

ARMY RESEARCH LABORATORY



**Lessons Learned With a Global Graph and Ozone Widget
Framework (OWF) Testbed**

by Mark R. Mittrick, John T. Richardson, and Michael H. Lee

ARL-TR-6440

May 2013

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Aberdeen Proving Ground, MD 21005-5066

ARL-TR-6440

May 2013

Lessons Learned With a Global Graph and Ozone Widget Framework (OWF) Testbed

Mark R. Mittrick, John T. Richardson, and Michael H. Lee
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188		
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) May 2013		2. REPORT TYPE		3. DATES COVERED (From - To) October 2011–August 2012	
4. TITLE AND SUBTITLE Lessons Learned With a Global Graph and Ozone Widget Framework (OWF) Testbed			5a. CONTRACT NUMBER		
			5b. GRANT NUMBER		
			5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S) Mark R. Mittrick, John T. Richardson, and Michael H. Lee			5d. PROJECT NUMBER		
			5e. TASK NUMBER		
			5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-CII-C Aberdeen Proving Ground, MD 21005-5066			8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TR-6440		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSOR/MONITOR'S ACRONYM(S)		
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)		
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT Computational services that will dramatically increase the operational and tactical understanding of the local operational human environment are under development. The goal is to give commanders insight into the population and its culture in order to enhance operational effectiveness and reduce military and civilian conflict. Utilizing innovative research in the discovery and social human network analysis, the U.S. Army Research Laboratory has developed advanced information exploitation services to enhance Army intelligence. These services are targeted to support the military analyst in revealing underlying relationships, increasing situational understanding, and providing support for tactical operations.					
15. SUBJECT TERMS global graph, ozone widget framework, distributed common ground system, web service, OWF					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Mark Mittrick
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			UU

Contents

List of Figures	iv
List of Tables	iv
1. Introduction	1
2. Background	1
3. The Global Graph (SQL Version)	4
3.1 Install Database and Utilities.....	4
3.1.1 Installing PostgreSQL 8.4.x	4
3.1.2 Install pgAdmin Version 1.16.x	5
3.1.3 Install PostGIS Support	5
3.1.4 Edit the Postgres Configuration File Data\pg_hba.conf.....	5
3.1.5 Configure PostgreSQL Users/Roles	6
3.2 Load Global Graph Schema	6
3.2.1 Import GG Database Dump File (from Potomac Fusion)	6
3.2.2 Restore GG Database From Previous Backup File (from pgAdmin Tool)	6
3.2.3 Install Global Graph Rest Services	7
3.2.4 Tomcat Setup.....	7
3.2.5 Modify OWF Configuration Files	7
3.2.6 Configure and Deploy Global Graph Web Application Archive (WAR)	8
3.2.7 Deserialization Issues with the Global Graph Rest Service	9
4. Ozone Widget Framework (OWF)	12
4.1 Configure and Install OWF	12
4.2 Deploy Global Graph Core Widgets	14
5. Conclusion/Future	15
6. References/Other Documentation	15
List of Symbols, Abbreviations, and Acronyms	16
Distribution List	17

List of Figures

Figure 1. Basic system architecture.	3
---	---

List of Tables

Table 1. Log levels.....	8
--------------------------	---

1. Introduction

The U.S. Army Research Laboratory (ARL) Tactical Information Fusion Branch (TIFB) is conducting research to advance the use of information analytics for intelligence operations. Focusing on the Company Intelligence Support Team (COIST), a number of information-processing techniques are being developed and analyzed for improvement of situational understanding. The Distributed Common Ground Station - Army (DCGS-A) is a program of record that is at the forefront of the U.S. Army's networked-based systems architecture. Leveraging this fact, ARL has chosen to focus on algorithmic approaches, which transitions easily with its web-based widget framework.

Algorithms under consideration include clustering, graph-based, and statistical approaches. In August 2012, ARL conducted laboratory and field testing of one information analytics method — multidimensional scaling. Testing in the field yielded useful feedback on the efficacy of the MDS technique.¹ The algorithm was embedded in the DCGS-A Ozone Widget Framework (OWF) and used the Global Graph for data transfer.

