OGC Testbed-15

Portrayal Summary ER

Table of Contents

Publication Date: 2020-02-07

Approval Date: 2019-11-22

Submission Date: 2019-10-31

Reference number of this document: OGC 19-019

Reference URL for this document: http://www.opengis.net/doc/PER/t15-D017

Category: OGC Public Engineering Report

Editor: Martin Klopfer

Title: OGC Testbed-15: Portrayal Summary ER

OGC Public Engineering Report

COPYRIGHT

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 Public 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.

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.

Chapter 1. Subject

This OGC Engineering Report provides an executive summary of the Open Portrayal Framework (OPF) Thread in OGC Testbed-15. The work in this testbed occurred between April and November 2019. Full details of the requirements, high-level architecture, and solutions are provided in the following Engineering Reports:

- OGC Testbed-15: Styles API Draft Specification Engineering Report [http://www.opengis.net/doc/PER/t15-D012]
- OGC Testbed-15: Encoding and Metadata Conceptual Model for Styles Engineering Report [http://www.opengis.net/doc/PER/t15-D011]
- OGC Testbed-15: Maps and Tiles API Draft Specification Engineering Report [http://www.opengis.net/doc/PER/t15-D014]
- OGC Testbed-15: Images and Change Sets Draft Specification Engineering Report [http://www.opengis.net/doc/PER/t15-D016]
- OGC Testbed-15: Open Portrayal Framework Engineering Report [http://www.opengis.net/doc/PER/t15-D015]

Chapter 2. Executive Summary

The OGC Open Portrayal Framework (OPF) is a set of emerging models and Application Programming Interface (API) specifications that support interoperable portrayal of heterogeneous geospatial data. The OPF facilitates the rendering of geospatial data in a uniform way, according to specific user requirements. The primary topics addressed in the OPF thread covered supporting style sharing and updates, client- and server-side rendering of both vector- and raster data, and converting styles from one encoding to another. This work was based on the concepts, relationships and terms defined in a draft conceptual style model. In addition, the requirement to render data according to style definitions in a scenario with denied, disrupted, intermittent, and limited bandwidth (DDIL) infrastructure was addressed.

2.1. Document contributor contact points

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

Contacts

Name	Organization	Role
Martin Klopfer	Frisia IT	Editor
Jeff Yutzler	Image Matters	Contributor
Joe Jagiella	Image Matters	Contributor
Clemens Portele	interactive instruments	Contributor
Keith Pomakis	CubeWerx Inc.	Contributor
Andrea Aime	GeoSolutions	Contributor
Stefano Bovio	GeoSolutions	Contributor
Jerome St-Louis	Ecere	Contributor
Joan Maso Pao	Universitat Autònoma de Barcelona (CREAF)	Contributor
Jeff Harrison	AGC	Contributor
Matt Sorenson	AGC	Contributor
Carl Reed	OGC	Contributor
Ingo Simonis	OGC	Contributor

2.2. 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 3. References

There are no normative references to content in this document.

Chapter 4. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

Style

a sequence of rules of symbolizing instructions to be applied by a rendering engine on one or more features and/or coverages

Style encoding

specification to express a style as one or more files

NOTE In Testbed-15 Mapbox Styles, OGC SLD versions 1.0 and 1.1 are used.

Icons

In computing, an icon is a pictogram or ideogram displayed on a computer screen in order to help the user navigate a computer system. https://en.wikipedia.org/wiki/Icon_(computing)

Layer

basic unit of geographic information that may be requested as a map from a server.

Portrayal

The ISO defines portrayal as presentation of information for human. See ISO 19117:2012, Geographic information - Portrayal. https://www.iso.org/standard/46226.html

Sprites

A sprite is a computer graphics term for a two-dimensional bitmap that is integrated into a larger scene.

Stylesheet

representation of a style in a style encoding

Style metadata

essential information about a style needed to support users to discover and select styles for rendering their data and for visual style editors to create user interfaces for editing a style

Coverages API

The draft OGC API - Coverages specification provides an API building block for accessing coverages as defined by the Coverage Implementation Schema (CIS) 1.1 on the Web.

Features API

The OGC API - Features - Part 1: Core standard provides API building blocks to create, modify and query features on the Web.

Maps API

The draft OGC API - Maps specification provides an API building blocks to describe, build and retrieve web maps.

