Testbed-12 Testbed-12 GeoPackage Mobile Apps Integration Engineering Report

Table of Contents

Publication Date: 2017-05-15

Approval Date: 2017-01-26

Posted Date: 2016-10-31

Reference number of this document: OGC 16-030

Reference URL for this document: http://www.opengis.net/doc/PER/t12-A083

Category: Public Engineering Report

Editor: Jeff Yutzler

Title: Testbed 12 GeoPackage Mobile Apps Integration Engineering Report

---

OGC Engineering Report

COPYRIGHT

WARNING

This document is an OGC Public Engineering Report created as a deliverable of an initiative from the OGC Innovation Program (formerly OGC Interoperability Program). It is not an OGC standard and 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. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

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.

Abstract

Testbed 12 work evaluates the interoperability of the Common Map API tool with commercial vendor tools supporting GeoPackage. Ideally data can be shared and exchanged between apps on a single device via GeoPackage. The demonstration will show the vector and/or routing data being used by disparate applications.

Business Value

This Engineering Report (ER) demonstrates that an approach for sharing geospatial data between multiple applications on a single handheld device. Using this approach allows organizations to make more efficient use of the limited storage space available on these devices. Organizations may benefit through reduced hardware costs (less storage space required) or improved operations (higher quality data available to users).

What does this ER mean for the Working Group and OGC in general

One of the goals of GeoPackage is to provide a way to support multiple applications with a single dataset.

How does this ER relate to the work of the Working Group

If the experiment is successful, the ER will provide best practices and implementation guidance that will benefit members of the GeoPackage community.

Keywords

ogcdocs, testbed-12, GeoPackage, Common Map API, mobile

Chapter 1. Introduction

1.1. Scope

Testbed 12 is evaluating the interoperability of the Common Map API tool [1] with commercial vendor tools supporting GeoPackage. The goal is to understand how data can be exchanged between apps using GeoPackage as a shared memory space, potentially enriched with direct interprocess communication or shared memory as supported by the host operating system.

1.2. Document contributor contact points

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

Table 1. Contacts

Name	Organization
Jeff Yutzler (editor)	Image Matters LLC
Ziheng Sun	George Mason

1.3. Future Work

No future work is planned to this document.

1.4. Foreword

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.

Chapter 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.

- OGC: OGC 06-121r9, OGC® Web Services Common Standard (2010)

NOTE: This OWS Common Standard contains a list of normative references that are also applicable to this Implementation Standard.

- OGC: OGC 12-128r12, OGC® GeoPackage Encoding Standard (2015)

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r9] shall apply.

3.1. native library

A library built for a specific hardware platform and operating system.

3.2. Abbreviated terms

- APIApplication Program Interface
- CMAPI Common Map API
- GPKG GeoPackage
- NGA National Geospatial-Intelligence Agency
- SDK Software Development Kit

Chapter 4. Overview

The following table describes the sections that will appear later in this document.

Section Number	Description
5 - Application Requirements	Description of example scenario, status quo, and requirements for achieving the goals of the scenario
6 - Solutions	User Interface
7 - Recommendations	Feedback for the GeoPackage SWG
Appendix A	Revision History
Appendix B	Bibliography

Chapter 5. Application Requirements

The mobile application must support the viewing and editing of GeoPackages produced by different providers. The application must be designed in such a way that peer applications running on a single device have access to the same GeoPackage data. Additionally, the peers must respond appropriately to messages issued by each other. The user interface will use APIs such as Google Maps and OpenLayers that do not directly support GeoPackage.

5.1. Example Scenario

A GeoPackage resident on a mobile device contains network data. An application calculates routes based on the network. The user starts along the calculated route then decides to explore on foot. The GeoPackage contains feature data for the area surrounding the routing network (area of interest). The user displays this information in an application.

5.2. Status Quo

The authors are not aware of any prior tests of this approach.

5.3. Requirements Statement

The GeoPackage must contain National System for GEOINT (NSG) Application Schema feature and attribute content including aspects such as:

- Transportation (e.g., roads, railroads, and bridges);
- Hydrography (e.g., bodies of water, coastlines);
- Cultural (e.g., buildings, facilities, landmarks);
- Terrain (e.g., vegetation and soils); and
- Administrative areas and boundaries.