This report will provide background on the DCGS-A OWF, the Global Graph, the ARL implementation, as well as lessons learned from the use of the architecture for research applications.

2. Background

To understand how to effectively exploit the DCGS-A Global Graph, one must first understand the OWF. OWF, according to its creator — Potomic Fusion, is a combination layout manager and messaging mechanism for hosting widgets within a web browser. Its goal is to provide a common thin-client environment for dynamic analytic workflows using lightweight web-applications (widgets) distributed across the enterprise.*

¹Hanratty, T.; Richardson J. *The Heterogeneous Data-reduction Proximity Tool – A Visual Analytic for High-Dimensional Data Exploitation*; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, March 2013.

*<http://widget.potomacfusion.com/main/home>

Outlined in figure 1 is Potomic Fusion's diagram showing the OWF basic system architecture and components. In this diagram, computational services in the form of application widgets interact between a user and the system.

These widgets manipulate data from the system and then display results in the users browser. Interaction between widgets and the Global Graph is accomplished via JavaScript Object Notation (JSON) structures. Interaction among widgets is also possible with eventing, which is an asynchronous publish/subscribe messaging system.

The simplest deployment scenario would have all of the components on the same physical machine; however, they can all be broken out if necessary. Additionally, not all components are necessary depending on your goal.

In figure 1, OWF is accessed by a web browser (Firefox) connecting to a web server via the Hyper Text Transfer Protocol (HTTP) over Secure Sockets Layer (SSL) protocol on the local network. From there, the user must authenticate to OWF via the Central Authentication Service (CAS).

In this diagram, CAS interacts with Lightweight Directory Access Protocol (LDAP), but this is an optional component and typically only used in large production environments. Once authenticated, the user has access to run widgets. Once the user is done, they just simply log off.