Rendering engine

A rendering engine is an automated process that produces graphics using a pipeline of layers and styles as inputs. A rendering engine is commonly found in desktop or server-based geographic information systems.

Styles API

The draft OGC API - Styles specification is a Web API that enables map servers and clients as well as visual style editors to manage and fetch styles.

Tiles API

The draft OGC API - Tiles specification provides an API building block to describe, build and retrieve tiles from any resource that can be subdivided in a regular set of tiles (e.g., maps, features and coverages)

Web API

API using an architectural style that is founded on the technologies of the Web

4.1. Abbreviated terms

API

Application Programming Interface

OGC

Open Geospatial Consortium

SLD

OGC Styled Layer Descriptor

SE

OGC Symbology Encoding

Chapter 5. Open Portrayal Framework: High-Level Overview

This chapter provides an overview of the main requirements and achievements of the OGC Testbed-15 Open Portrayal Framework (OPF) Thread.

In the first section of this chapter the OPF scenario is briefly described based on the four major scenario aspects:

- Applying styles to data;
- Modifying and managing styles;
- Managing “changesets”;
- Addressing offline or DDIL situations.

The second section of this chapter describes how the work completed in the OPF thread set a milestone towards realizing a fully interoperable multi-source/multi-data type geospatial data rendering environment.

The third and final section of this chapter highlights the achievements of the testbed participants, demonstrating how the OPF scenario requirements were addressed.

5.1. OPF Scenario

Figure 1 illustrates the overall Testbed 15 OPF scenario. The scenario consists of four main parts. These four parts are further illustrated with screen captures from live online and video demonstrations below.

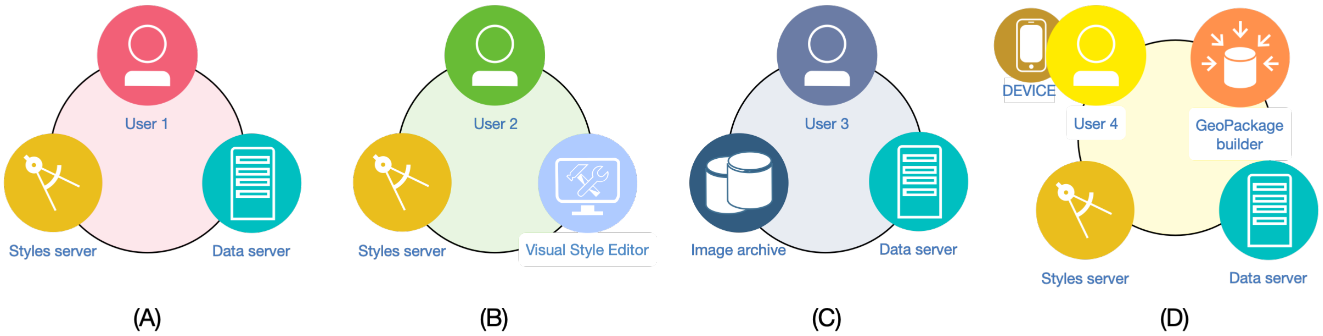


Figure 1. Open Portrayal Framework usage scenario

In OGC Testbed 15, participants assessed the ability of the OPF to support simulated users in a humanitarian relief situation. To accomplish this task, governmental and nongovernmental partners must understand the environment and infrastructure in the Daraa, Syria area - and quickly share that understanding with a variety of partners.

To promote this 'common picture', simple maps with styles for day or night operations must be rapidly customized and shared between partnering organizations from many nations. The most recent satellite imagery for the Daraa, Syria, area must also be added to the 'common picture', as illustrated below:

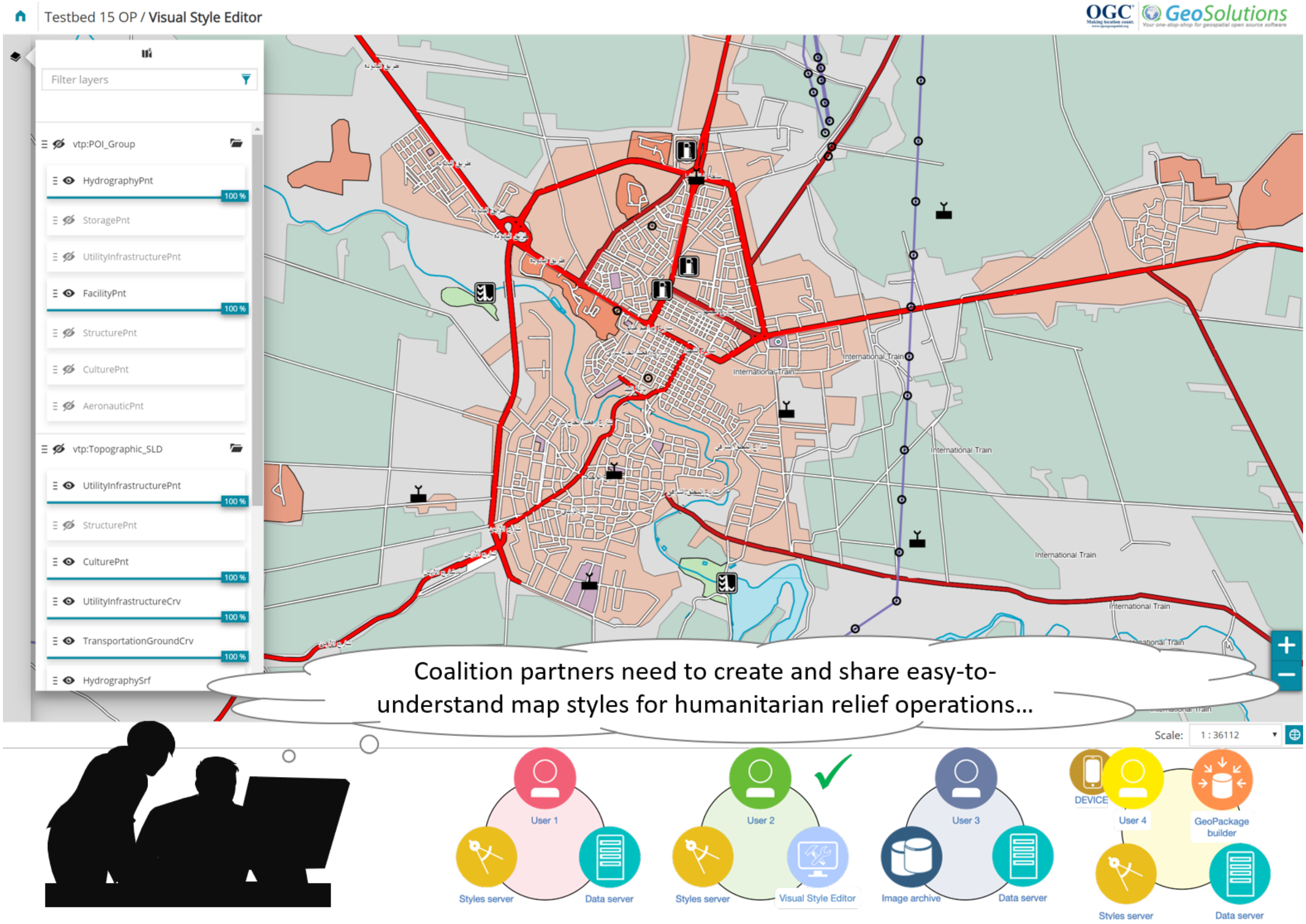


Figure 2. Humanitarian relief challenge - Need to interoperate with partners on different rendering environments

The following sections describe the components from Figure 1 and Figure 2:

- **Part (A)**: Users access geospatial data and explore corresponding styles that are available at a dedicated styles endpoint. The styles, following the styles conceptual model developed in this testbed [http://www.opengis.net/doc/PER/t15-D011], can use different models for the actual stylesheet definitions, such as OGC OGC Styled Layer Descriptor (SLD), Symbology Encoding (SE), Mapbox GL, Cascading Style Sheets (CSS), or others. There are bi-directional associations from data to styles and from styles to data to allow understanding which style can be used for which data set.

- **Part (B)**: Styles can be managed and modified by users with implementations of the *OGC API–Styles draft specification*. This draft API specification enhances reusability of styles by making them available to other users and services in a well-defined, standardized way. Style modification with *Visual Style Editors* was demonstrated in Testbed-15.

