were used to display the data in web browsers, desktop applications, and mobile devices. Symbology from MIL-STD-2525D was used in the data sets and was displayed visually on client applications in web browsers, desktop applications, and mobile devices.

- **Experiment #3:** Extract geospatial data that includes GML content and add additional NIEM attributes (e.g., MilOps content specified as a feature) and then display the data on a client.

Finding #3: GML and other data was transmitted in NIEM conformant XML, stored in a database, and delivered by OGC conformant web services in WFS/WMS for display on several clients. Four vendor products were used to display the data in web browsers, desktop applications, and mobile devices. This demonstrated data can be read by clients without a priori access/knowledge of the data. Refer to GEO4NIEM Engineering Report on the OGC portal for additional details and best practices for embedding GML in NIEM conformant XML.

- **Experiment #4:** Experiment #4: Demonstrate no loss of precision or accuracy when transforming GML content (e.g., location) embedded in a NIEM conformant IEP. Expose data from a NIEM conformant IEP transformed into a GML application schema compliant form deliverable via a Web Feature Services interface and make the GML content available for vendor tools to display on a client.

Finding #4: Based on the MOGIE results documented in Appendix G, NIEM XML does not change numerical data being exchanged; therefore NIEM had no impact on the accuracy and precision of position data exchanged in MOGIE.

- **Experiment #5:** Demonstrate implementation of the January 2013 draft version of MIL-STD-2525D

Finding #5: Envitia and Luciad independently demonstrated implementation of the Jan 2013 draft of MIL-STD-2525D in their client applications. Both reported no issues were encountered. They also commented that the change to a binary identifier and the simplification of the symbol rules (e.g., isolating center and frame symbols to specific bit groups) was a significant improvement over previous versions of the standard, which made it easier to support.

- **Additional Findings:**

Finding #6: Section 6.6.5 describes the developer skills and experience working on MOGIE, which indicated using NIEM XML did not require any specialty skills. Commodity software

development and XML skills provided a sufficient base to learn and use NIEM from training and materials available publicly online.

Finding #7: The bit level storage requirements and binary structure prescribed by IEEE 754 Standard for floating-point arithmetic introduced limits on accuracy and precision. Details are provided in Section 6.6.6.

Finding #8: Minor limitations of the OGC validator tool added some time to experiment #1's completion. However, the accessibility and flexibility built into the validator's architecture and software design provided the means to accomplish the validation. More information is provided in Appendix C.

Finding #9: An evaluation of the OGC GML validation tool produced three recommended changes:

1. Provide Authentication Support
2. Provide Better Internal Error Reporting
3. Provide additional Source Code Comments in TEAM Engine

Appendix C contains the details of the OGC validator tool evaluation.

Finding #10: Supporting graphics, not the data, set a practical limit of accuracy that can be distinguished on a graphical display.

APPENDIX A: Geolocation Accuracy and Precision in NIEM

This paper was originally written in the context of C2 Core in December 2012, prior to the start of MOGIE. The paper was updated in a NIEM context in May 2013 after the DoD NIEM adoption memo (see Appendix D) was released. This was possible because the C2 Core v2.0, NIEM v2.1 and NIEM v3.0 NDR prescribing the “adapter pattern” were identical.

Geolocation Accuracy and Precision in NIEM¹

This paper demonstrates that a NIEM-conforming data exchange can represent geolocations with the accuracy and precision demanded by “precision applications” such as targeting and safety of navigation. Our method is to show the equivalence between the geospatial information content in a GML document and in a NIEM information exchange package (IEP), treating GML as the gold standard because it is approved for such uses. We conclude that the use of NIEM introduces no problems so far as the accuracy and precision of the information content are concerned.

Here is a GML document containing a precise geographic location:

```

1 <gas:Goal
2   xmlns:gas="http://metadata.ces.mil/mdr/ns/example/GoalAS"
3   xmlns:gml="http://www.opengis.net/gml/3.2"
4   gml:id="T01">
5   <gml:description>Dr. Scott's Desk Chair</gml:description>
6   <gas:controlPointLocation>
7     <gml:Point gml:id="P02" srsName="http://metadata.ces.mil/mdr/ns/GSIP/crs/WGS84E_2D">
8       <gml:pos>38.92133 -77.20438</gml:pos>
9     </gml:Point>
10  </gas:controlPointLocation>
11 </gas:Goal>

```

This is a conforming GML document, because it is valid against a conforming GML application schema² which defines the gas:Goal element as substitutable for gml:AbstractFeature. The interpretation of this document is:

- there is a Goal feature
- that feature has a “control point location” property
- the value of that property is a Point using the WGS84E_2D coordinate reference system
- the Point coordinates are latitude 38.92133 degrees North, longitude 77.20438 degrees West, using the WGS84 datum

Those coordinates describe a point within about one meter of my desk chair.³ Observe that the accuracy and precision of this information is in no way guaranteed by the GML specification, but instead depends entirely on the producing application. That is, the application could instead generate a perfectly valid GML document with the same accuracy but less precision. Then we might have

```
<gml:pos>39 -77</gml:pos>
```

¹ National Information Exchange Model. An earlier version of this paper contained examples based on C2 Core. There are no differences between NIEM and C2 Core that are relevant to this paper.

² Provided as an appendix

³ Ignoring elevation, for simplicity.

which denotes a point that we may roughly describe as somewhere within sixty miles of Silver Spring, Maryland. That would be accurate but imprecise. Another perfectly valid document might have

```
<gml:pos>38.88948 -77.03525</gml:pos>
```

which denotes with one-meter precision the location of the Washington Monument. That would be precise but inaccurate. We can see that GML makes it *possible* to provide an accurate and precise geolocation, by rigorously specifying the coordinate reference system and the coordinate format. That is all we can ask of GML. The *actual* accuracy and precision are controlled by the producing application.

Now, here is a NIEM IEP with identical information content:

```
1 <ge:Goal
2   xmlns:gml="http://www.opengis.net/gml/3.2"
3   xmlns:ge="http://metadata.ces.mil/mdr/ns/example/Goal/extension/1.0"
4   <ge:GoalDescription>Dr. Scott's Desk Chair</ge:TargetDescription>
5   <ge:GoalControlPointLocation>
6     <gml:Point gml:id="P02" srsName="http://metadata.ces.mil/mdr/ns/GSIP/crs/WGS84E_2D">
7       <gml:pos>38.92133 -77.20438</gml:pos>
8     </gml:Point>
9   </ge:GoalControlPointLocation>
10 </ge:Goal>
```

This is a conforming IEP, because it is valid against a conforming NIEM information exchange package documentation (IEPD). The interpretation of this document is exactly the same as above. The embedded GML content (lines 6-8) is character for character the same as that found in the GML document above (lines 7-9). The embedded GML is validated against exactly the same GML schema as the GML document. Because there is no difference in the data, the defining schema, or the interpretation, the accuracy and precision of this content must be the same. There is no difference that can make a difference.

As before, the accuracy and precision of the IEP depends on the producing application. There is nothing in NIEM or the IEPD to keep the producer from generating `<gml:pos>39-77</gml:pos>`, or `<gml:pos>38.88948 -77.03525</gml:pos>`. NIEM makes it *possible* to provide an accurate and precise geolocation, and that is all we can ask of NIEM.

Embedded GML in NIEM requires an “adapter”, because GML is an external standard that does not conform to all of the rules for NIEM schemas. When the IEPD designer chose to use `gml:Point` to represent the geographic location of the “goal control point”, he then needed to make use of an adapter type. That adapter type is defined by the following schema fragment

```
1 <xs:complexType name="PointAdapterType">
2   <xs:annotation>
3     <xs:appinfo>
4       <i:Base i:namespace="http://niem.gov/structures/2.0" i:name="Object"/>
5       <i:ExternalAdapterTypeIndicator>true</i:ExternalAdapterTypeIndicator>
```

```

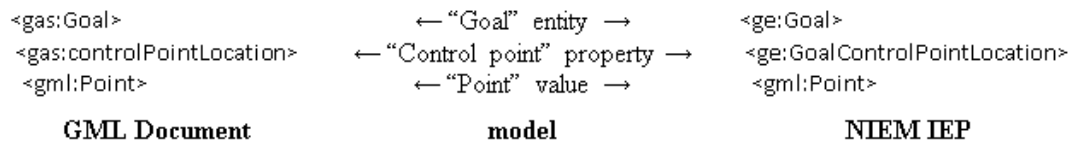
6   </xs:appinfo>
7   </xs:annotation>
8   <xs:complexContent>
9     <xs:extension base="s:ComplexObjectType">
10      <xs:sequence>
11        <xs:element ref="gml:Point"/>
12      </xs:sequence>
13    </xs:extension>
14  </xs:complexContent>
15 </xs:complexType>

```

Part of this type definition is schema metadata (lines 2-8), providing documentation in machine-readable form. It informs developers, and can be exploited by developer tools. It has no effect on the valid data or the validation process. The other part is the component definition (lines 9-15), which has the effect of attaching the attributes defined by default for every NIEM object. Developers usefully depend on finding these attributes as needed in the IEP. (For example, the `xsid` attribute allows any element to serve as the target of an IDREF association, and the `s:metadata` attribute permits common metadata such as “valid time” to be represented and retrieved by consuming applications in a standard way.)

Nothing about this adapter type changes the validation process. A single schema (composed, as usual, by importing several schemas for several namespaces) is used to assess the validity of the entire document. There is no special “NIEM processor” and no need to separately validate the embedded GML within the adapter element.

Nothing about this adapter type requires an “extra” wrapper element. Every element in this document has a normal and customary interpretation. In fact, the element hierarchy is the same in the GML document, as illustrated below:



Any GML element⁴ can be embedded within a NIEM IEP in the same way, with no difference in data, defining schema, or interpretation, and thus no difference in accuracy and precision. The only thing necessary is to use an adapter type whenever the IEPD designer decides to use GML representation for some part of the information content. There is nothing special about <gml:Point> in this example.

Some people might argue that GML conformance itself is a difference that is essential to accuracy and precision. We cannot agree. For a counter-example, consider that the standard output from a GPS device is not GML. Yet no one rejects GPS data because it is not GML. Instead they find a programmer to write a transformation that will convert the GPS information

⁴ That is, any element defined in the GML schema, or in a GML application schema

content into the data representation they desire. Precisely the same thing can be done with a NIEM IEP.