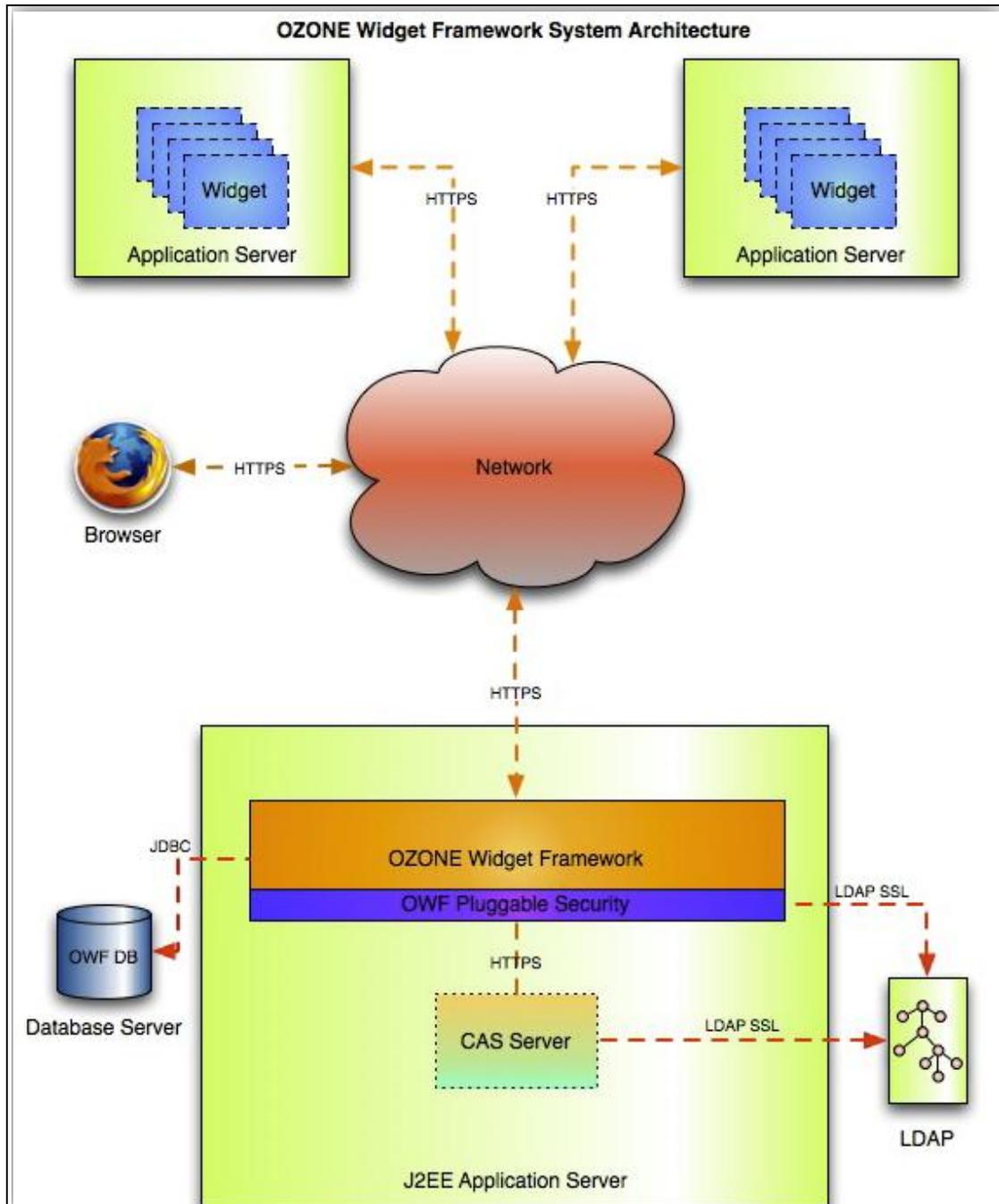


Figure 1. Basic system architecture.

3. The Global Graph (SQL Version)

The first step in exploiting the Global Graph (GG) is understanding the installation process. This section details the steps required to install and configure the SQL version of the GG.

The steps include:

- Installing of the database and utilities
- Loading the data into GG
- Installing/Configuring of Apache Tomcat
- Configuring/Deploying GG Web Service

3.1 Install Database and Utilities

The GG can be customized to run in a variety of operating system and database environments. The following is one example.

Requirements are: Java 1.6 + and a Relational Database Management System (RDBMS)

ARL chose PostgreSQL as its database based on the recommendation of other developers and support by Potomac Fusion.

***Lesson Learned:** We originally tried to use MySQL as our database, because we were more familiar with it, but since the database dumps as well as most of the documentation and developers supported PostgreSQL, we decided to go that route, which ended up saving us time and effort down the road.*

3.1.1 Installing PostgreSQL 8.4.x

1. Download and install PostgreSQL for your operation system version. Available from <http://www.enterprisedb.com/products-services-training/pgdownload>.

Ex: postgresql-8.4.15-windows.exe (~45 MB)

2. Set up administrator password

Ex: Postgres12345

3. Keep default listen port: 5432

***Lesson Learned:** Make certain to set the PostgreSQL Administrator password to something secure as well as something you will remember, as you will need to set up all of the configuration files to use it.*

3.1.2 Install pgAdmin Version 1.16.x

1. Download and install pgAdmin for your operating system version. Available from <http://www.pgadmin.org/download>.

Ex: pgadmin3-1.16.0.zip (~15 MB)

***Lesson Learned:** If you create a backup of your PostgreSQL database tables using a certain version of pgAdmin, make certain to use that same version to load it back. In our experience, using two different versions of pgAdmin was problematic.*

3.1.3 Install PostGIS Support

1. Download and install PostGIS for your operating system version. This can be done by running the “Application Stack Builder” Tool that comes with PostgreSQL or by downloading it manually from <http://postgis.refractor.net/download>.