- **Part (C)**: The OPF supports updates to raw data repositories that can cause updates further downstream in the processing chain, such as for tile production, feature detection, and other applications. The OPF provides early support for these connections. In this Testbed, new satellite scenes caused re-generation of affected tiles. Using dedicated API mechanisms, clients were enabled to request only the modified/updated tiles.

- **Part (D)**: The OPF has the vision of enabling smooth online/offline transitions such as when users navigate from online to offline situations. The current OPF provides support for both situations, with styles, symbols, and other rendering specific aspects being available on both sides. For this purpose, Testbed-15 produced Web APIs for online usage and proposed extensions to the OGC GeoPackage standard for offline usage.

5.1.1. Part A: Applying Styles to Data

A Style is a sequence of symbolizing and rendering instructions that are to be applied by a rendering engine to one or more features and/or coverages. Styles in the OGC standards baseline have traditionally played two different roles:

1. As opaque identifiers in implementations of the Web Map Service (WMS) and Web Map Tile Service (WMTS) standards. This approach allows selection of different visual appearance for server side rendered map layers.
2. As Extensible Markup Language (XML) specifications such as the OGC Styled Layer Descriptor/Symbology Encoding Standard (SLD/SE) that served as an exchange format between systems, where styling rules are interpreted and applied to data.

These two approaches worked well in the past. More recently evolution for Web based mapping systems now requires more robust support for styling. In particular:

1. The emerging client-side styling capabilities require run-time access to a style served at a service endpoint. This requirement includes a way to locate a suitable style on the server side, determine the style's applicability to the current data layers, and retrieve the style.
2. As style complexity increases, the demand for re-using and sharing styles increases. Re-using often includes editing of styles that only serves as templates, rather than direct application of a specific style.
3. Style catalogs and style reuse require a way to describe styles (what kind of symbolization is used, what layers are involved, what attributes are needed).
4. Both client and server applications are increasingly supporting a wider variety of open styling encodings. Multiple style encodings can be made available either through a manual setup, or through automated conversion.
5. The Service Oriented Architecture is not dead and will remain in operation for quite a while. Yet, other architectural approaches that rely on Web APIs are also gaining popularity. This requires Web APIs that favor simplicity and ease of implementation. As such, clients can be enabled to locate, reason about, and retrieve styles with minimum information overhead.

In order to meet these requirements, two fundamental components had to be developed.

- First, a conceptual model for style encoding and metadata that provides information for understanding styles intended usage, availability, compatibility with existing data layers, as well as supporting style discovery.
- Second, a Web API for Styles to fetch and manage styles in a distributed environment.

The conceptual style model [http://www.opengis.net/doc/PER/t15-D011] is expressed in UML and builds around the central element *Style*. A *Style* has *StyleMetadata* to describe typical metadata aspects such as title, keywords, abstract, point of contact, version, or access constraints. Style metadata links to a thumbnail to allow quick style assessment. A *Style* can include any number of layers, and each layer is defined by a set of required attributes that the data needs to provide for proper styling. A *Stylesheet* describes the actual style encodings available.

The organization of the style model makes it complementary with the draft OGC Symbology Core standard (18-067r2). Overlap and duplication between the two models have been minimized to facilitate future convergence.

The Styles API is a Web API that enables map servers and clients as well as visual style editors to manage and fetch styles. This API is fully documented in OGC Testbed-15: Styles API Engineering Report (OGC 19-010) [http://www.opengis.net/doc/PER/t15-D012]. The Styles API is consistent with the emerging OGC API family of standards. The Styles API implements the conceptual model for style encodings and style metadata as documented in the OGC Testbed-15: Concept Model for Style Encoding & Metadata Model Engineering Report (OGC 19-023) [http://www.opengis.net/doc/PER/t15-D011].

Both the conceptual model and the Style API work well in conjunction with OGC API - Features and the emerging specifications OGC API - Coverages, -Tiles, and -Maps. With features, a collection can be rendered client-side using styles that the API might provide by reference, or a client might be looking for a suitable style for the layer on a shared style catalog. The OGC API - Coverages relation to style is similar to the Features API. The difference is that coverage rendering can involve a number of operations on the raster bands, including band selection, contrast stretching, color map application and hill shading. Client-side rendering of raw raster data, while possible, has not yet reached the same widespread support as the vector case. Using the emerging specification OGC API - Tiles, styles can be used to support server-side or client-side rendering. In the case of server side rendering, a tile can identify the applied style, which allows future modifications to that style if available at a Web endpoint with Styles API support. In the case of client-side rendering, the client can consume the style to retrieve rendering instructions. In conjunction with the emerging OGC API - Maps, the behavior is similar to the API Tiles with server-side rendering applied.