Chapter 6. Solutions

Three distinct challenges were identified and addressed in this work.

1. Common mapping APIs such as Google Maps and OpenLayers do not support GeoPackages or even GeoPackage constructs like Well-Known Binary (WKB).
2. Common mapping APIs do not have a direct way to communicate with each other.
3. JavaScript APIs are not suitable for processing large volumes of data.

6.1. Delivering GeoPackage Support Through the NGA JavaScript Library

Since Google Maps and OpenLayers do not support GeoPackage directly, there is need for an additional library to provide that support. In addition to read and write support, the library that should expose the WKB geometries and transform them into the GeoJSON and Well-Known Text (WKT) formats supported by Google Maps and OpenLayers.

The researchers considered the following alternatives:

- Native applications designed for the specific mobile operating system (iOS or Android)
- NGA’s JavaScript library [4]

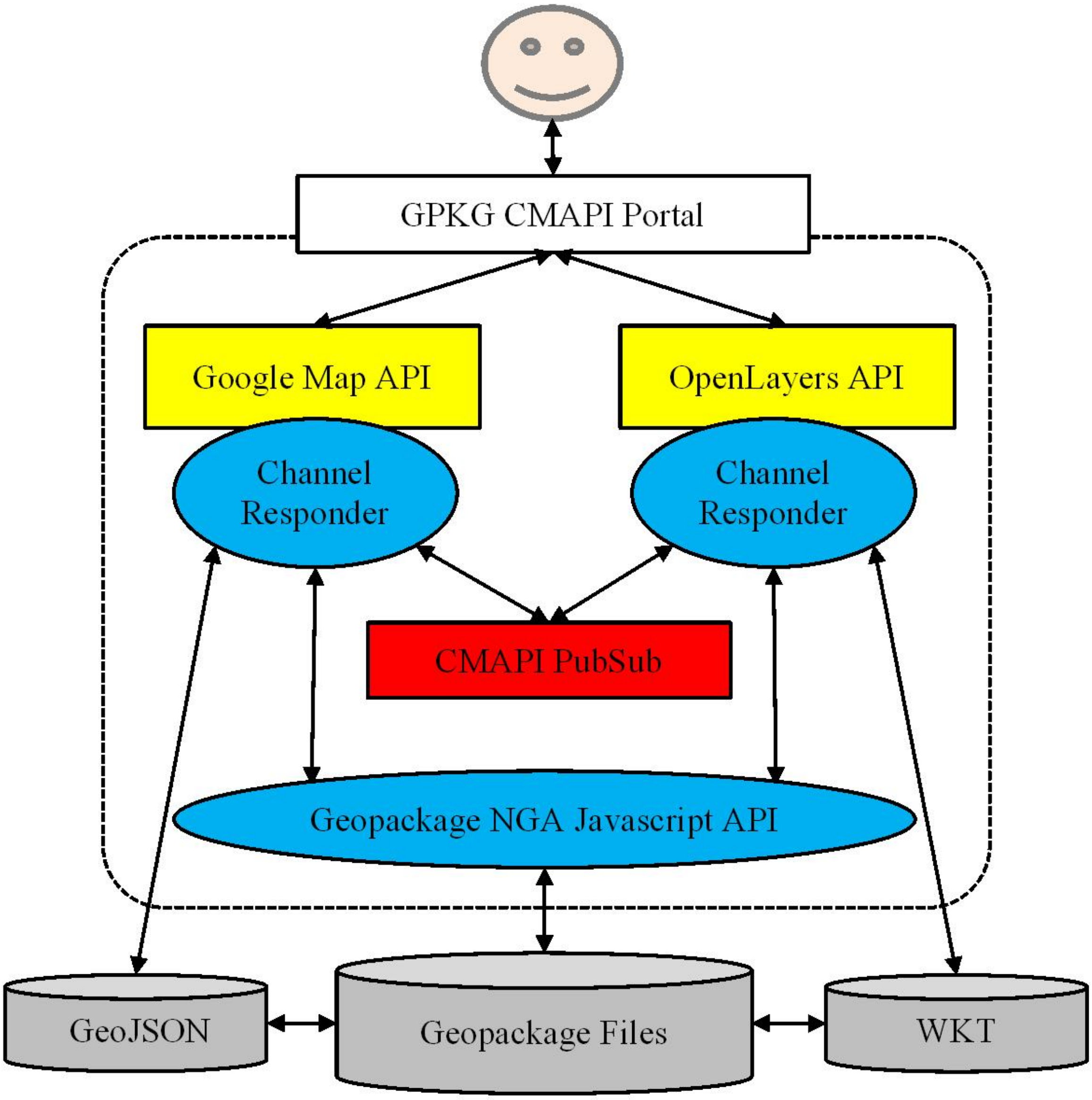
While a native application was compelling for many reasons, mainstream map APIs such as Google Maps and OpenLayers lack native SDKs. Since the most common map APIs are all written in JavaScript, it was decided to use JavaScript as the primary programming language for the apps. A comparison was made regarding the completeness and flexibility of most of the existing geopackage libraries, and the conclusion was that the NGA library was the best alternative available so this library was integrated with the Google Maps and OpenLayers-based applications.