The possibility of data transformation opens the door to other geospatial representations within a NIEM IEP. For example, the IEPD designer might choose to use KML components to represent location. Our example IEP would then be:

```

1 <ge:Goal
2   xmlns:kml="http://www.opengis.net/kml/2.2"
3   xmlns:ge="http://metadata.ces.mil/mdr/ns/example/Goal/extension/1.0"
4   <ge:GoalDescription>Dr. Scott's Desk</ge:TargetDescription>
5   <ge:GoalControlPointLocation>
6     <kml:longitude>38.92133</kml:longitude>
7     <kml:latitude>-77.20438</kml:latitude>
8   </ge:GoalControlPointLocation>
9 </ge:Goal>
```

The schema documentation for `ge:GoalControlPointLocation` would state that the coordinates in the child elements are to be interpreted against the WGS84_2D coordinate reference system specified by "http://metadata.ces.mil/mdr/ns/GSIP/crs/WGS84E_2D". It is then possible to transform the KML elements into the equivalent `gml:Point` element without loss of accuracy or precision. The schema documentation might even refer to an XSLT implementation of that transform, stating something like "the meaning of this element is the meaning of the GML you get by applying this XSLT transform to the contents."

Conclusion

We have shown that a NIEM IEP can represent geolocations with the same accuracy and precision as a GML document. The IEP may contain embedded GML, in which case the geolocation is represented with the exact same data found in the GML document. Nothing about the NIEM "adapter" changes the meaning of the embedded GML, or the way it is validated.

We also showed that a NIEM IEP can represent geolocations in formats which can be transformed into GML. There need be no difference in accuracy and precision before and after this transform.

We therefore conclude that a NIEM data exchange can represent geolocations with the accuracy and precision demanded by "precision applications" such as targeting and safety of navigation. Using NIEM does not introduce any real problems, at least so far as the accuracy and precision of the information content are concerned.

Appendix – GML application schema

```

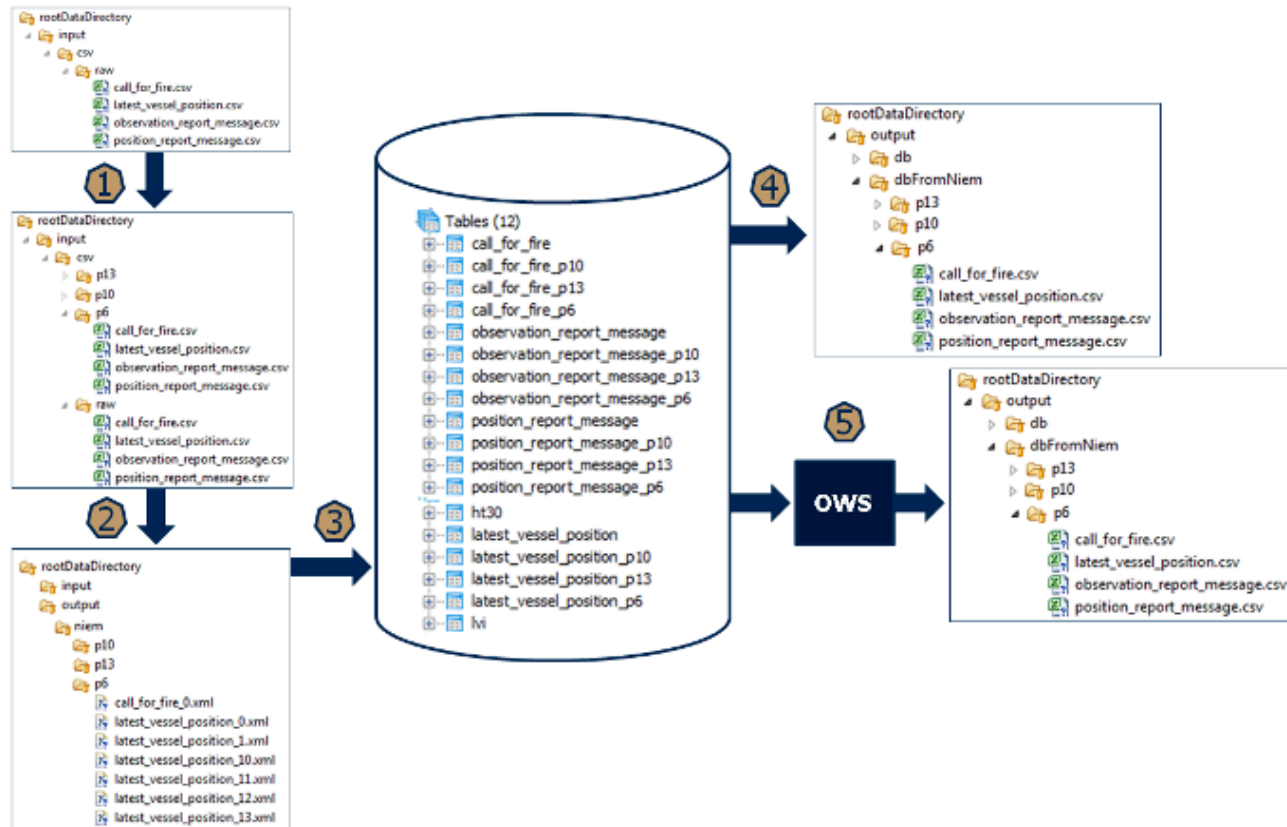
1 <xs:schema
2   targetNamespace="http://metadata.ces.mil/mdr/ns/example/GoalAS"
3   xmlns:gas="http://metadata.ces.mil/mdr/ns/example/GoalAS"
4   xmlns:gml="http://www.opengis.net/gml/3.2"
5   xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
6   <xs:import namespace="http://www.opengis.net/gml/3.2" schemaLocation="gml.xsd" />
7   <xs:element name="Goal" type="gas:GoalType" substitutionGroup="gml:AbstractFeature"/>
8   <xs:complexType name="GoalType">
9     <xs:complexContent>
10      <xs:extension base="gml:AbstractFeatureType">
11        <xs:sequence>
12          <xs:element ref="gas:controlPointLocation"/>
13        </xs:sequence>
14      </xs:extension>
15    </xs:complexContent>
16  </xs:complexType>
17  <xs:element name="controlPointLocation" type="gas:ControlPointLocationType"/>
18  <xs:complexType name="ControlPointLocationType">
19    <xs:sequence>
20      <xs:element ref="gml:Point"/>
21    </xs:sequence>
22  </xs:complexType>
23 </xs:schema>

```

APPENDIX B: Experiment Data Flow Description

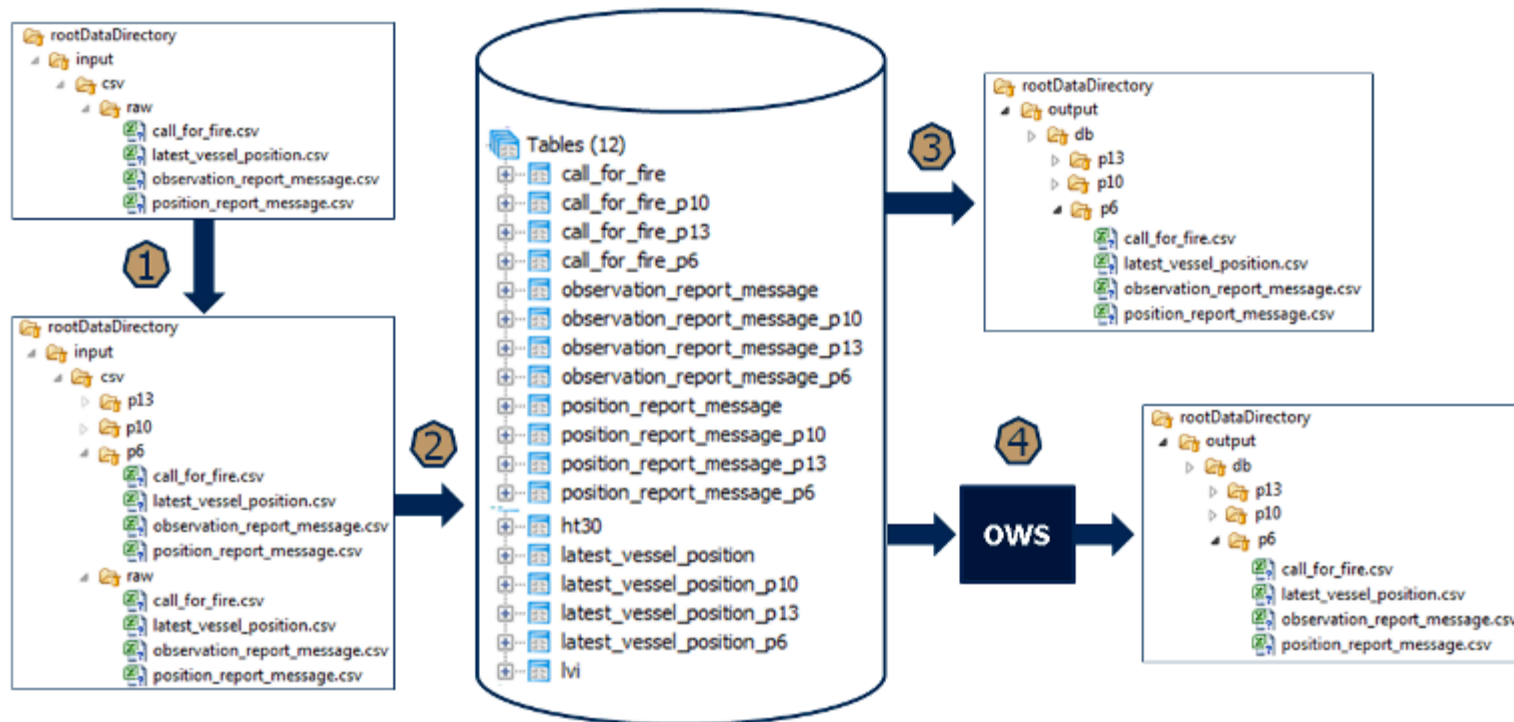
Experiment Run #1 Data Flow Description:

1. The raw CSV data files are copied into P6, P10, P13 folders and random numbers added to achieve proper precision while maintaining CSV format.
2. Java code is used to parse the CSV files and generate XML documents.
3. The database tables were created; XML files were parsed and data inserted into the database by Java code using SQL.
4. Java code using SQL exports data from database and saved in CSV formatted output files.
5. Java code calls to OWS and exports data from the database saving it in CSV formatted output files.