Interestingly enough, the conceptual styles model already supports tile containers that include both vector and raster data, though these containers are not supported by available tile encodings.

5.1.2. Part B: Modifying and Managing Styles

Styles can be managed and modified by users through the OGC API - Styles. The Styles API is a Web API that enables map servers and clients as well as visual style editors to manage and fetch styles. Visual style editors that create, update and delete styles for datasets that are shared by other Web APIs implementing the OGC API - Features - Part 1: Core standard or the draft OGC API - Coverages or draft OGC API - Tiles specifications.

Testbed-15 did some experiments with converting styles from one encoding to another. In summary, it can be stated that style encodings do not allow seamless conversion from one model to another.

5.1.3. Part C: Handling Changesets

One of the key requirements for the OPF activity was to reduce the transmission traffic caused by updates to tile stores. A tile store containing a pyramid of tiles with twenty zoom levels may include billions of tiles. Updating entire tile stores or tile caches would cause enormous traffic if all tiles are replaced, just because a small subset of the source data is to be updated. OPF participants therefore looked into the option to update only the subset of tiles where changes occurred, i.e. changesets.

Testbed-15 experimented with this situation assuming a tight connection between an image archive and the tile store. The use case implemented in Testbed 15 added a new satellite scene to the image archive, which then triggered updates to the connected tile store. The draft OGC Changesets API specification enables clients to retrieve only the tiles that have been affected by the new imagery. For sure, the number of affected tiles eventually depends on a number of aspects, such as the tiling strategy, or additional processing steps such as for feature extraction, interpolation, or generalization.

To achieve this functionality, the draft Changesets API is designed to operate with checkpoints. The Changesets API works with any data resource that supports a mechanism to request for server-side changes in a resource collection since a previous checkpoint.

In order to implement this capability, the service needs to keep an audit of every change in the resource(s) and assign a checkpoint identifier to each change. At some point in time, a client will request a resource collection and receive that collection with a checkpoint identifier. After a while, the client may send a request for updates and notify the server about the original checkpoint. The server will then inform the client about the changes in the resources up to a current checkpoint and the client can then update its copy of the resources.

In addition, the Changesets API supports further filtering on top of checkpoints. For example, this additional filtering allows the client or service to retrieve only updates that are flagged as important. This functionality was not tested extensively in Testbed-15.

5.1.4. Part D: Offline Usage

The offline use case complements the other components of the Open Portrayal Framework. A key goal of the OPF thread is to enable a seamless online/offline experience. That is, a user with full connectivity receives data and styling information from Web endpoints. The user then loses connectivity and switches transparently to a local copy of the data with corresponding styling information. Once connectivity is re-established, the user reconnects to Web endpoints while the offline data on the device gets updated in the background to be prepared for future offline situations again.

The OPF activity used the GeoPackage standard as the container solution for data, styles, and corresponding elements such as symbols or fonts. Please note that fonts were not addressed in Testbed-15, but are a potential subject for future GeoPackage research. The GeoPackage standard supports a flexible extension mechanism that worked well for the additional style data. However, the associations between data, styles, and symbols led to the development of a new construct called "Semantic Annotation". The Semantic Annotation concept allows for any type of associations to be added to a GeoPackage without changing the underlying table structure of existing tables. While adding some overhead, the advantages of the new solution quickly outperformed the added complexity.

5.2. OPF Implementation Scenario

The Open Portrayal Framework (OPF) set a milestone towards fully interoperable multi-source/multi-data type geospatial data rendering environments. The following figure illustrates the key elements of the framework from an implementation perspective.