6.2. Connecting Applications Together Through CMAPI and Channel Responders

There needs to be a way for front-end map APIs (Google Maps and OpenLayers) to communicate with our GeoPackage API. For example, if the mobile app needs to load a data layer from a GeoPackage file, a message composed by the app needs to be sent to the GeoPackage API and back.

The Common Map API (CMAPI) was the only reasonable alternative for performing this integration. CMAPI provides a standardized publish/subscribe model that allows different map APIs to interact with each other through a common messaging interface. Fortunately CMAPI has an open source JavaScript implementation [1].

To integrate the CMAPI with the map APIs, two sets of channel responders (one for Google Maps, one for OpenLayers) were developed. Following this example, a message containing the geometries contained in the data layer is published through the map.feature.plot channel. The channel responders receive messages, extract the relevant data, transform it to an acceptable format, and render it on the corresponding map using the map API. This architecture is illustrated in the figure below.

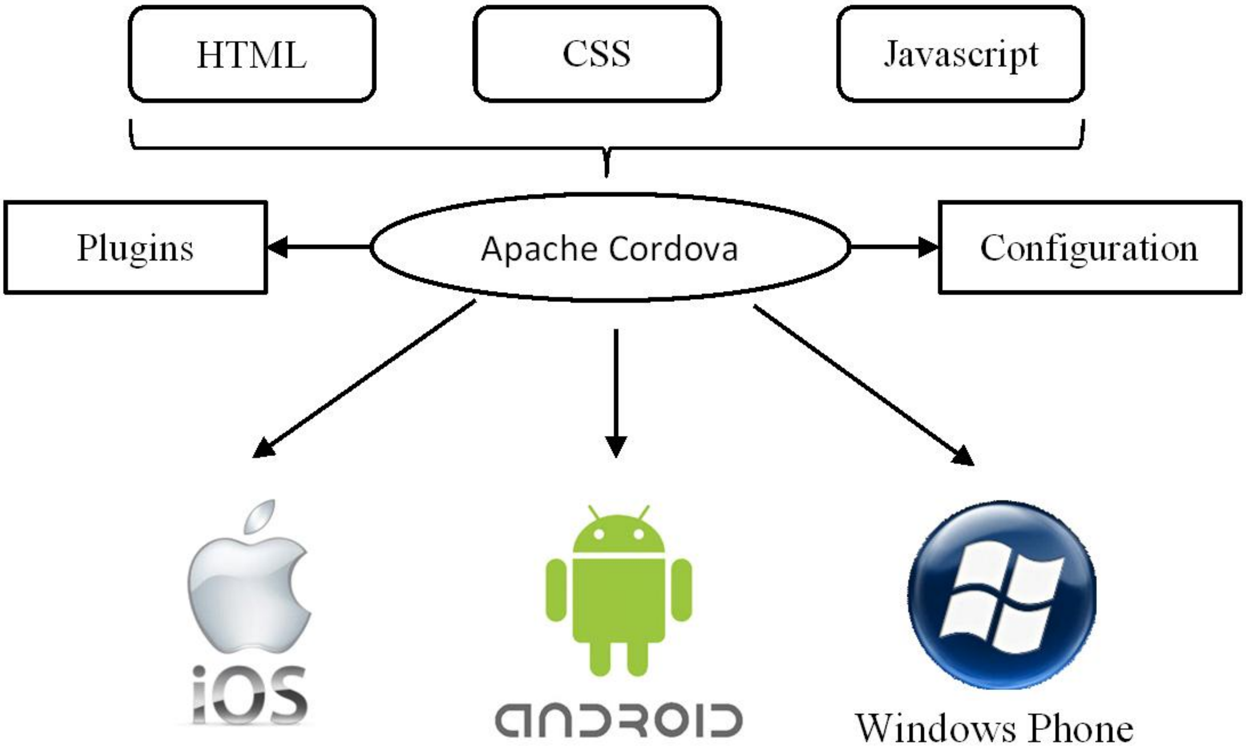


6.3. Interfacing with Native GeoPackage Libraries Through Apache Cordova

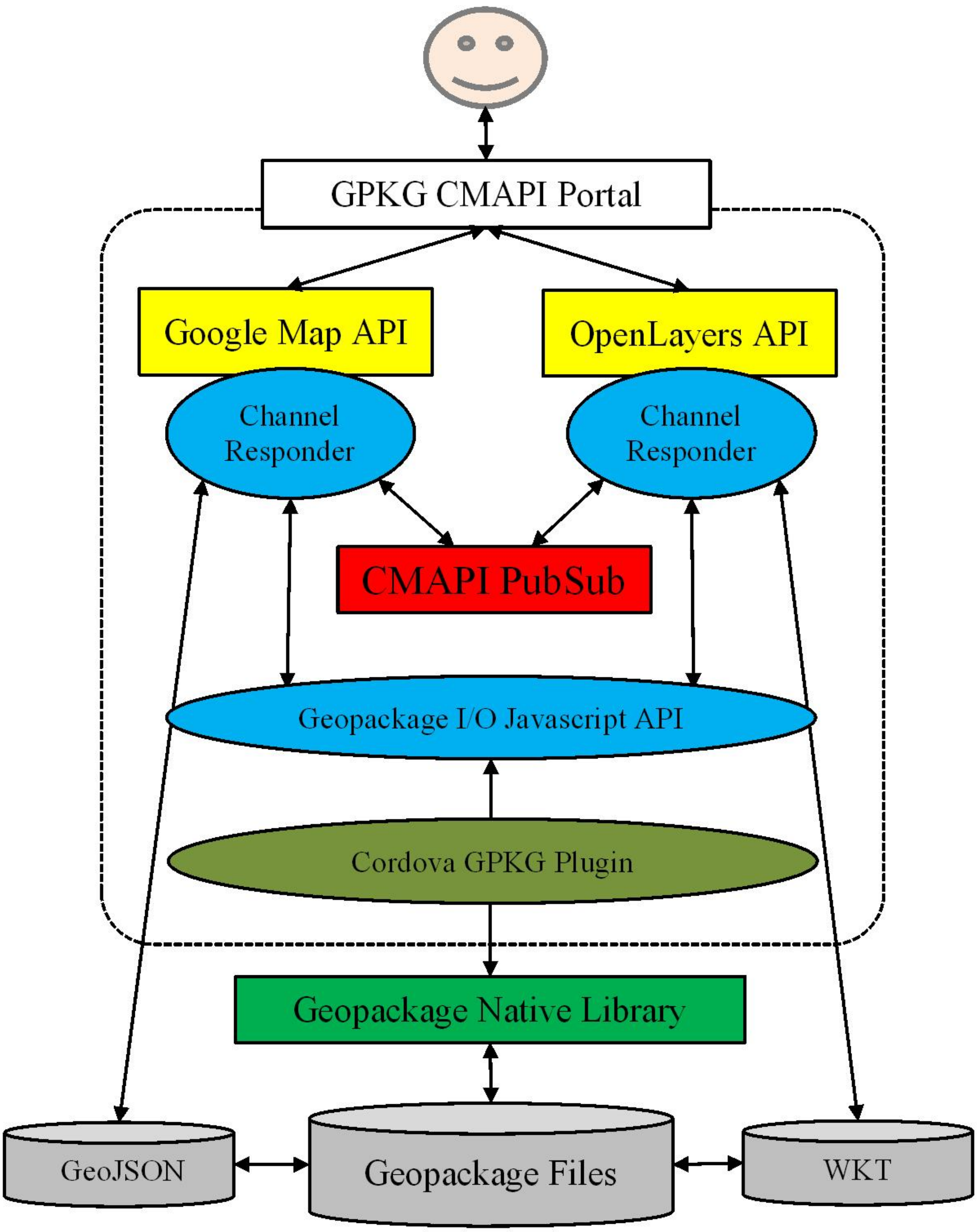
JavaScript was designed to be a lightweight programming language dedicated to processing small amounts of data. In large-scale application scenarios where the required date volume is at the gigabyte or terabyte level, JavaScript applications do not perform well.