Experiment Run #2 Data Flow Description:

1. Raw CSV data files were copied into P6, P10, P13 folders and random numbers added to achieve proper precision while maintaining CSV format.
2. Database tables were created; CSV files were parsed and data inserted into the database by Java code using SQL.
3. Java code using SQL exports data from database and saved in CSV formatted output files.
4. Java code calls to OWS and exports data from the database saving it in CSV formatted output files



APPENDIX C: OGC GML Validation Tool

An evaluation of the OGC GML validation tool was conducted during MOGIE. This section provides details on the evaluation and details for three recommended changes:

1. Provide Authentication Support
2. Provide Better Internal Error Reporting
3. Provide additional Source Code Comments in TEAM Engine

Overview

A validator is a tool used to verify that an implementation of a certain standard conforms to the rules of that standard. These tools are invaluable in the testing and debugging process during development because they can point out potential issues, specific to a standard, that a developer may not consider during standard testing process. Validators are also often used at run-time to ensure a given input conforms to the claimed standard before the input is processed. For example, a Geographic Markup Language (GML) validator might be used by a GML client application to validate an external GML document before attempting to display its contents. This makes it important for developers to ensure their implementations can be successfully validated during testing; otherwise external applications may refuse to interoperate. This report will focus on the validation capabilities provided by OGC Compliance Testing Program (CITE), specifically, GML.²²

CITE: CITE is an OGC program focused on providing test tools for OGC standards. “The OGC Compliance Testing Program is an ongoing initiative that develops tests for OGC standards, and makes those tests available for online testing. This web site is a resources [sic] for developers that are working with their software to make it OGC compliant, as well as for developers improving the testing scripts and software. It provides a summary of available tests, information about building the source code, reference implementation, etc.”²³

TEAM Engine: CITE makes good use of an open source testing engine called TEAM Engine. “TEAM Engine (Test, Evaluation, And Measurement Engine) is an engine for testing web services and other resources written in JAVA. It executes test scripts written in Compliance Test Language (CTL), TestNG and other languages. It is lightweight and easy to run as a command line or to setup as a service. It can be used to test any type of service or encoding. It is also the official tool used by the OGC for compliance testing.”²⁴ With TEAM Engine, one can develop test suites in CTL without worrying about all of the mechanics necessary to execute the tests.

ETS-GML: As an alternative, one can develop a standalone test suite capable of running without a testing dependency such as TEAM Engine. ETS-GML is one such application. “This executable test suite (ETS) verifies the conformance of GML data and application schemas with respect to ISO 19136:2007 (GML 3.2.1). As shown in Figure 1, a conforming GML data instance must refer to the relevant GML application schema, which in turn imports the complete GML schema.”²⁵

The remainder of this report describes our use of both TEAM Engine and ETS-GML in the MOGIE initiative.

²² This document was written in early July, 2013. CITE applications are being actively developed which may result in some of the links and quotes being out of date.

²³ <http://cite.opengeospatial.org/>

²⁴ <http://sourceforge.net/projects/teamengine/>

²⁵ <http://cite.opengeospatial.org/te2/about/gml/3.2.1-r3/web/overview.html>

Accessing the Validator

CITE does an excellent job of providing various options for accessing and executing the validator tools. The following section contains a summary of each option.

Public Services

TEAM Engine is capable of being hosted as a web service. CITE leverages this by hosting stable and beta versions of their CTL suites on the OGC website. The official (stable) site is here: <http://cite.openeospatial.org/teamengine/> and the beta version is here: <http://cite.openeospatial.org/te2/>. Between the two, one can test most of the published OGC standards. Some of the later versions are only available on the beta site. While this option is extremely convenient, in many cases, network authentication and security often prohibit its use when working with secure services or those behind firewalls during development.

Pre-built Packages

CITE also makes the software available to the public as downloadable packages to allow testers to validate on the machine of their choosing. Two approaches are available, command-line access and self-hosting. As mentioned, TEAM Engine can be hosted as a web service so developers are free to host the service on their own network. Instructions for this can be found in the files section of the SourceForge page. This is a link to the latest version as of this analysis: <http://iweb.dl.sourceforge.net/project/teamengine/Team%20Engine/4.0beta1/teamengine-4.0-beta1.pdf>. While very useful for making internal testing available to a private network, this approach is overkill for some other scenarios. A simpler process to enable internal testing is to pass the correct parameters to TEAM Engine at the command-line. Instructions for this can be found here: <http://cite.openeospatial.org/easytesting>. While this option does overcome the limitations of the public services, sometimes more control over the application is necessary for debugging or customizations. Also, the very latest features and bug-fixes aren't always available in the pre-built packages. When this is the case, building the applications from source may be the best option.

Building from Source

As TEAM Engine, and the ETS code, is open source, one can build the applications from source available in public repositories. TEAM Engine is hosted on source forge: <http://sourceforge.net/projects/teamengine>. However, the OGC CTL scripts, as well as the ETS source, must be retrieved from the OGC repositories. The scripts, requiring TEAM Engine, can be checked out here: <https://svn.openeospatial.org/ogc-projects/cite/scripts>, and ETS source here: <https://svn.openeospatial.org/ogc-projects/cite/ets/index.html>

More information on all of these approaches can be found on the CITE homepage: <http://cite.openeospatial.org>. For the purposes of MOGIE, TEAM Engine and ETS-GML was built from source to validate the GML schema.

MOGIE GML Validation

To ensure the MOGIE experiment was working with a valid GML application schema, TEAM Engine and ETS-GML were employed. The schema validated is shown in Appendix I.

Initial Failed Attempt

Because TEAM Engine does not currently support validating services that require authentication, a couple of work-arounds were attempted to use the public services described above. The first attempt was to provide the URL of the schema directly along with its authority part.²⁶ The second attempt was to download the schema file to the local hard drive and provide the local URI to the downloaded xml file. Both of these attempts produced identical results, the results page reported an error that appeared to be an unhandled internal Java exception:²⁷

```
Error in call to extension function {public java.lang.Object
com.occamlab.te.TECore.callFunction(net.sf.saxon.expr.XPathContext,
java.lang.String,java.lang.String,net.sf.saxon.om.NodeInfo) throws java.lang.Exception}:
Exception in extension function java.lang.Exception: Error invoking function
{http://www.opengis.net/cite/iso19136}run-ets-gml java.lang.RuntimeException: Failed to parse XML
Schema resource located at file:///C:/Users/mweber7/mogie_maritime_schema.xsd Result: Failed
```

After other approaches were attempted with similar results, the decision was made to build from source, as described earlier in this appendix.

Final Successful Attempt

To ensure consistency, two approaches were used to validate the GML schema, ETS-GML was executed directly from within Eclipse and the ETS-GML CTL script was passed to TEAM Engine for execution at the command line. Both the ETS-GML and TEAM Engine source was checked out from trunk on 6/26/2013. This process proved no more difficult than building any other open source Java application.

After building ETS-GML, the main class (org.opengis.cite.iso19136.TestNGController) was executed with a properties file as a parameter. The properties file contained XML specifying the location of the schema file. This approach successfully produced a detailed log which suggested the validation had executed successfully.

After building TEAM Engine, it was executed from the command-line with the GML CTL script location as a parameter. This produces a dialog similar to the one found on the beta testing site where the local schema file URI was provided. When executed, a detailed report was generated which suggested the validation had executed successfully.²⁸

The content of both results were consistent and are explained next.

Results

As mentioned, the results of the two successful approaches were consistent. While, two minor problems with the schema were identified, the results also confirmed that the schema was indeed a valid GML application schema.

²⁶ http://en.wikipedia.org/wiki/URI_scheme#Generic_syntax

²⁷ the displayed exception message is for the local file attempt, the authority part attempt had an identical message except the last line displayed the remote (MOGIE server) URI

²⁸ It should be noted that the developer did receive the same error after building from source. While following the errors through the source code, many things were tried until a successful result was produced. It was the developers intention to track down the specific fix that led to the successful validator execution, however time constraints prevented this.

The initial validation failed for two reasons. The targetNamespace was not abstract and the feature type name was not in UpperCamelCase. It is important to note that neither of these are critical failures and have no impact on the producing or consuming applications. The schemas were modified until zero failures were reported. Having the TEAM Engine graphic validation report made identifying and fixing these errors very simple.

Conclusion/Recommendations

Overall, the experience using TEAM Engine and ETS-GML was very positive. Minor technical recommendations are provided, but in the end, the problems were more than overcome by the accessibility and flexibility of the software. Once everything was running, the generated validation report made finding and fixing the errors quick and easy. The developers made great use of the TEAM Engine validation results user interface which made navigating and analyzing the results effortless. The GIS lab at GTRI was used to host the mentioned packages and look forward to their continued use down the road. In conclusion, the CITE validation tools are an invaluable asset to any OGC standards implementation effort.

While the CITE tools are a great resource there are some missing components that would have saved time and effort, support for authenticated services, better internal error reporting and more comments in the source code.

Recommendation 1: Provide Authentication Support

As more applications use the CITE tools for validation, the demand for authentication support will certainly increase. While the tools are available to work around this, it takes time and effort to implement them. In the case of MOGIE, only GML had to be validated so the work-around was particularly easy. However, the secondary goal of validating the WFS had to be abandoned entirely because of the lack of this feature. In addition, this feature would allow others to easily verify one's reported results when dealing with secure services.

Recommendation 2: Provide Better Internal Error Reporting

The initial error received appeared to be an internal Java exception; however, it provided no clues as to what the problem might be or what the user could do to fix it. It also reported a result of failed, but did not indicate that the validator failed, rather than the validation itself, because an internal error prevented it from being executed. A better error report might indicate that the validation results were inconclusive because of an internal error. Also, if no hints can be provided to the user as to possible problems, the error message should indicate this and provide links to debugging resources for TEAM Engine and the respective CTL script being executed.

Recommendation 3: Provide Additional Source Code Comments in TEAM Engine

The initial error received did supply a Java developer with enough information to possibly track down a fix by simply browsing the source code from a web browser, rather than downloading and building it. While generally tedious compared to browsing code in an IDE such as Eclipse, sometimes clues can be quickly found by reading the developer's comments in key parts of the source. While not entirely absent, informative comments were scarce. For example, com.occamlab.te.Test is a key Java class for TEAM Engine when running from the command-line. It initializes numerous variables that effect the entire application, but very few of these are explained, at least within the source. If explanations are available elsewhere, outside of the source code, the comments should indicate this. Well commented source code also has many other well-known benefits in large applications.

Appendix D: DoD NIEM Adoption Memo



CHIEF INFORMATION OFFICER

DEPARTMENT OF DEFENSE
6000 DEFENSE PENTAGON
WASHINGTON, D.C. 20301-6000

MAR 28 2013

MEMORANDUM FOR SECRETARIES OF THE MILITARY DEPARTMENTS
CHAIRMAN OF THE JOINT CHIEFS OF STAFF
UNDER SECRETARIES OF DEFENSE
DEPUTY CHIEF MANAGEMENT OFFICER
COMMANDERS OF THE COMBATANT COMMANDS
DIRECTOR, COST ASSESSMENT AND PROGRAM EVALUATION
DIRECTOR, OPERATIONAL TEST AND EVALUATION
GENERAL COUNSEL OF THE DEPARTMENT OF DEFENSE
INSPECTOR GENERAL OF THE DEPARTMENT OF DEFENSE
ASSISTANT SECRETARIES OF DEFENSE
ASSISTANTS TO THE SECRETARY OF DEFENSE
DIRECTOR, ADMINISTRATION AND MANAGEMENT
DIRECTOR, NET ASSESSMENT
DIRECTORS OF THE DEFENSE AGENCIES
DIRECTORS OF THE DOD FIELD ACTIVITIES

Subject: Adoption of the National Information Exchange Model within the Department of Defense

In order to comply with White House guidance on the adoption of reference information exchanges, DoD will adopt the National Information Exchange Model (NIEM) as the best suited option for standards-based data exchanges. This adoption will involve a series of phased implementations by Components/Programs using NIEM content, guidance, and tools in an integrated effort to transition current DoD data exchange standards, specifications, and policies to a NIEM-based approach. In addition, the DoD will work with the NIEM Program Management Office to create a Military Operations (MilOps) Domain as part of NIEM.