Ex: postgis-pg84-setup-2.0.2.exe (~5 MB)

2. If using the “Application Stack Builder Tool” on the “Choose Components” screen, ensure both PostGIS and create option are checked.
3. Use administrator password and port from PostgreSQL 8.4.x install.
4. Enter “postgis” for the database name.
5. Select “Yes” when prompted to install “shp2pgsql” plugin.

3.1.4 Edit the Postgres Configuration File Data\pg_hba.conf

Modify the following lines in order to allow internal access to computers on the same subnet other than just the localhost. Make sure to use the IP address appropriate for your server. The “pg_hba.conf” file can be accessed via pgAdmin, under the “Tools/Server Configuration” menu. Additionally, you can navigate to the source directory and edit it manually with Notepad or some other text editor. **Note:** The “X’s” below represent your IP Address Scheme.

```
# IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all X.X.X.1/0 md5
host all all X.X.X.1/0 trust
# IPv6 local connections:
#host all all ::1/128 md5
```

***Lesson Learned:** This is very important to get correct. If you don’t have the proper permissions set up, then the global graph won’t be able to connect to the database. The X.X.X.1/0 allows any computer on the local subnet access to connect. You can adjust it to make it more or less restrictive. The md5/trust section refers to the type of authentication you are using. We ended up using both at different times, so we added both lines.*

3.1.5 Configure PostgreSQL Users/Roles

Next, open the “pgAdmin” tool you just installed and proceed to do the following:

1. Create a new user.
2. Edit the user’s permissions (grant all).
3. Create a new database called “globalgraph.”
4. Assign the new user to the new database.

3.2 Load Global Graph Schema

Once the database and utilities have been successfully installed and configured, there are two methods for loading data into the database. First, you can ingest the database dump that was provided by Potomac Fusion (PFI). The dump file contained a set of database tables with sample data for testing purposes. Second, you can restore from a pgAdmin backup file. Typically, since you don’t have any previous backups available, this method is reserved for use after the initial database load.

3.2.1 Import GG Database Dump File (from Potomac Fusion)

1. Open a command prompt in C:\Program Files\PostgreSQL\8.4\bin.
2. Copy the dump file, “dcgs-vcab-dump.sql” to c:\temp (for command line convenience).
3. Execute command: `createdb -U postgres globalgraph.`
4. Execute command: `psql -U postgres -d globalgraph -f c:\temp\dcgs-vcab-dump.sql.`

3.2.2 Restore GG Database From Previous Backup File (from pgAdmin Tool)

1. Open pgAdmin and connect to your server.
2. Locate the backup file, “globalgraph.backup” (or whatever you have named it).
3. Create a new database called globalgraph with the appropriate permissions.
4. Right click on the new globalgraph database and choose the “Restore” option
5. Select the backup file you located earlier and click ok to start the restore process

Lesson Learned: *Make certain to use the same version of pgAdmin, and make certain to use that same version of pgAdmin to load the database that was used to generate it. In our experience, using two different versions of pgAdmin was problematic. Also, note that if it doesn’t work the first time, clear the database completely (right click on the database name and select “delete/drop”) and try it again.*

3.2.3 Install Global Graph Rest Services

In order to set up the Global Graph Rest Services, you will need to have the following dependencies installed: Java 1.6 + and a Tomcat 6 +. Tomcat comes with the OWF bundle or can be downloaded separately from the Apache Tomcat project (<http://tomcat.apache.org>). In our discussion, we will be using Tomcat bundled with OWF.

3.2.4 Tomcat Setup

1. The following change may be necessary to make to the following script:

```
apache-tomcat-7.x\bin\catalina.bat
```

Set JRE_HOME=C:\Program Files\java\jdk1.6.x (or wherever Java has been installed to)

2. In the computer's Environment, set the following:

```
CATALINA_HOME = C:\OWF_x.x\apache-tomcat-x.x
```

3.2.5 Modify OWF Configuration Files

Several of the GG Tomcat configuration files will need to be edited in order to work properly. This consists of changing the default server name with your server name, as well as changing the default PostgreSQL user name and password with your user name and password. Below are the required configuration files that need to be changed and the elements that need to be changed within them.