To provide the required performance, it was determined that an option that uses native GeoPackage libraries should be provided. In addition to the JavaScript API, NGA has also developed libraries for iOS [2] and Android [3].

Integration with the native GeoPackage libraries was through Apache Cordova. Apache Cordova simplifies the effort required to implement mobile apps for various platforms by programing in HTML, CSS and JavaScript. Unlike web browsers, Cordova also provides direct access to the underlying hardware such as the camera and compass. In addition, it supports an extension plugin mechanism that allows us to use native libraries written in Objective-C, C, and Java.



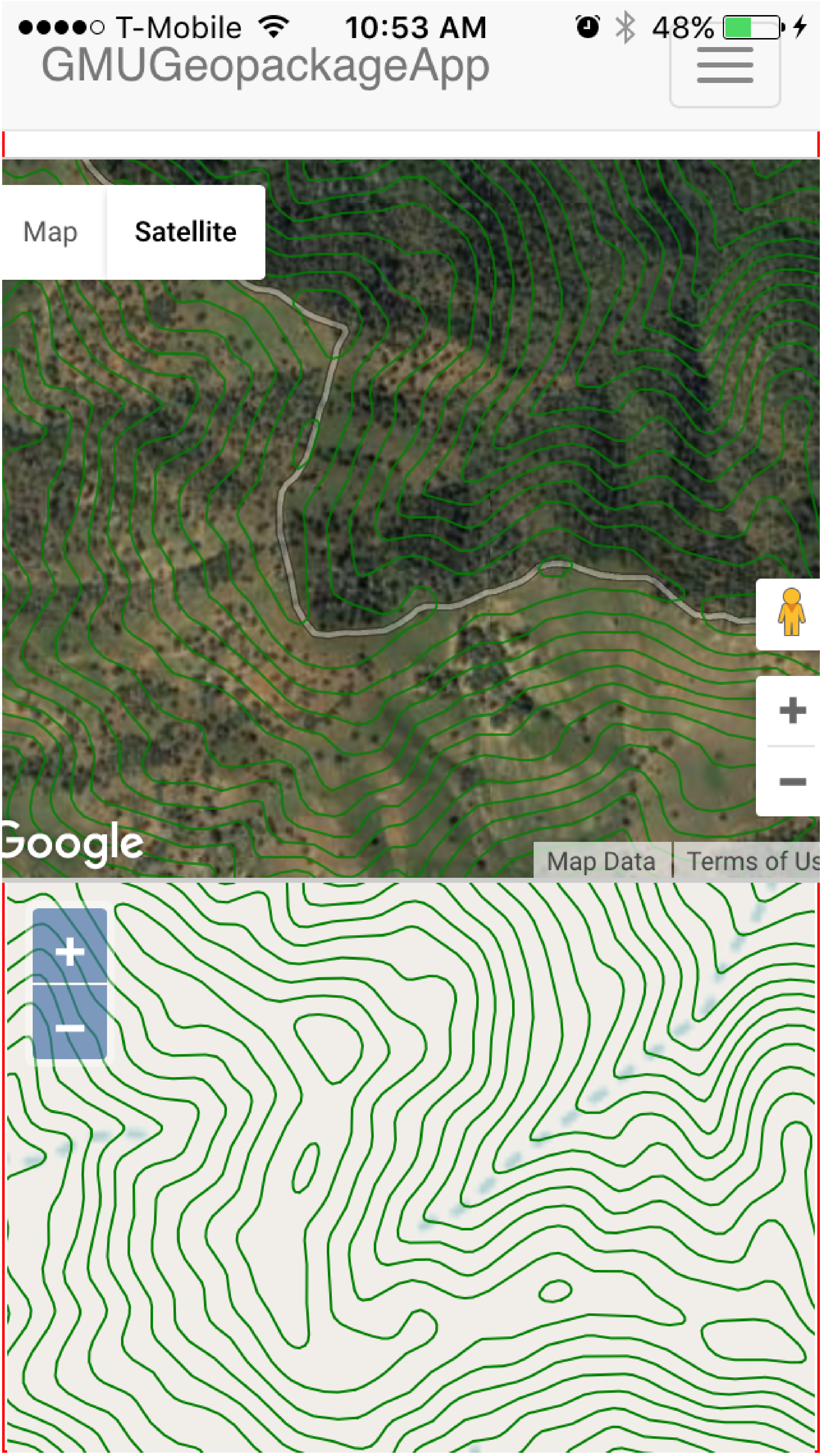
In this experiment, a plugin was developed for Cordova that allowed the use of NGA's native GeoPackage libraries. The plugin is composed of two matching parts, a JavaScript program and a native executable. The native executable invokes the native GeoPackage library while the JavaScript program provides an interface for other JavaScript programs to call. Otherwise this approach is the same as the pure-JavaScript approach described in the previous section. This architecture is illustrated in the diagram below.