Given the Clinger-Cohen Act mandates and today's fiscal pressures, DoD must adopt a DoD-wide sustainable business model for information sharing that supports the DoD data strategy, the Joint Information Environment, the DoD Information Enterprise Architecture, and emerging government data sharing guidance. Adoption of NIEM offers potential efficiencies, long-term development cost savings, streamlined governance, and most importantly, improved information sharing across the DoD and with our mission partners.


To facilitate the transition to NIEM, the Office of the DoD Chief Information Officer will lead the development of a DoD Data Framework to include targeted guidance on governance and technical direction regarding NIEM adoption. Specifically, the DoD Data Framework will build upon the existing DoD data strategy and will provide principles, rules and additional guidance for managing data artifacts to improve information sharing. This framework will provide a foundation for how DoD views, manages, and shares its data. As part of the framework effort and to assist in shaping the DoD involvement in NIEM, Combatant Commands, Services and Agencies are encouraged to identify and conduct pilots and demonstrations.

DoD organizations shall first consider NIEM for their information sharing solutions when deciding which data exchange standards or specifications meet their mission and operational needs. Every effort shall be made to ensure that a rigorous analysis of NIEM be conducted as a part of this consideration.

The adoption of NIEM as a common standards-based data exchange may present challenges to some DoD stakeholders, particularly when viewed from the perspective of an individual program. In those situations where an organization's design goals and requirements are not fulfilled by NIEM, the organization shall provide justification why the use of NIEM is not feasible; exceptions may be granted by the DoD CIO. In such cases, an organization shall use an alternative or complementary data standard that is interoperable with NIEM.

This DoD transition to NIEM will incorporate the ongoing efforts of DoD Universal Core (UCore) and Command and Control (C2) Core. As of the date of this memo, the DoD CIO will no longer support the development of further enhancements to UCore and C2 Core as unique DoD data exchange models. The applicable data components in UCore and C2 Core will form the initial content in the NIEM Core and the NIEM MilOps Domain, respectively.

When published, the DoD implementing instruction, DoDI 8320.xx "Implementing the Sharing of Data, Information, and Information Technology (IT) Services in the Department of Defense" will incorporate policy and guidance on DoD adoption of NIEM. The point of contact for this matter is Mr. Bill Barlow at email: William.Barlow@osd.mil, 571-372-4646.



Teresa M. Takai

Appendix E: Base Maritime CSV Data

DoD vessel position report base data. . The raw data was used to generate P6, P10 and P13 data sets.

KEY		RAW DATA		P6		P10		P13	
fid	vessel_name	time	position	time	position	time	latest_position	time	position
80	USSJOHNPAULJONES	13T18:00:00	POINT(-123.558357466727 37.1498819680237)	13T18:00:00	POINT(-123.558357 37.149881)	13T18:00:00	POINT(-123.5583574667 37.149881968)	13T18:00:00	POINT(-123.5583574667275 37.1498819680237)
79	ACTIVE	13T18:00:00	POINT(-123.115604323143 37.9968472588804)	13T18:00:00	POINT(-123.115604 37.996847)	13T18:00:00	POINT(-123.1156043231 37.9968472588)	13T18:00:00	POINT(-123.1156043231439 37.9968472588804)
81	WAESCHE	13T18:00:00	POINT(-122.418566534546 37.3592019818444)	13T18:00:00	POINT(-122.418566 37.359201)	13T18:00:00	POINT(-122.4185665345 37.3592019818)	13T18:00:00	POINT(-122.4185665345461 37.3592019818444)
82	SOCKEYE	13T18:00:00	POINT(-122.431755557827 38.0818013578952)	13T18:00:00	POINT(-122.431755 38.081801)	13T18:00:00	POINT(-122.4317555578 38.0818013578)	13T18:00:00	POINT(-122.4317555578271 38.0818013578952)
73	USSJOHNPAULJONES	14T02:48:00	POINT(-123.144836485947 37.450691596815)	14T02:48:00	POINT(-123.144836 37.450691)	14T02:48:00	POINT(-123.1448364859 37.4506915968)	14T02:48:00	POINT(-123.1448364859476 37.4506915968159)
77	USSJOHNPAULJONES	14T03:18:00	POINT(-123.063912995402 37.5094173234359)	14T03:18:00	POINT(-123.063912 37.509417)	14T03:18:00	POINT(-123.0639129954 37.5094173234)	14T03:18:00	POINT(-123.0639129954021 37.5094173234359)
74	USSJOHNPAULJONES	14T03:48:00	POINT(-123.016447687762 37.5438411540581)	14T03:48:00	POINT(-123.016447 37.543841)	14T03:48:00	POINT(-123.0164476877 37.543841154)	14T03:48:00	POINT(-123.0164476877627 37.5438411540581)
72	USSJOHNPAULJONES	14T01:48:00	POINT(-123.338676474924 37.3098354132718)	14T01:48:00	POINT(-123.338676 37.309835)	14T01:48:00	POINT(-123.3386764749 37.3098354132)	14T01:48:00	POINT(-123.3386764749242 37.3098354132718)
76	USSJOHNPAULJONES	14T02:18:00	POINT(-123.241479724312 37.3804976834214)	14T02:18:00	POINT(-123.241479 37.380497)	14T02:18:00	POINT(-123.2414797243 37.3804976834)	14T02:18:00	POINT(-123.2414797243125 37.3804976834214)
83	TERN	13T18:00:00	POINT(-122.189322235658 37.603180285202)	13T18:00:00	POINT(-122.189322 37.60318)	13T18:00:00	POINT(-122.1893222356 37.6031802852)	13T18:00:00	POINT(-122.1893222356584 37.6031802852026)
66	WAESCHE	14T02:48:00	POINT(-122.534434350078 37.7337506047079)	14T02:48:00	POINT(-122.534434 37.73375)	14T02:48:00	POINT(-122.53443435 37.7337506047)	14T02:48:00	POINT(-122.5344343500786 37.7337506047079)
62	ACTIVE	14T03:18:00	POINT(-122.527899209249 37.8093582646386)	14T03:18:00	POINT(-122.527899 37.809358)	14T03:18:00	POINT(-122.5278992092 37.8093582646)	14T03:18:00	POINT(-122.5278992092491 37.8093582646386)
67	WAESCHE	14T03:18:00	POINT(-122.530276006525 37.7830643806786)	14T03:18:00	POINT(-122.530276 37.783064)	14T03:18:00	POINT(-122.5302760065 37.7830643806)	14T03:18:00	POINT(-122.5302760065255 37.7830643806786)
59	ACTIVE	14T01:48:00	POINT(-122.849755648709 37.9120956950377)	14T01:48:00	POINT(-122.849755 37.912095)	14T01:48:00	POINT(-122.8497556487 37.912095695)	14T01:48:00	POINT(-122.8497556487091 37.9120956950377)
60	ACTIVE	14T02:18:00	POINT(-122.673649155211 37.8558997760467)	14T02:18:00	POINT(-122.673649 37.855899)	14T02:18:00	POINT(-122.6736491552 37.855899776)	14T02:18:00	POINT(-122.6736491552111 37.8558997760467)
65	WAESCHE	14T02:18:00	POINT(-122.539873640128 37.6691964810591)	14T02:18:00	POINT(-122.539873 37.669196)	14T02:18:00	POINT(-122.5398736401 37.669196481)	14T02:18:00	POINT(-122.5398736401287 37.6691964810591)
70	TERN	14T02:18:00	POINT(-122.301594216327 37.7729762720888)	14T02:18:00	POINT(-122.301594 37.772976)	14T02:18:00	POINT(-122.3015942163 37.772976272)	14T02:18:00	POINT(-122.3015942163278 37.7729762720888)
61	ACTIVE	14T02:48:00	POINT(-122.585256731182 37.8276774233891)	14T02:48:00	POINT(-122.585256 37.827677)	14T02:48:00	POINT(-122.5852567311 37.8276774233)	14T02:48:00	POINT(-122.5852567311824 37.8276774233891)
54	USSJOHNPAULJONES	14T00:48:00	POINT(-123.558357466727 37.1498819680237)	14T00:48:00	POINT(-123.558357 37.149881)	14T00:48:00	POINT(-123.5583574667 37.149881968)	14T00:48:00	POINT(-123.5583574667272 37.1498819680237)
53	ACTIVE	14T00:48:00	POINT(-123.115604323143 37.9968472588804)	14T00:48:00	POINT(-123.115604 37.996847)	14T00:48:00	POINT(-123.1156043231 37.9968472588)	14T00:48:00	POINT(-123.1156043231434 37.9968472588804)
55	WAESCHE	14T00:48:00	POINT(-122.418566534546 37.3592019818444)	14T00:48:00	POINT(-122.418566 37.359201)	14T00:48:00	POINT(-122.4185665345 37.3592019818)	14T00:48:00	POINT(-122.4185665345467 37.3592019818444)
56	SOCKEYE	14T00:48:00	POINT(-122.431755557827 38.0818013578952)	14T00:48:00	POINT(-122.431755 38.081801)	14T00:48:00	POINT(-122.4317555578 38.0818013578)	14T00:48:00	POINT(-122.4317555578274 38.0818013578952)
57	TERN	14T00:48:00	POINT(-122.189322235658 37.603180285202)	14T00:48:00	POINT(-122.189322 37.60318)	14T00:48:00	POINT(-122.1893222356 37.6031802852)	14T00:48:00	POINT(-122.1893222356583 37.6031802852028)
58	ACTIVE	14T01:18:00	POINT(-122.990028965057 37.9568264898608)	14T01:18:00	POINT(-122.990028 37.956826)	14T01:18:00	POINT(-122.990028965 37.9568264898)	14T01:18:00	POINT(-122.9900289650576 37.9568264898608)
75	USSJOHNPAULJONES	14T01:18:00	POINT(-123.454811859965 37.2253175495898)	14T01:18:00	POINT(-123.454811 37.225317)	14T01:18:00	POINT(-123.4548118599 37.2253175495)	14T01:18:00	POINT(-123.4548118599657 37.2253175495898)
63	WAESCHE	14T01:18:00	POINT(-122.492598735186 37.453713520708)	14T01:18:00	POINT(-122.492598 37.453713)	14T01:18:00	POINT(-122.4925987351 37.4537135207)	14T01:18:00	POINT(-122.4925987351862 37.4537135207083)
68	TERN	14T01:18:00	POINT(-122.223993376212 37.6556571186078)	14T01:18:00	POINT(-122.223993 37.655657)	14T01:18:00	POINT(-122.2239933762 37.6556571186)	14T01:18:00	POINT(-122.2239933762126 37.6556571186078)
78	SOCKEYE	14T01:18:00	POINT(-122.320831465055 38.0625569091185)	14T01:18:00	POINT(-122.320831 38.062556)	14T01:18:00	POINT(-122.320831465 38.0625569091)	14T01:18:00	POINT(-122.3208314650555 38.0625569091185)
64	WAESCHE	14T01:48:00	POINT(-122.55166718201 37.5290363765762)	14T01:48:00	POINT(-122.551667 37.529036)	14T01:48:00	POINT(-122.551667182 37.5290363765)	14T01:48:00	POINT(-122.5516671820142 37.5290363765762)
69	TERN	14T01:48:00	POINT(-122.264954616372 37.7176066329608)	14T01:48:00	POINT(-122.264954 37.717606)	14T01:48:00	POINT(-122.2649546163 37.7176066329)	14T01:48:00	POINT(-122.2649546163722 37.7176066329608)
71	SOCKEYE	14T01:48:00	POINT(-122.272454977521 38.05416238737)	14T01:48:00	POINT(-122.272454 38.054162)	14T01:48:00	POINT(-122.2724549775 38.0541623873)	14T01:48:00	POINT(-122.2724549775216 38.0541623873787)

Appendix F: Land Scenario Base Data

Observation report and call for fire base data. The raw data was used to generate P6, P10 and P13 data sets.