1. C:\OWF_x.x\apache-tomcat-x.x\webapps\gg\META-INF\context.xml
 - a. Database User Name
 - b. Database Password
 - c. Server Name
2. C:\OWF_x.x\apache-tomcat-x.x\webapps\gg\WEB-INF\classes\jdbc.properties
 - a. Database User Name
 - b. Database Password
 - c. Server Name
3. C:\OWF_x.x\apache-tomcat-x.x\webapps\gg\WEB-INF\classes\webserver.properties
 - a. Server Name
4. C:\OWF_x.x\apache-tomcat-x.x\webapps\gg\WEB-INF\spring-config-rest-services-web.xml

For this file – configure the appropriate authentication method and comment the other one out.

```
<import resource="classpath:spring-config-rest-services-digest-  
auth.xml"/>  
<import resource="classpath:spring-config-rest-services.xml"/>
```

3.2.6 Configure and Deploy Global Graph Web Application Archive (WAR)

1. Verify that an instance of Tomcat is running.
2. Unzip the file globalgraph-rest-services.war into a directory called gg in the directory `${CATALINA_HOME}/webapps/gg`.
3. Copy the postgres*.jar and postgis*.jar from the dist/all-lib directory into the `${CATALINA_HOME}/lib` directory.
4. Alter the log4j properties file to appropriate log settings for the application. Log file names and log levels may be altered to more appropriate values (see table 1).

Table 1. Log levels.*

Log Level	Description
Severe (Highest)	Captures exception and error
Warning	Warning messages
Info	Informational message, related to the server activity
Config	Configuration message
Fine	Detailed activity of the server transaction (similar to debug)
Finer	More detailed logs than fine
Finest (Least)	Entire flow of events (similar to trace)

5. The application is now configured and can be started up via command line or the Apache Tomcat Windows Service.
6. The application can be tested by entering `http://localhost:<port>/gg/entity/Person/<valid-uuid>` into browser to perform retrieval for given person.
7. The request will pop up a window in the browser for you to enter a user name and password. The system comes preloaded with an admin user: `user1/ggsecret`. You may change the login information to your suit your needs ant any time.

Lesson Learned: Make certain all configurations files are correct and that the user names and passwords all match or it will not work. Also, if your system is a standalone system, please ensure that the hosts file (Windows -> System32 -> drivers -> etc -> hosts) is set up to identify your systems name with its IP Address. If not, you won't be able to connect to the Global Graph.

*<http://my.safaribooksonline.com/book/-/9781849516624/6dot-logging-in-tomcat-7/id286772259>

3.2.7 Deserialization Issues with the Global Graph Rest Service

Once the Global Graph Rest Service is setup and running, it will respond to entity queries (e.g., Person, Organization) with the JSON string, which captures all attributes (e.g., name, address, nationality, etc.) of the requested entity. Client application will consume this information by either parsing the JSON string or deserializing it to a Java object. Deserialization is the preferred method because string parsing may introduce errors and it will need to be carefully reconstructed when the JSON format is altered by the Rest Service creator. PFI, creator of the Global Graph Rest Service, does not describe how clients should deserialize JSON string to PFI Entity Java objects in the user's manual. Deserializing PFI JSON string to PFI Java object is particularly more desirable than deserializing it to a custom-made Java object, because a custom Java object will need to be carefully updated by the clients as PFI changes the JSON structure. PFI provided the JSON deserialization method upon request.

***Lesson Learned:** At the time of writing this report, PFI Global Graph Entity serialization/deserialization methods are located in `globalgraph-javamodel-1.4.6-A2SF-RC2-postgres.jar`. PFI Entity Java objects are serialized by invoking `com.potomacfusion.globalgraph.util.JsonUtils.marshall(Class<T> arg0, T arg1)`.*

Example:

```
String p1Marshaled = JsonUtils.marshall(Person.class, p1);
```

PFI Entity JSON strings are deserialized by invoking `com.potomacfusion.globalgraph.util.JsonUtils.unmarshall(Class<T> arg0, String arg1)`

Example:

```
Person p2 = JsonUtils.unmarshall(Person.class, p1Marshaled);
```

Array of PFI Entity JSON strings are deserialized to a Collection by invoking `com.potomacfusion.globalgraph.util.JsonUtils.parseEntities(JsonParser arg0, Map<String, Class> arg1)`.