GPKG CMAPI Portal
Google Map API
OpenLayers API
Channel Responder
Channel Responder
CMAPI PubSub
Geopackage I/O Javascript API
Cordova GPKG Plugin
Geopackage Native Library
GeoJSON
Geopackage Files
WKT

Chapter 7. Interface Design

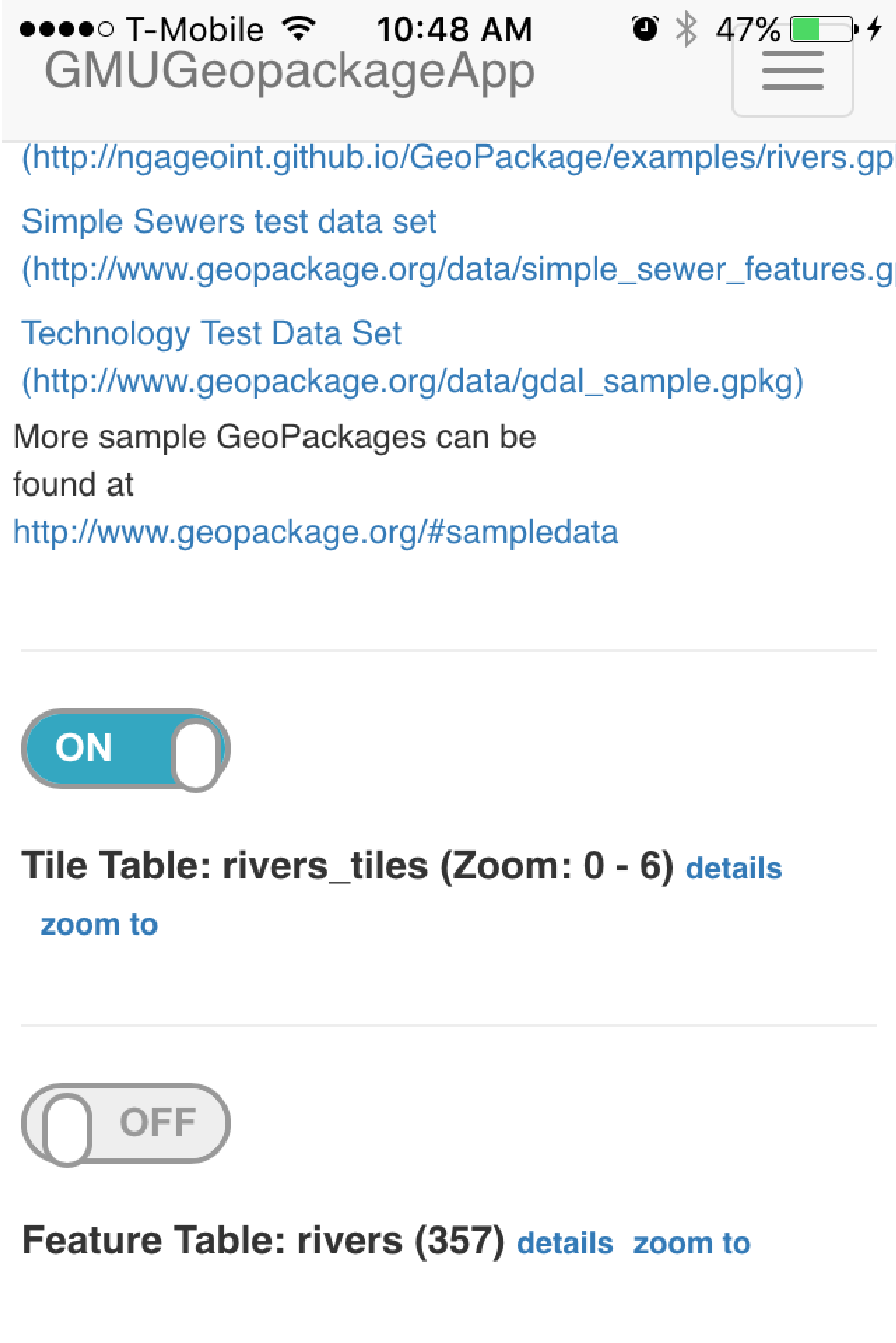
7.1. GMUGeopackageApp

7.1.1. iPhone Version

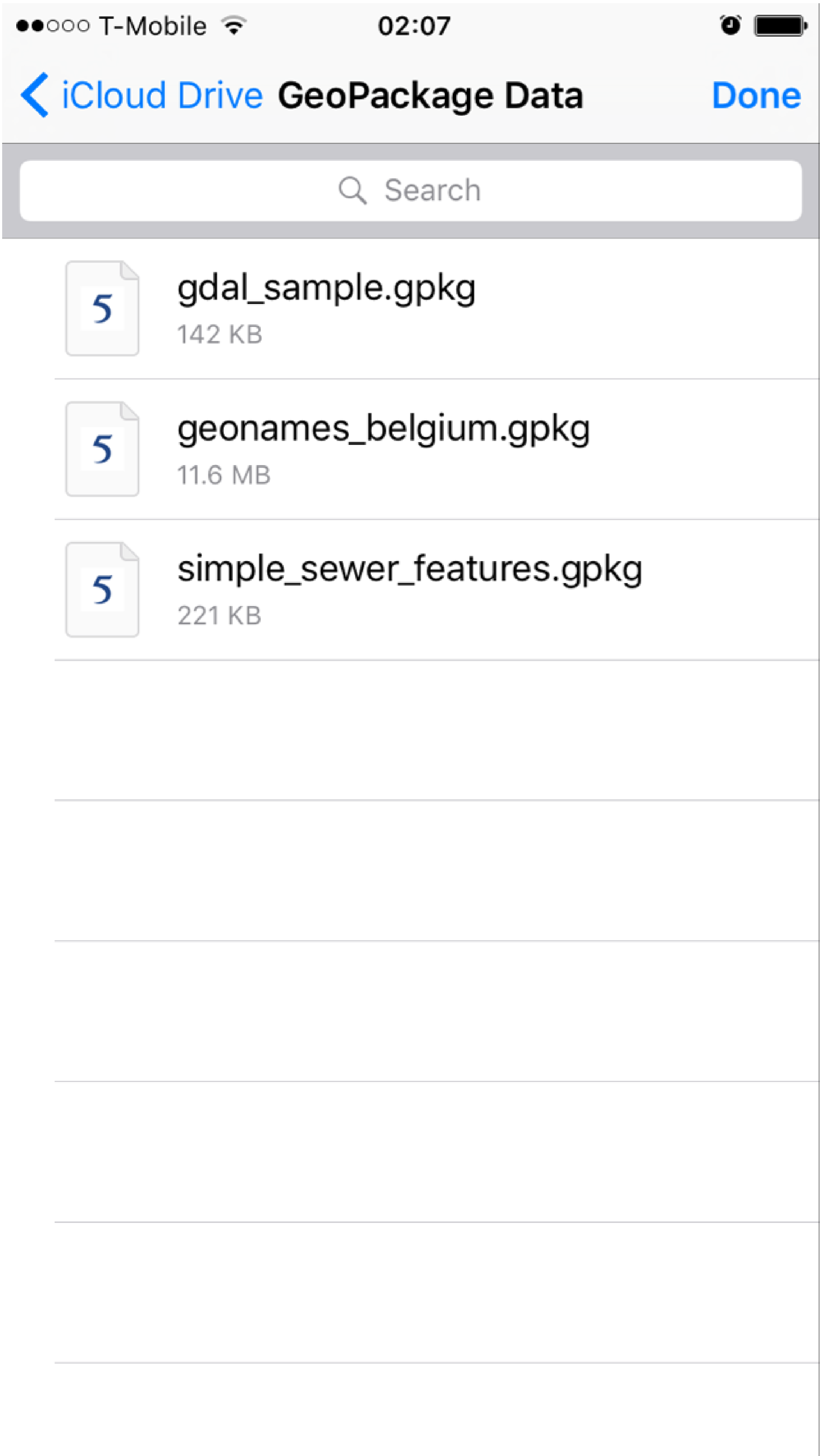


T-Mobile
10:53 AM
48%
GMUGeopackageApp
Map
Satellite
Google
Map Data
Terms of Us

7.1.2. Button Page


●●●●○ T-Mobile 10:48 AM 47%
GMUGeopackageApp
(http://ngageoint.github.io/GeoPackage/examples/rivers.gp
Simple Sewers test data set
(http://www.geopackage.org/data/simple_sewer_features.g
Technology Test Data Set
(http://www.geopackage.org/data/gdal_sample.gpkg)
More sample GeoPackages can be found at
http://www.geopackage.org/#sampledata
ON
Tile Table: rivers_tiles (Zoom: 0 - 6) details