KEY		RAW DATA			P6 DATA		
fid	FID	target_position	observer_position	message_time	target_position	observer_position	message_time
4	observation_report_message_v2.4	POINT(70.461245 34.426487)	POINT(70.457811 34.42224)	2001-12-17T09:30:28	POINT(70.461245 34.426487)	POINT(70.457811 34.422245)	2001-12-17T09:30:28
5	observation_report_message_v2.5	POINT(70.461245 34.426487)	POINT(70.457811 34.42224)	2001-12-17T09:31:00	POINT(70.461245 34.426487)	POINT(70.457811 34.422248)	2001-12-17T09:31:00

KEY		P10 DATA			P13 DATA		
fid	FID	target_position	observer_position	message_time	target_position	observer_position	message_time
4	observation_report_message_v2.4	POINT(70.4612454886 34.4264877127)	POINT(70.4578115699 34.4222489744)	2001-12-17T09:30:28	POINT(70.4612459399691 34.4264876164852)	POINT(70.4578116545343 34.4222422652171)	2001-12-17T09:30:28
5	observation_report_message_v2.5	POINT(70.4612451481 34.4264874521)	POINT(70.4578115296 34.4222438558)	2001-12-17T09:31:00	POINT(70.4612453834585 34.4264874738852)	POINT(70.4578111925758 34.4222443465932)	2001-12-17T09:31:00

KEY		RAW DATA			P6 DATA		
FID	fid	fire_mission_observer_position	target_position	message_time_text	fire_mission_observer_position	target_position	message_time_text
1	call_for_fire_v2.1	POINT(70.457811 34.42224)	POINT(70.461245 34.426487)	12/17/2001 9:32	POINT(70.457811 34.422246)	POINT(70.461245 34.426487)	12/17/2001 9:32

KEY		P10 DATA			P13 DATA		
FID	fid	fire_mission_observer_position	target_position	message_time_text	fire_mission_observer_position	target_position	message_time_text
1	call_for_fire_v2.1	POINT(70.4578115655 34.4222467357)	POINT(70.4612458186 34.4264875635)	12/17/2001 9:32	POINT(70.4578114799166 34.4222472232535)	POINT(70.4612455299533 34.4264872848745)	12/17/2001 9:32

MOGIE position report base data. The raw data was used to generate P6, P10 and P13 data sets.

KEY		RAW DATA		P6 DATA		P10 DATA		P13 DATA	
fid	FID	message time	unit location	message time	unit location	message time	unit location	message time	unit location
149	position_report_message_v2.149	2001-12-17T09:31:16	POINT (70.467218 34.426735)	2001-12-17T09:31:16	POINT (70.467218 34.426735)	2001-12-17T09:31:16	POINT (70.4672182962 34.4267357244)	2001-12-17T09:31:16	POINT (70.4672124742145 34.4267351262567)
150	position_report_message_v2.150	2001-12-17T09:31:18	POINT (70.455279 34.428895)	2001-12-17T09:31:18	POINT (70.455279 34.428895)	2001-12-17T09:31:18	POINT (70.4552791212 34.4288958917)	2001-12-17T09:31:18	POINT (70.4552799521325 34.4288958585689)
151	position_report_message_v2.151	2001-12-17T09:31:20	POINT (70.467124 34.429426)	2001-12-17T09:31:20	POINT (70.467124 34.429426)	2001-12-17T09:31:20	POINT (70.4671244914 34.4294269813)	2001-12-17T09:31:20	POINT (70.4671246994579 34.4294262847241)
123	position_report_message_v2.123	2001-12-17T09:30:24	POINT (70.457811 34.42224)	2001-12-17T09:30:24	POINT (70.457811 34.42224)	2001-12-17T09:30:24	POINT (70.4578114839 34.4222442667)	2001-12-17T09:30:24	POINT (70.4578114818457 34.4222439147562)
124	position_report_message_v2.124	2001-12-17T09:30:26	POINT (70.464127 34.422204)	2001-12-17T09:30:26	POINT (70.464127 34.422204)	2001-12-17T09:30:26	POINT (70.4641251717 34.4222041939)	2001-12-17T09:30:26	POINT (70.4641217154839 34.4222041284883)
155	position_report_message_v2.155	2001-12-17T09:30:12	POINT (70.464163 34.415973)	2001-12-17T09:30:12	POINT (70.464163 34.415973)	2001-12-17T09:30:12	POINT (70.4641637436 34.4159734628)	2001-12-17T09:30:12	POINT (70.4641634243627 34.4159734787385)
156	position_report_message_v2.156	2001-12-17T09:30:14	POINT (70.457768 34.420505)	2001-12-17T09:30:14	POINT (70.457768 34.420505)	2001-12-17T09:30:14	POINT (70.4577684354 34.4205058469)	2001-12-17T09:30:14	POINT (70.4577685928137 34.4205052714489)
157	position_report_message_v2.157	2001-12-17T09:30:16	POINT (70.464124 34.420823)	2001-12-17T09:30:16	POINT (70.464124 34.420823)	2001-12-17T09:30:16	POINT (70.4641296144 34.4208233298)	2001-12-17T09:30:16	POINT (70.4641215967795 34.4208235131875)
125	position_report_message_v2.125	2001-12-17T09:30:30	POINT (70.461116 34.421673)	2001-12-17T09:30:30	POINT (70.461116 34.421673)	2001-12-17T09:30:30	POINT (70.4611167175 34.4216732684)	2001-12-17T09:30:30	POINT (70.4611169141321 34.4216737848372)
126	position_report_message_v2.126	2001-12-17T09:30:32	POINT (70.459528 34.419974)	2001-12-17T09:30:32	POINT (70.459528 34.419974)	2001-12-17T09:30:32	POINT (70.4595289723 34.4199744337)	2001-12-17T09:30:32	POINT (70.4595285346292 34.4199746977399)
127	position_report_message_v2.127	2001-12-17T09:30:34	POINT (70.46279 34.420009)	2001-12-17T09:30:34	POINT (70.46279 34.420009)	2001-12-17T09:30:34	POINT (70.4627988168 34.4200091381)	2001-12-17T09:30:34	POINT (70.4627958881738 34.4200093471561)
128	position_report_message_v2.128	2001-12-17T09:30:36	POINT (70.457811 34.424541)	2001-12-17T09:30:36	POINT (70.457811 34.424541)	2001-12-17T09:30:36	POINT (70.4578112233 34.4245412732)	2001-12-17T09:30:36	POINT (70.4578118728336 34.4245416496466)
129	position_report_message_v2.129	2001-12-17T09:30:38	POINT (70.464077 34.424895)	2001-12-17T09:30:38	POINT (70.464077 34.424895)	2001-12-17T09:30:38	POINT (70.4640774546 34.4248957512)	2001-12-17T09:30:38	POINT (70.4640777423417 34.4248957473228)
131	position_report_message_v2.131	2001-12-17T09:30:40	POINT (70.46103 34.424541)	2001-12-17T09:30:40	POINT (70.461033 34.424541)	2001-12-17T09:30:40	POINT (70.4610376184 34.4245418758)	2001-12-17T09:30:40	POINT (70.4610394128748 34.4245412489533)
153	position_report_message_v2.153	2001-12-17T09:30:08	POINT (70.461159 34.417283)	2001-12-17T09:30:08	POINT (70.461159 34.417283)	2001-12-17T09:30:08	POINT (70.4611598189 34.4172831796)	2001-12-17T09:30:08	POINT (70.4611599972473 34.4172837385182)
141	position_report_message_v2.141	2001-12-17T09:30:04	POINT (70.400906 34.419372)	2001-12-17T09:30:04	POINT (70.400906 34.419372)	2001-12-17T09:30:04	POINT (70.4009065911 34.4193726779)	2001-12-17T09:30:04	POINT (70.4009067492128 34.4193721367816)
130	position_report_message_v2.130	2001-12-17T09:30:02	POINT (70.381851 34.436646)	2001-12-17T09:30:02	POINT (70.381851 34.436646)	2001-12-17T09:30:02	POINT (70.3818511943 34.4366464934)	2001-12-17T09:30:02	POINT (70.3818518536822 34.4366464698214)
119	position_report_message_v2.119	2001-12-17T09:30:00	POINT (70.436783 34.414982)	2001-12-17T09:30:00	POINT (70.436783 34.414982)	2001-12-17T09:30:00	POINT (70.4367838529 34.4149827749)	2001-12-17T09:30:00	POINT (70.4367832856884 34.4149822361354)
152	position_report_message_v2.152	2001-12-17T09:30:06	POINT (70.461073 34.418098)	2001-12-17T09:30:06	POINT (70.461073 34.418098)	2001-12-17T09:30:06	POINT (70.4610737896 34.4180989288)	2001-12-17T09:30:06	POINT (70.4610738396813 34.418098947438)
132	position_report_message_v2.132	2001-12-17T09:30:42	POINT (70.459313 34.42231)	2001-12-17T09:30:42	POINT (70.459313 34.42231)	2001-12-17T09:30:42	POINT (70.4593138276 34.4223184667)	2001-12-17T09:30:42	POINT (70.4593133117518 34.4223145672987)
133	position_report_message_v2.133	2001-12-17T09:30:44	POINT (70.462618 34.422314)	2001-12-17T09:30:44	POINT (70.462618 34.422314)	2001-12-17T09:30:44	POINT (70.4626182471 34.4223166687)	2001-12-17T09:30:44	POINT (70.4626185296518 34.4223182868568)
134	position_report_message_v2.134	2001-12-17T09:30:46	POINT (70.459185 34.425744)	2001-12-17T09:30:46	POINT (70.459185 34.425744)	2001-12-17T09:30:46	POINT (70.4591859943 34.4257441879)	2001-12-17T09:30:46	POINT (70.4591853721816 34.4257445365357)
135	position_report_message_v2.135	2001-12-17T09:30:48	POINT (70.463862 34.426204)	2001-12-17T09:30:48	POINT (70.463862 34.426204)	2001-12-17T09:30:48	POINT (70.4638624252 34.4262049685)	2001-12-17T09:30:48	POINT (70.4638622628899 34.4262044881117)
136	position_report_message_v2.136	2001-12-17T09:30:50	POINT (70.461202 34.426487)	2001-12-17T09:30:50	POINT (70.461202 34.426487)	2001-12-17T09:30:50	POINT (70.4612029287 34.4264877843)	2001-12-17T09:30:50	POINT (70.4612026366761 34.4264876338345)
137	position_report_message_v2.137	2001-12-17T09:30:52	POINT (70.457726 34.426027)	2001-12-17T09:30:52	POINT (70.457726 34.426027)	2001-12-17T09:30:52	POINT (70.4577261669 34.4260272288)	2001-12-17T09:30:52	POINT (70.4577261499958 34.4260275635246)
138	position_report_message_v2.138	2001-12-17T09:30:54	POINT (70.463862 34.426134)	2001-12-17T09:30:54	POINT (70.463862 34.426134)	2001-12-17T09:30:54	POINT (70.4638622481 34.4261341984)	2001-12-17T09:30:54	POINT (70.4638627866995 34.4261349219761)
139	position_report_message_v2.139	2001-12-17T09:30:56	POINT (70.457897 34.428859)	2001-12-17T09:30:56	POINT (70.457897 34.428859)	2001-12-17T09:30:56	POINT (70.4578975981 34.4288597338)	2001-12-17T09:30:56	POINT (70.4578975869774 34.4288594564269)
140	position_report_message_v2.140	2001-12-17T09:30:58	POINT (70.46648 34.427514)	2001-12-17T09:30:58	POINT (70.466482 34.427514)	2001-12-17T09:30:58	POINT (70.4664862724 34.4275144686)	2001-12-17T09:30:58	POINT (70.4664811457727 34.4275141746689)
142	position_report_message_v2.142	2001-12-17T09:31:02	POINT (70.458541 34.427726)	2001-12-17T09:31:02	POINT (70.458541 34.427726)	2001-12-17T09:31:02	POINT (70.4585412423 34.4277262583)	2001-12-17T09:31:02	POINT (70.4585413525986 34.4277267446448)
143	position_report_message_v2.143	2001-12-17T09:31:04	POINT (70.457768 34.428293)	2001-12-17T09:31:04	POINT (70.457768 34.428293)	2001-12-17T09:31:04	POINT (70.4577682694 34.4282936418)	2001-12-17T09:31:04	POINT (70.4577682198864 34.4282934478899)
144	position_report_message_v2.144	2001-12-17T09:31:06	POINT (70.46721 34.426735)	2001-12-17T09:31:06	POINT (70.467216 34.426735)	2001-12-17T09:31:06	POINT (70.4672191182 34.4267359135)	2001-12-17T09:31:06	POINT (70.4672148118892 34.4267351714177)
145	position_report_message_v2.145	2001-12-17T09:31:08	POINT (70.455279 34.428895)	2001-12-17T09:31:08	POINT (70.455279 34.428895)	2001-12-17T09:31:08	POINT (70.4552792819 34.4288955368)	2001-12-17T09:31:08	POINT (70.4552796657336 34.4288953377983)
146	position_report_message_v2.146	2001-12-17T09:31:10	POINT (70.467124 34.429426)	2001-12-17T09:31:10	POINT (70.467124 34.429426)	2001-12-17T09:31:10	POINT (70.4671242692 34.4294269613)	2001-12-17T09:31:10	POINT (70.4671249535637 34.4294269759855)
147	position_report_message_v2.147	2001-12-17T09:31:12	POINT (70.458541 34.427726)	2001-12-17T09:31:12	POINT (70.458541 34.427726)	2001-12-17T09:31:12	POINT (70.4585419669 34.4277264317)	2001-12-17T09:31:12	POINT (70.4585418469745 34.4277267485664)
148	position_report_message_v2.148	2001-12-17T09:31:14	POINT (70.457768 34.428293)	2001-12-17T09:31:14	POINT (70.457768 34.428293)	2001-12-17T09:31:14	POINT (70.4577681649 34.4282932146)	2001-12-17T09:31:14	POINT (70.4577684584748 34.428293118765)
154	position_report_message_v2.154	2001-12-17T09:30:10	POINT (70.457983 34.416115)	2001-12-17T09:30:10	POINT (70.457983 34.416115)	2001-12-17T09:30:10	POINT (70.4579836491 34.4161154217)	2001-12-17T09:30:10	POINT (70.4579834311258 34.4161152886158)
120	position_report_message_v2.120	2001-12-17T09:30:18	POINT (70.461073 34.419018)	2001-12-17T09:30:18	POINT (70.461073 34.419018)	2001-12-17T09:30:18	POINT (70.4610733437 34.4190183557)	2001-12-17T09:30:18	POINT (70.4610732869797 34.4190186452734)
121	position_report_message_v2.121	2001-12-17T09:30:20	POINT (70.459614 34.417283)	2001-12-17T09:30:20	POINT (70.459614 34.417283)	2001-12-17T09:30:20	POINT (70.4596147898 34.4172835441)	2001-12-17T09:30:20	POINT (70.4596148976431 34.4172836224629)
122	position_report_message_v2.122	2001-12-17T09:30:22	POINT (70.462832 34.417319)	2001-12-17T09:30:22	POINT (70.462832 34.417319)	2001-12-17T09:30:22	POINT (70.4628327142 34.4173192635)	2001-12-17T09:30:22	POINT (70.4628328511142 34.4173198942838)