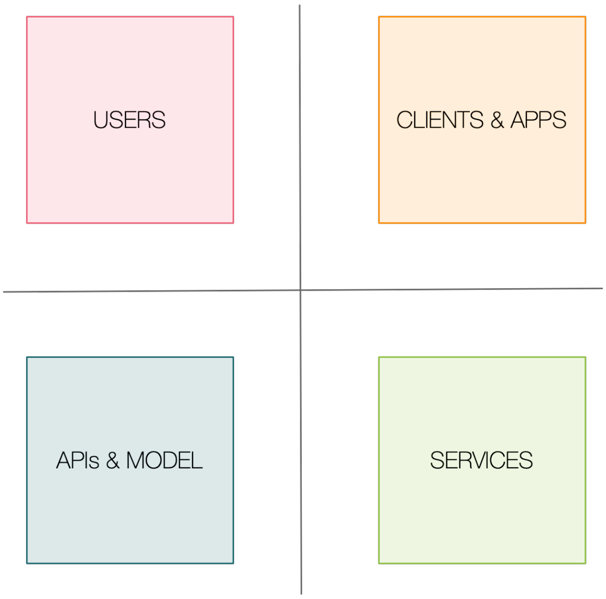


Figure 3. Abstract OPF components

The OPF is being designed to serve different types of users or clients. These include - but are not limited to - system integrators, decision makers, operational picture analysts etc. A user might be interested in receiving a map built from a number of different layers from a variety of sources, while another user is interested in optimizing the style definition for their community. In either case, users require an easy to use interface that allows any type of client software to interact with the OPF smoothly.

The different types of users can operate the OPF with any type of client, including web browser, mobile client, or dedicated desktop application.

The OPF is being designed to further enable integrating any type of data being served as Web interfaces using OGC APIs, including features, coverages, tiled data, maps, and images. At the same time, the OPF is being designed for continuous support for legacy systems that implement the OGC Web Service (OWS) standards, such as WMS, WMTS, Web Feature Service (WFS), or Web Coverage Service (WCS) or image archives front-ended with catalog interfaces.

The following figure illustrates all components that have been developed in Testbed-15 to explore a potential Open Portrayal Framework.

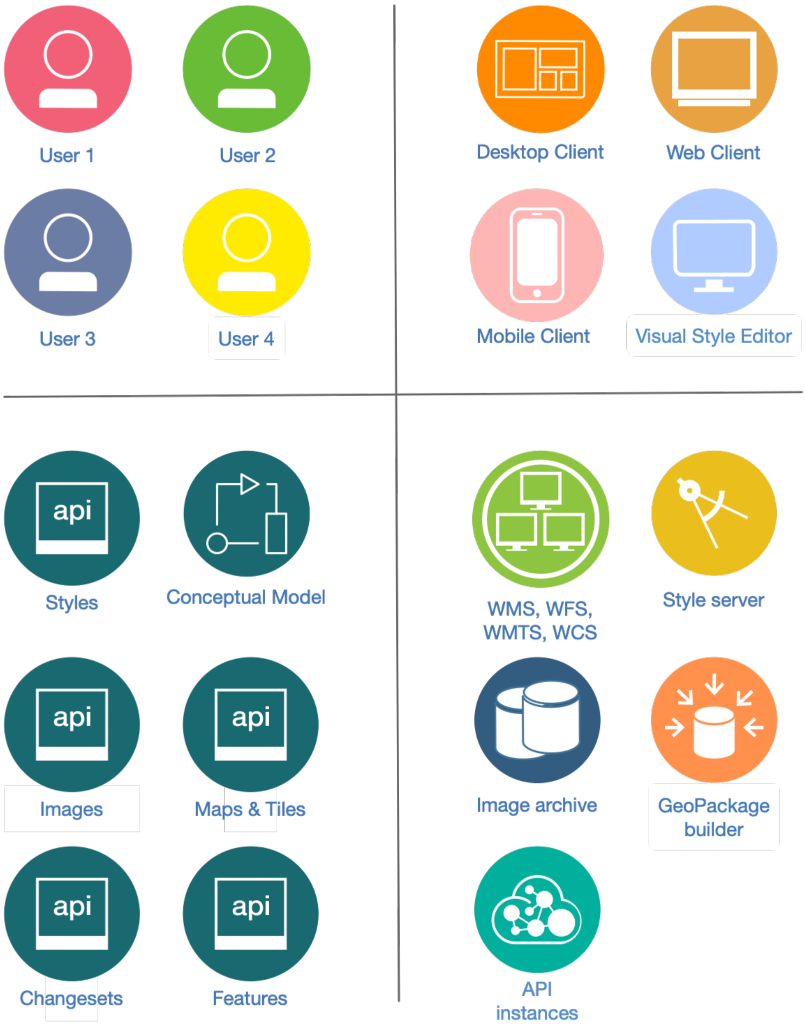


Figure 4. OPF components, with users in the upper left, client applications upper right, server technology lower right, and APIs lower left