zoom to
OFF
Feature Table: rivers (357) details zoom to

7.1.3. Apple Cloud Page


T-Mobile
02:07
iCloud Drive
GeoPackage Data
Done
Search
gdal_sample.gpkg
142 KB
geonames_belgium.gpkg
11.6 MB
simple_sewer_features.gpkg
221 KB

The user interface is composed of a menu bar and two map widgets. The menu bar contains buttons for browsing and opening GeoPackage files. In Apple iOS, the GeoPackage files in Apple Cloud can be retrieved and loaded into the app. The data layers in GeoPackage files are listed in the page associated with slide switches. Turning a switch on triggers the rendering of the corresponding data layer onto the two map widgets. The Google Maps widget shows a satellite base map while OpenLayers displays OpenStreetMap as base map. An elevation contour is overlaid on both maps so users can capture more information from various perspectives.

Chapter 8. Recommendations

Since these tests were successful, it is recommended that the GeoPackage SWG promotes the use of the approach and considers adopting it as a best practice. It may also be useful to investigate integration with web processing services that manage GeoPackage files.

Appendix A: Revision History

Table 2. Revision History

Date	Release	Editor	Primary clauses modified	Descriptions
June 15, 2016	J. Yutzler	.1	all	initial version
October 20, 2016	J. Yutzler	.2	all	comments integrated

Appendix B: Bibliography

[1] Common Map API, http://cmapi.org/

[2] NGA objective-C GeoPackage library, https://github.com/ngageoint/geopackage-ios

[3] NGA Java GeoPackage library, https://github.com/ngageoint/geopackage-android

[4] NGA JavaScript GeoPackage library, https://github.com/ngageoint/geopackage-js