Appendix G: Data Analysis Results

The following is the results of the data analysis completed for MOGIE described in Section 6.

test	precision	type	lonDiffSum	latDiffSum	baseFile	comparedFile
csvToNiemToDb	6	position_report_message	0	0	rootDataDirectory/input/csv/p6/position_report_message.csv	rootDataDirectory/output/dbFromNiem/p6/position_report_message.csv
csvToNiemToDb	6	observation_report_message	0	0	rootDataDirectory/input/csv/p6/observation_report_message.csv	rootDataDirectory/output/dbFromNiem/p6/observation_report_message.csv
csvToNiemToDb	6	latest_vessel_position	0	0	rootDataDirectory/input/csv/p6/latest_vessel_position.csv	rootDataDirectory/output/dbFromNiem/p6/latest_vessel_position.csv
csvToNiemToDb	6	call_for_fire	0	0	rootDataDirectory/input/csv/p6/call_for_fire.csv	rootDataDirectory/output/dbFromNiem/p6/call_for_fire.csv
csvToNiemToDb	10	position_report_message	0	0	rootDataDirectory/input/csv/p10/position_report_message.csv	rootDataDirectory/output/dbFromNiem/p10/position_report_message.csv
csvToNiemToDb	10	observation_report_message	0	0	rootDataDirectory/input/csv/p10/observation_report_message.csv	rootDataDirectory/output/dbFromNiem/p10/observation_report_message.csv
csvToNiemToDb	10	latest_vessel_position	0	0	rootDataDirectory/input/csv/p10/latest_vessel_position.csv	rootDataDirectory/output/dbFromNiem/p10/latest_vessel_position.csv
csvToNiemToDb	10	call_for_fire	0	0	rootDataDirectory/input/csv/p10/call_for_fire.csv	rootDataDirectory/output/dbFromNiem/p10/call_for_fire.csv
csvToNiemToDb	13	position_report_message	0	0	rootDataDirectory/input/csv/p13/position_report_message.csv	rootDataDirectory/output/dbFromNiem/p13/position_report_message.csv
csvToNiemToDb	13	observation_report_message	0	0	rootDataDirectory/input/csv/p13/observation_report_message.csv	rootDataDirectory/output/dbFromNiem/p13/observation_report_message.csv
csvToNiemToDb	13	latest_vessel_position	0	0	rootDataDirectory/input/csv/p13/latest_vessel_position.csv	rootDataDirectory/output/dbFromNiem/p13/latest_vessel_position.csv
csvToNiemToDb	13	call_for_fire	0	0	rootDataDirectory/input/csv/p13/call_for_fire.csv	rootDataDirectory/output/dbFromNiem/p13/call_for_fire.csv
csvToDb	6	position_report_message	0	0	rootDataDirectory/input/csv/p6/position_report_message.csv	rootDataDirectory/output/db/p6/position_report_message.csv
csvToDb	6	observation_report_message	0	0	rootDataDirectory/input/csv/p6/observation_report_message.csv	rootDataDirectory/output/db/p6/observation_report_message.csv
csvToDb	6	latest_vessel_position	0	0	rootDataDirectory/input/csv/p6/latest_vessel_position.csv	rootDataDirectory/output/db/p6/latest_vessel_position.csv
csvToDb	6	call_for_fire	0	0	rootDataDirectory/input/csv/p6/call_for_fire.csv	rootDataDirectory/output/db/p6/call_for_fire.csv
csvToDb	10	position_report_message	0	0	rootDataDirectory/input/csv/p10/position_report_message.csv	rootDataDirectory/output/db/p10/position_report_message.csv
csvToDb	10	observation_report_message	0	0	rootDataDirectory/input/csv/p10/observation_report_message.csv	rootDataDirectory/output/db/p10/observation_report_message.csv
csvToDb	10	latest_vessel_position	0	0	rootDataDirectory/input/csv/p10/latest_vessel_position.csv	rootDataDirectory/output/db/p10/latest_vessel_position.csv
csvToDb	10	call_for_fire	0	0	rootDataDirectory/input/csv/p10/call_for_fire.csv	rootDataDirectory/output/db/p10/call_for_fire.csv
csvToDb	13	position_report_message	0	0	rootDataDirectory/input/csv/p13/position_report_message.csv	rootDataDirectory/output/db/p13/position_report_message.csv
csvToDb	13	observation_report_message	0	0	rootDataDirectory/input/csv/p13/observation_report_message.csv	rootDataDirectory/output/db/p13/observation_report_message.csv
csvToDb	13	latest_vessel_position	0	0	rootDataDirectory/input/csv/p13/latest_vessel_position.csv	rootDataDirectory/output/db/p13/latest_vessel_position.csv
csvToDb	13	call_for_fire	0	0	rootDataDirectory/input/csv/p13/call_for_fire.csv	rootDataDirectory/output/db/p13/call_for_fire.csv
dbToOws	6	position_report_message	0	0	rootDataDirectory/output/wfs/csv/p6/position_report_message.csv	rootDataDirectory/output/wfs/fromDb/p6/position_report_message.csv
dbToOws	6	latest_vessel_position	0	0	rootDataDirectory/output/wfs/csv/p6/latest_vessel_position.csv	rootDataDirectory/output/wfs/fromDb/p6/latest_vessel_position.csv
dbToOws	6	observation_report_message	0	0	rootDataDirectory/output/wfs/csv/p6/observation_report_message.csv	rootDataDirectory/output/wfs/fromDb/p6/observation_report_message.csv
dbToOws	6	call_for_fire	0	0	rootDataDirectory/output/wfs/csv/p6/call_for_fire.csv	rootDataDirectory/output/wfs/fromDb/p6/call_for_fire.csv
dbToOws	10	position_report_message	0	0	rootDataDirectory/output/wfs/csv/p10/position_report_message.csv	rootDataDirectory/output/wfs/fromDb/p10/position_report_message.csv
dbToOws	10	latest_vessel_position	0	0	rootDataDirectory/output/wfs/csv/p10/latest_vessel_position.csv	rootDataDirectory/output/wfs/fromDb/p10/latest_vessel_position.csv
dbToOws	10	observation_report_message	0	0	rootDataDirectory/output/wfs/csv/p10/observation_report_message.csv	rootDataDirectory/output/wfs/fromDb/p10/observation_report_message.csv
dbToOws	10	call_for_fire	0	0	rootDataDirectory/output/wfs/csv/p10/call_for_fire.csv	rootDataDirectory/output/wfs/fromDb/p10/call_for_fire.csv
dbToOws	13	position_report_message	0	0	rootDataDirectory/output/wfs/csv/p13/position_report_message.csv	rootDataDirectory/output/wfs/fromDb/p13/position_report_message.csv
dbToOws	13	latest_vessel_position	0	0	rootDataDirectory/output/wfs/csv/p13/latest_vessel_position.csv	rootDataDirectory/output/wfs/fromDb/p13/latest_vessel_position.csv
dbToOws	13	observation_report_message	0	0	rootDataDirectory/output/wfs/csv/p13/observation_report_message.csv	rootDataDirectory/output/wfs/fromDb/p13/observation_report_message.csv
dbToOws	13	call_for_fire	0	0	rootDataDirectory/output/wfs/csv/p13/call_for_fire.csv	rootDataDirectory/output/wfs/fromDb/p13/call_for_fire.csv