Example:

```
ArrayList<IIdentifiable> entities = (ArrayList<IIdentifiable>) JsonUtils.  
parseRelationships(jsonParser, entityMap);
```

Array of PFI Relationship JSON strings are deserialized to a Collection by invoking `com.potomacfusion.globalgraph.util.JsonUtils.parseRelationships(JsonParser arg0, ObjectMapper arg1)`.

Example:

```
ArrayList<IIdentifiable> relationships = (ArrayList<IIdentifiable>) JsonUtils.  
parseRelationships(jsonParser, objectMapper);
```

Lesson Learned: *There are different set of jars for SQL based Global Graph and DSC Global Graph. The globalgraph-javamodel-1.4.6-A2SF-RC2-postgres.jar referenced above is only found in the SQL based Global Graph. Clients will need to contact PFI to request serialization/deserialization methods for the DSC Global Graph.*

Testing showed all PFI serialization/deserialization methods worked well when the client application ran on the machine where the Global Graph PostgreSQL database was installed. However, there were problems invoking *JsonUtils.parseRelationships(JsonParser, ObjectMapper)* when the client application ran on a different machine than where the Global Graph PostgreSQL database was installed. In the latter case, *parseRelationships* method consistently returned an empty list instead of list of relationship objects. This problem indicated that *parseRelationships* method (or another dependent method) was attempting to access the Global Graph PostgreSQL database, but the database connection URL was incorrect somewhere. After searching for connection strings, two configuration files (*hibernate.cfg.xml* and *jdbc.properties*) were found in the *globalgraph-javamodel-1.4.6-A2SF-RC2-postgres.jar*.

Lesson Learned: *Database connection URL and credentials are specified in hibernate.cfg.xml and jdbc.properties, both of which are embedded in globalgraph-javamodel-1.4.6-A2SF-RC2-postgres.jar.*

The following four lines need to be updated in *hibernate.cfg.xml* with correct URL and credentials:

```
<property name="connection.url">jdbc:postgresql://localhost:5432/  
PFI_GG_DB_NAME?useUnicode=true&characterEncoding=utf-8</property>  
<property name="connection.driver_class">org.postgresql.Driver</property>  
<property name="connection.username"> USERID</property>  
<property name="connection.password"> PASSWORD</property>
```

The following eight lines need to be update in *jdbc.properties* with correct URL and credentials:

```
jdbc.driverClassName=org.postgresql.Driver  
jdbc.url=jdbc:postgresql://localhost:5432/PFI_GG_DB_NAME?useUnicode=true&ch  
aracterEncoding=utf-8  
jdbc.username=USERID
```

```
jdbc.password=PASSWORD
jdbc.validationQuery=select 1 as test
time.zone.init=SET timezone = 'UTC'
hibernate.dialect=org.hibernate.spatial.postgis.PostgisDialect
globalgraph.jndiName=java:comp/env/jdbc/globalgraph
```

After hibernate.cfg.xml and jdbc.properties are updated, they must be repackaged back into globalgraph-javamodel-1.4.6-A2SF-RC2-postgres.jar before deployment.

Lesson Learned: *PostgreSQL Database refuses all connections it receives from any remote address out-of-the-box for security purpose. Please refer to PostgreSQL documentation to manually reconfigure pg_hba.conf to authorize remote connection.*

Finding “localhost” specified as the database connection URL in the configuration files was an encouraging sign, which explained why *JsonUtils.parseRelationships(JsonParser, ObjectMapper)* method only worked when the client application ran on the machine where the Global Graph PostgreSQL database was installed. Replacing the existing “localhost” hostname in the URL was logically believed to be the solution to the relationships deserialization error. However, updating the configuration files did not correct the problem and *JsonUtils.parseRelationships(JsonParser, ObjectMapper)* continued to produce an empty list. It is believed that PFI has hard-coded “localhost” somewhere in their code and as a result *JsonUtils.parseRelationships(JsonParser, ObjectMapper)* will work only if it is invoked on the same machine that the PostgreSQL Global Graph database was installed. At the time of writing this report, this error remains unresolved. There are two workarounds to this problem.

- (1) Not to use PFI methods to serialize/deserialize JSON strings to PFI Java objects. This workaround requires the client to create custom entity Java classes (e.g., Person, Organization, Relationship, etc.) and serialize/deserialize using the open-source Jackson library. Downside of this option is that creating custom entity classes to match PFI’s extensive JSON structure is very time consuming and more importantly all these classes will need to be updated immediately after PFI changes the Global Graph Rest Service API to reflect the changes.
- (2) Copy a snapshot of the Global Graph PostgreSQL database on the same machine that the client application must run. The obvious downside to this option is that a snapshot of the Global Graph database will not have the same data as the primary Global Graph database shortly after a snapshot is created, and will need to be regularly synchronized in a later time. Synchronizing the database can be a very time consuming process depending on how large the database is and the client application will not be usable during the downtime.

Both options are not preferred, but option no. 1 is the only viable workaround. Our team decided to create custom entity classes and serialize/deserialize with the Jackson library.

4. Ozone Widget Framework (OWF)

OWF, as mentioned previously, is a combination layout manager and messaging mechanism for hosting application widgets within a Web browser. Below are the basic steps to get OWF installed and configured. For more detailed information please see the references section.

4.1 Configure and Install OWF

1. Download and unzip OWF-bundle-X-GA.zip.
2. Several of the OWF configuration files will need to be edited in order to work properly. This consists of changing the default server name with your server name, changing the default PostgreSQL user name and password with your user name and password, and adjusting the port numbers as necessary. The following are the required configuration files that need to be changed and the elements that need to be changed within them.

- a. C:\OWF_x.x\apache-tomcat-x.x\webapps\cas\WEB-INF\cas.properties
 - i. Server Name
- b. C:\OWF_x.x\apache-tomcat-x.x\conf\server.xml
 - i. Port Number
- c. C:\OWF_x.x\apache-tomcat-x.x\lib\CASSpringOverrideConfig.xml
 - i. Server Name
 - ii. Port Number
- d. C:\OWF_x.x\apache-tomcat-x.x\lib\OzoneConfig.properties
 - i. Server Name
 - ii. Port Number
- e. C:\OWF_x.x\apache-tomcat-x.x\lib\OWFConfig.groovy
 - i. Database User Name
 - ii. Database Password
 - iii. Server Name
 - iv. JDBC Driver

- f. C:\OWF_x.x\apache-tomcat-x.x\lib\OWFConfig.xml
- i. Server Name

3. Generate / Configure Server Certificates.

(Example information below — change as necessary)

- a. Open a Command Prompt and navigate to the apache certs folder
- b. Ensure the Java bin folder is in your PATH environment
- c. Command: `keytool -genkey -alias %FULL_SERVERNAME% -keyalg rsa -validity 3650 -keystore OTM.jks`
- d. Keystore password: globalgraph
- e. First and last name: %FULL_SERVERNAME%
- f. Org unit: ARL
- g. Organization: CISD
- h. City: APG
- i. State: MD
- j. Country code: US

Lesson Learned: *Generating the proper certificates took multiple attempts to get correct in our experience. Please see Potomic Fusion's documentation for more details if things are not working. Additionally, if you don't want to do this manually or are having trouble, there are tools on the Web that can generate these certificates for you.*

Finally, make certain to load these newly created certificates into your web browser. Note that these certificates might produce warning messages. You can allow exceptions to be added in order to permit access to OWF.