Appendix H: Vessel Position Report – NIEM XML

The following is the NIEM XML IEP for the initial USS JOHNPAULJONES position report.

```
<?xml version="1.0" encoding="UTF-8"?>
<posex:Message xsi:schemaLocation="http://niem.gov/niem/domains/maritime/2.1/position/exchange/3.2
../XMLSchemas/exchange/3.2/position-exchange.xsd" xmlns:m="http://niem.gov/niem/domains/maritime/2.1"
xmlns:mda="http://niem.gov/niem/domains/maritime/2.1/mda/3.2"
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:gml="http://www.opengis.net/gml/3.2" xmlns:ism="urn:us:gov:ic:ism"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" mda:securityIndicatorText="LEI"
mda:releasableNationsCode="USA" mda:releasableIndicator="true">
  <nc:DocumentCreationDate>
    <nc:Date>2011-12-01</nc:Date>
  </nc:DocumentCreationDate>
  <nc:DocumentExpirationDate>
    <nc:Date>2012-01-01</nc:Date>
  </nc:DocumentExpirationDate>
  <nc:DocumentCreator>
    <nc:EntityOrganization>
      <nc:OrganizationName>US Navy</nc:OrganizationName>
    </nc:EntityOrganization>
  </nc:DocumentCreator>
  <mda:RecordIDURI>00000001</mda:RecordIDURI>
  <mda:MessageStatusCode>Initial</mda:MessageStatusCode>
  <mda:MessageSourceSystemName>Track Source</mda:MessageSourceSystemName>
  <mda:ICISMMarkings ism:classification="U" ism:ownerProducer="USA"/>
  <mda:Vessel>
    <m:VesselAugmentation>
      <m:VesselCallSignText>DDG53</m:VesselCallSignText>
      <m:VesselHullNumberText>hull#1</m:VesselHullNumberText>
      <m:VesselIMONumberText>imo#1</m:VesselIMONumberText>
      <m:VesselMMSIText>mmsi#1</m:VesselMMSIText>
      <m:VesselName>USSJOHNPAULJONES</m:VesselName>
      <m:VesselNationalFlagISO3166Alpha3Code>USA</m:VesselNationalFlagISO3166Alpha3Code>
      <m:VesselOwner>
        <nc:EntityPerson>
          <nc:PersonName>
            <nc:PersonFullName>US Navy</nc:PersonFullName>
          </nc:PersonName>
          <nc:PersonNationalityISO3166Alpha3Code>USA</nc:PersonNationalityISO3166Alpha3Code>
        </nc:EntityPerson>
      </m:VesselOwner>
      <m:VesselSCONUMText>sco#1</m:VesselSCONUMText>
    </m:VesselAugmentation>
  </mda:Vessel>
  <mda:Position>
    <m:LocationPoint>
      <gml:Point srsName="http://metadata.ces.mil/mdr/ns/GISP/crs/WGS84E_2D">
        <gml:pos>37.1498819680237 -123.5583574667275</gml:pos>
      </gml:Point>
    </m:LocationPoint>
    <mda:PositionSpeedMeasure>
      <nc:MeasureText>0</nc:MeasureText>
      <nc:SpeedUnitCode>kt</nc:SpeedUnitCode>
    </mda:PositionSpeedMeasure>
    <mda:PositionCourseMeasure>
      <nc:MeasureText>47.56</nc:MeasureText>
      <m:AngleUnitText>deg</m:AngleUnitText>
    </mda:PositionCourseMeasure>
    <mda:PositionHeadingMeasure>
      <nc:MeasureText>47.56</nc:MeasureText>
      <m:AngleUnitText>deg</m:AngleUnitText>
    </mda:PositionHeadingMeasure>
    <mda:PositionNavigationStatus>
      <nc:StatusText>At Anchor</nc:StatusText>
    </mda:PositionNavigationStatus>
    <mda:PositionDateTime>
      <nc:DateTime>2013-04-13T18:00:00</nc:DateTime>
    </mda:PositionDateTime>
  </mda:Position>
</posex:Message>
```

Appendix I: Vessel Position Report – OWS Export GML XML

The following is the OWS output from the PostGIS database for the initial USS JOHNPAULJONES position report.

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:FeatureCollection xsi:schemaLocation="http://www.opengis.net/gml/3.2
http://falconservices.icl.gtri.org:8080/geoserver/schemas/gml/3.2.1/gml.xsd http://www.opengis.net/wfs/2.0
http://falconservices.icl.gtri.org:8080/geoserver/schemas/wfs/2.0/wfs.xsd mogiemaritime
http://falconservices.icl.gtri.org:8080/geoserver/wfs?service=WFS&version=2.0.0&request=DescribeFeatureType&typeName
=mogiemaritime%3Alatest_vessel_position" xmlns:mogie="mogie" xmlns:opengeo="http://opengeo.org"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:mogiemaritime="mogiemaritime" xmlns:test="http://opengeo.org/#test"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:mogieland="mogieland"
xmlns:wfs="http://www.opengis.net/wfs/2.0" xmlns:exactearth="exactearth">
  <wfs:boundedBy>
    <gml:Envelope>
      <gml:lowerCorner>37.1498819680237 -123.558357466727</gml:lowerCorner>
      <gml:upperCorner>38.0818013578952 -122.189322235658</gml:upperCorner>
    </gml:Envelope>
  </wfs:boundedBy>
  <wfs:member>
    <mogiemaritime:latest_vessel_position gml:id="latest_vessel_position.80">
      <gml:boundedBy>
        <gml:Envelope srsDimension="2" srsName="urn:ogc:def:crs:EPSG:4326">
          <gml:lowerCorner>37.1498819680237 -123.558357466727</gml:lowerCorner>
          <gml:upperCorner>37.1498819680237 -123.558357466727</gml:upperCorner>
        </gml:Envelope>
      </gml:boundedBy>
      <mogiemaritime:fid>80</mogiemaritime:fid>
      <mogiemaritime:call_sign>DDG53</mogiemaritime:call_sign>
      <mogiemaritime:hull_number>hull#1</mogiemaritime:hull_number>
      <mogiemaritime:imo_number>imo#1</mogiemaritime:imo_number>
      <mogiemaritime:mmsi>mmsi#1</mogiemaritime:mmsi>
      <mogiemaritime:vessel_name>USSJOHNPAULJONES</mogiemaritime:vessel_name>
      <mogiemaritime:national_flag_code>USA</mogiemaritime:national_flag_code>
      <mogiemaritime:owner_full_name>US Navy</mogiemaritime:owner_full_name>
      <mogiemaritime:owner_nationality_code>USA</mogiemaritime:owner_nationality_code>
      <mogiemaritime:sco_num>sco#1</mogiemaritime:sco_num>
      <mogiemaritime:speed>0.0</mogiemaritime:speed>
      <mogiemaritime:speed_units>kt</mogiemaritime:speed_units>
      <mogiemaritime:course>47.56</mogiemaritime:course>
      <mogiemaritime:course_units>deg</mogiemaritime:course_units>
      <mogiemaritime:heading>47.56</mogiemaritime:heading>
      <mogiemaritime:heading_units>deg</mogiemaritime:heading_units>
      <mogiemaritime:navigation_status>At Anchor</mogiemaritime:navigation_status>
      <mogiemaritime:latest_position_time>2013-04-13T18:00:00Z</mogiemaritime:latest_position_time>
      <mogiemaritime:latest_position>
        <gml:Point srsDimension="2" srsName="urn:ogc:def:crs:EPSG:4326">
          <gml:pos>37.1498819680237 -123.558357466727</gml:pos>
        </gml:Point>
      </mogiemaritime:latest_position>
      <mogiemaritime:time_stale>2013-04-14T00:48:00Z</mogiemaritime:time_stale>
      <mogiemaritime:latest_position_time_text>2013-04-13 18:00:00</mogiemaritime:latest_position_time_text>
      <mogiemaritime:time_stale_text>2013-04-14 00:48:00</mogiemaritime:time_stale_text>
      <mogiemaritime:unit_sidc_2525d>100330000121004000</mogiemaritime:unit_sidc_2525d>
    </mogiemaritime:latest_vessel_position>
  </wfs:member>
</wfs:FeatureCollection>
```

Appendix J: Position Data Processing

This appendix describes the method used to parse the latitude and longitude data into doubles and inserts the values into the PostGIS database. Source code is provided here so others could replicate the MOGIE process for parsing the position data and inserting it into a database.

The Java Application depicted in Figure 3 is where the PostgreSQL JDBC driver was used to populate the database.²⁹ This is where the longitude and latitude were converted into doubles, in Java, and then to a Point type in the database. Since this is the first area where error could be introduced, source code for the procedure is provided in

Figure 36 and the following text describes the actions taken in the code:

Figure 36: Line 1 initializes the coordinate input as it would come from the XML parser, a single string with latitude and longitude separated by a single space.

Figure 36: Lines 2 through 4 simply split the string into two strings, one for latitude and one for longitude.

Figure 36: Lines 5 and 6 actually parse the strings into double-precision 64-bit IEEE 754 floating point data types.³⁰

Figure 36: Line 7 simply defines a made up call sign.

Figure 36: Line 8 defines the EPSG code for the coordinate system. This is the value that will tell PostGIS which coordinate system is being used. (See <http://spatialreference.org/ref/epsg/4326/>)

Figure 36: Line 9 defines the SQL INSERT statement that will be used each time a new row is inserted. The question marks are placeholders for the actual values that will be added later. Notice the important `ST_GeomFromText(?,?)` part of the statement, this is where PostGIS comes in. Because we installed PostGIS on top of PostgreSQL, we now have access to the spatial objects and functions PostGIS provides. In order to use them, we must store our spatial data as a geometric type. `ST_GeomFromText(WKT, EPSG)` is a PostGIS function used to create such a geometric type from a geometry encoded in Well Known Text. (WKT) The PostGIS documentation states:³¹

The GIS objects supported by PostGIS are a superset of the "Simple Features" defined by the OpenGIS Consortium (OGC). As of version 0.9, PostGIS supports all the objects and functions specified in the OGC "Simple Features for SQL" specification.

Figure 36: Line 10 connects to the database.

²⁹ <http://docs.oracle.com/javase/6/docs/api/java/sql/Connection.html#prepareStatement%28java.lang.String%29>

³⁰ Java Language Specification Section 4.2.3, <http://docs.oracle.com/javase/specs/jls/se7/html/index.html>

³¹ http://postgis.net/docs/using_postgis_dbmanagement.html#RefObject

Figure 36: Line 11 creates the actual PreparedStatement object from the SQL string defined in line 9. from the official Java documentation, `connection.prepareStatement(sql)`.²⁹

[This method] creates a PreparedStatement object for sending parameterized SQL statements to the database. A SQL statement with or without IN parameters can be pre-compiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times.

Figure 36: Line 12 converts the longitude and latitude to WKT format. Now `pointAsWKT` is equal to `"POINT(-123.115604 37.996847)"`.

Figure 36: Lines 13 through 15 assign the actual values to the placeholders in the prepared statement. Line 13 sets the first question mark, in line 9, to the made up call sign. Line 14 sets the second question mark to the WKT point defined in line 12. Line 15 sets the third question mark to the EPSG code defined in line 8. At this point, if the SQL statement were printed it would look like this:

```
INSERT INTO maritime.latest_vessel_position(call_sign, latest_position)
VALUES ( 'a call sign', ST_GeomFromText('POINT(-123.115604 37.996847)', 4326))
```

Figure 36: Line 16 inserts the values in the database, and line 17 closes the data base connection.

Figure 37 provides the code necessary to create a database table that supports running the source code in Figure 36.

```

//initial input as it would come from the DOM parser, after parsing the GML
1: String latlonAsString = "37.996847 -123.115604";

//convert the latitude and longitude to doubles
2: String[] latlonAsStringArray = latlonAsString.split(" ");
3: String latitudeAsString = latlonAsStringArray[0];
4: String longitudeAsString = latlonAsStringArray[1];
5: double latitudeAsDouble = Double.parseDouble(latitudeAsString);
6: double longitudeAsDouble = Double.parseDouble(longitudeAsString);

//set values for the call sign and EPSG code
7: String callSign = "a call sign";
8: int epsgCode = 4326;

//define the insert statement
9: String sql =
    "INSERT INTO maritime.latest_vessel_position(call_sign, latest_position) "+
    "VALUES (?, ST_GeomFromText(?,?))";

//connect to the database
10: Connection connection = DriverManager.getConnection(
    "jdbc:postgresql://localhost:54321/mogie_database", "user_name",
    "password");

//create the PreparedStatement object
11: PreparedStatement ps = connection.prepareStatement(sql);

//convert the longitude and latitude to Well Known Text (WKT) format
12: String pointAsWKT =
    "POINT(" + longitudeAsDouble + " " + latitudeAsDouble + ")";

//set the prepared statement parameters
13: ps.setString(1, callSign);
14: ps.setString(2, pointAsWKT);
15: ps.setInt(3, epsgCode);

//execute the prepared statement
16: ps.execute();

//close the database connection
17: connection.close();

```

Figure 36: Source code for processing the location data


```
//This source is provided as a convenience,  
//it assumes the reader is familiar with Java, JDBC, SQL,...  
String dbName = "mogie_database";  
  
String createDb = "CREATE DATABASE "+ dbName+  
    " WITH ENCODING='UTF8' OWNER=opengeo "+  
    "TEMPLATE=template_postgis CONNECTION LIMIT=-1";  
String createSchema = "CREATE SCHEMA maritime "+  
    "AUTHORIZATION opengeo_user";  
String createTable = "CREATE TABLE maritime.latest_vessel_position"+  
    "(call_sign character varying(30),"+  
    "latest_position geometry(Point,4326))";  
  
//connect and create the database  
Connection connection = DriverManager.getConnection(  
    "jdbc:postgresql://localhost:54321/postgres",  
    "opengeo_user",  
    "opengeo_password");  
connection.createStatement().execute(createDb);  
connection.close();  
  
//connect to new database, create schema and create table  
connection = DriverManager.getConnection(  
    "jdbc:postgresql://localhost:54321/"+dbName,  
    "opengeo_user",  
    "opengeo_password");  
connection.createStatement().execute(createSchema);  
connection.createStatement().execute(createTable);  
connection.close();
```

Figure 37 Source code to set up environment to run code in the previous figure

Appendix K Embedding GML Best Practice Paper

This section contains a whitepaper written by Douglas D. Nebert, Senior Advisor for Geospatial Technology, System-of-Systems Architect, FGDC Secretariat. It was referenced by both MOGIE and GEO4NIEM projects. This version of the paper was circulated via email in October 2012; no additional information is known regarding the original date of its publication.

Embedding GML Best Practice Paper

Overview

This note provides a best practice guideline for the embedding of GML inside an XML language that is not specified by a GML application schema, in fact may not be specified by an XML Schema at all. The intent of this guideline is to encourage this practice while at the same time achieving as much standardization as possible, and maintaining consistency with GML itself.

The basic structure of GML is based on the so called “object-property-value” rule. This means that objects (things, entities) are encoded as elements, and their element children are properties of the parent object. The element children of these property elements are the values of the property for complex valued properties, or are just the element content in the case of properties of simple type.

When embedding GML in another language, one should attempt to stick to this basic GML model by using the following best practice guidelines.

1. **Only embed GML objects.** These are things in the GML core schemas, whose content models derive directly or indirectly from `AbstractGMLType`. Such objects include geometry elements, topological elements, observations, coordinate reference systems and temporal elements.
2. **Include any GML dependent attributes, elements and content models.** The embedded GML object may depend on attributes, elements etc that are defined elsewhere in the GML core schemas. Be sure to include these in your embedding.
3. **Embed GML objects in “properties”.** Where possible, identify container elements in the parent non-GML language that behave like attributes or properties. Examples might include Slots in ebRIM, or the “where” property in GeoRSS.
4. **Create a GML profile schema.** Create a GML profile schema by taking the desired embedded elements and their dependencies and constructing a single schema file, containing them. You need then only import this single profile schema into your non-GML language.

The following sections provide additional details using these simple rules and illustrates them with some simple examples.

Only Embed GML Objects

Essentially this means do not embed GML properties, GML attributes/attribute groups, nor use GML types to define the content models of elements in the non-GML language. While all of these are possible, GML objects are complete GML components and thus are more readily employed in the non-GML language. Additionally there is less opportunity for confusion with GML objects having the same content model. This practice also recommends against the embedding of GML properties (e.g.

pointProperty) as the semantics of such properties is clear in the GML context, but would be much less so in the non-GML language in which GML is being embedded.

Note that this does not mean you cannot define new application objects which use GML content models. For example, in an emergency response language, it is desired to define an ImpactArea using a GML Polygon. It is recommended that this be encoded as follows:

```
<abc:ImpactArea>
  <abc:extent>
    <gml:Polygon gml:id="...">
      <gml:exterior>
        <gml:LinearRing> ... </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </abc:extent>
</abc:ImpactArea>
```

Rather than writing:

```
<abc:ImpactArea>
  <gml:exterior>
    <gml:LinearRing> ... </gml:LinearRing>
  </gml:exterior>
</abc:ImpactArea>
```

The semantics of the former encoding is clearer in that the ImpactArea while having Polygon extent is NOT a Polygon as implied in the second encoding.

If the parent language wishes to define objects already defined in GML (e.g. Point) then this best practice would require that they use the GML defined objects rather than creating new ones with the same content model.

For example, the following encoding (in an XML schema with a non-GML namespace) would not comply with this best practice:

```
<element name="Point" type="gml:PointType"/>
```

There are cases where one might like to add properties (e.g. special ID attributes or elements) to a GML defined object. Of course this can be done by inheriting from the GML content models, and adding the desired attributes, or elements. This should be avoided if possible, as it complicates the processing software for the non-GML language (i.e. requires schema parsing techniques).

Include any dependent attributes, type definitions etc.

Of course this is required to construct a valid schema. It is recommended that the user create a GML profile schema file containing the needed objects, and all dependent attributes, elements, and type definitions. Automated tools for this purpose are available for most versions of GML.

The profile schema file then isolates all GML content used by the non-GML language, and only this file need be imported by the XML Schema for the non-GML language.

Embed GML Objects in “Properties”

Properties in GML frequently represent associations between the parent and child elements of the property, the property in effect defining the context of the child object in relation to the parent.

Consider the following simple example:

```
<abc:Road id= "j12k">
  <abc:centerline>
    <gml:LineString> ... </gml:LineString>
  </abc:centerline>
  <abc:rightEdge>
    <gml:LineString> ... </gml:LineString>
  </abc:rightEdge>
</abc:Road>
```

The properties `centerline` and `rightEdge` define the relationship of the geometric objects (both `LineStrings`) to the `Road` object.

In GML Application Schemas this structure of objects-properties-values is rigidly enforced. In a non-GML language there may be no such requirement, however, it is considered best practice to look for (or if possible) create properties or elements with a similar role as the container for GML objects.

The following encoding would then not comply with this best practice:

```
<abc:Road>
  <gml:LineString> ... </gml:LineString>
</abc:Road>
```

Most non-GML languages will have some sort of property construct and it will usually be possible to comply with this best practice.

Following this best practice will ensure that GML usage is clearer in the non-GML language.