In Internet Explorer, client certificates can be added by selecting Tools -> Internet Options -> Content -> Certificates -> Personal, and then clicking the Import button.

In Firefox, this menu is accessed via Tools -> Options -> Advanced -> Encryption -> View Certificates -> Your Certificates -> Import.

Finally, we suggest that you edit the certs trust (Firefox only) once you have imported it. This can be done via Tools -> Options -> Advanced -> Encryption -> View Certificates -> Servers -> Edit Trust.

- 4. Update catalina.bat (tomcat\bin) to point to the new JKS file w/ appropriate password.

5. Update service.bat (tomcat\bin) to point to the new JKS file w/ appropriate password (for running as service).
6. Update server.xml (tomcat\conf) to point to the new JKS file w/ appropriate password.

4.2 Deploy Global Graph Core Widgets

As part of the Global Graph distribution, several core widgets are provided. These widgets are necessary in order to interact with OWF and its data.

The core widgets are as follows:

- Data Generator Widget
- Filter Widget
- Import Widget
- Export Widget
- Property Editor Widget
- Search Widget
- Viewer Widget
- Clipboard Widget
- Timeline Widget
- Timewheel Widget



In order to get the core widgets to work, the war files need to be deployed in the C:\OWF_x.x\apache-tomcat-x.x\webapps folder and then configured for use by an OWF administrator.

5. Conclusion/Future

With some time and effort, a Global Graph environment can be set up to develop widgets/tools for interaction with DCGS-A.

The installation and configuration information mentioned in this paper is an overview with lessons learned of how we went about setting up our particular environment in order to develop our tool for the Global Graph. It is by no means an all-inclusive list. More detailed information is available from Potomac Fusion. For that information, please see section 6.

In the future, TIFB plans to expand and refine the widgets that were developed and to eventually transition them to the DCGS-A Cloud.

6. References/Other Documentation

For additional information please see:

- Mittrick, M. *Smartphone Application Enabling Global Graph Exploitation and Research*; ARL-TR-6439; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, May 2013.
- Hanratty T.; Richardson, J. *The Heterogeneous Data-reduction Proximity Tool – A Visual Analytic for High-Dimensional Data Exploitation*; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, in press.
- Potomac Fusion Documentation:
<http://widget.potomacfusion.com/main/home>
<http://www.forge.mil/Community.html>

List of Symbols, Abbreviations, and Acronyms

ARL	U.S. Army Research Laboratory
CISD	Computational Information Sciences Directorate
COIST	Company Intelligence Support Team
DCGS-A	Distributed Common Ground System - Army
GG	Global Graph
HVI	High-valued individuals
ISD	Information Sciences Division
OWF	Ozone Widget Framework
PFI	Potomac Fusion
TIFB	Tactical Information Fusion Branch
WAR	Web Application Archive

NO. OF
COPIES ORGANIZATION

1 DEFENSE TECHNICAL
(PDF) INFORMATION CTR
DTIC OCA

1 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CIO LL

1 GOV'T PRINTG OFC
(PDF) A MALHOTRA

1 ARMY G2
(PDF) D WALSH

1 MULTISOURCE INFORMATION
(PDF) FUSION
RSRCH PROFESSOR
(EMERITUS) J LLINAS

1 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CII
B BROOME

1 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CII A
S H YOUNG

2 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CII B
M LEE
L TOKARCIK

1 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CII T
V M HOLLAND

ABERDEEN PROVING GROUND

1 US ARMY CERDEC I2WD
(PDF) RDER IWP
D PORTER

24 DIR USARL
(14 PDF RDRL CII C
10 HC) B BODT
E BOWMAN
F BRUNDICK
J DUMER
T HANRATTY
C HANSEN

NO. OF
COPIES ORGANIZATION

E HEILMAN
S KASE
M MITTRICK (1 PDF, 10 HC)
A NEIDERER
K OGAARD
J RICHARDSON
H ROY
M THOMAS

INTENTIONALLY LEFT BLANK.