- **Users**: Testbed-15 experimented with map consumers, style editors, image handlers, offline scenario users.
- **Clients**: Testbed-15 demonstrated a great variety of clients, including browser-based rendering clients for online and offline scenarios, and visual style editors.
- **Service endpoints**: The OPF was explored with both, modern Web API endpoints, as well as traditional WFS, WCS, and WMTS service instances. Style servers were implemented to test style management. An image archive and a GeoPackage builder complement the server instances.
- **APIs**: Testbed-15 implemented the conceptual styles model and all defined APIs to support different types of data and to experiment with both, server and client side rendering.

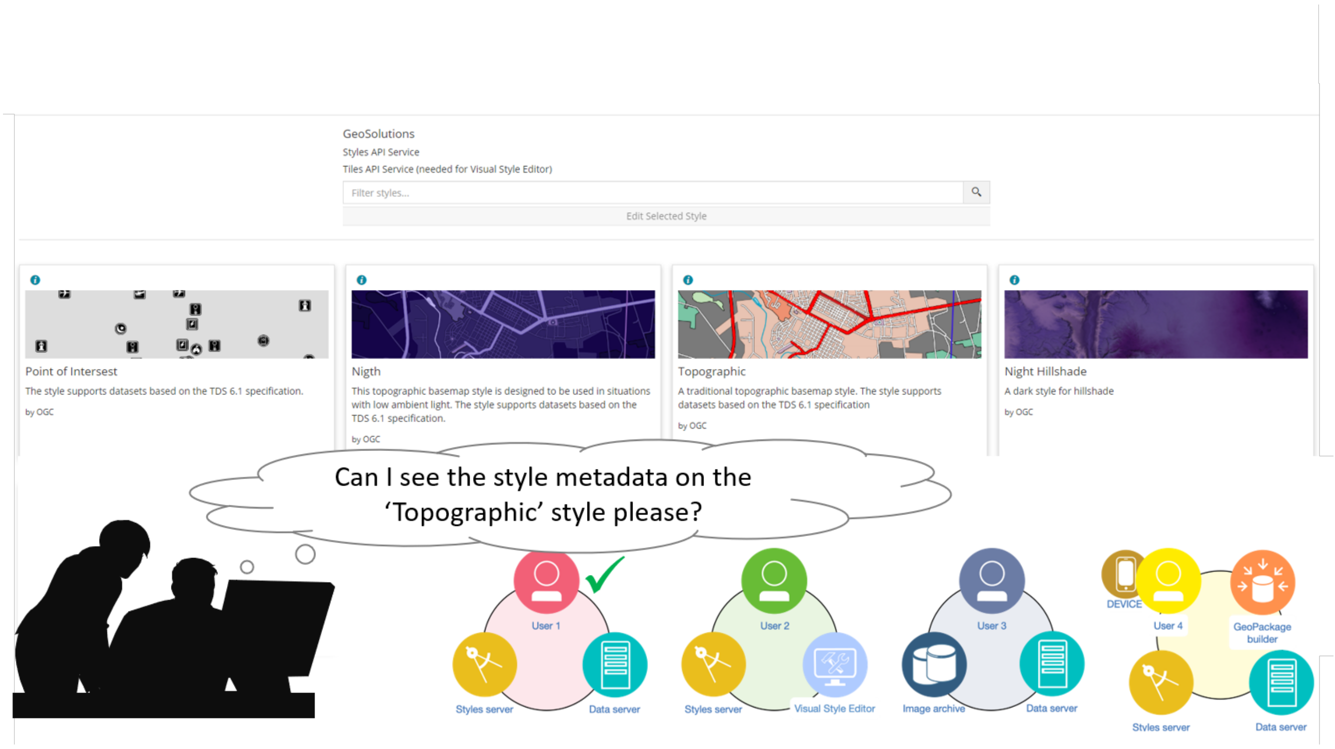
5.3. OPF Full Scenario Walk Through

This section highlights the achievements of the testbed participants, demonstrating how the OPF scenario requirements were addressed.

In OGC Testbed-15, participants tested the four parts of the scenario in a simulated humanitarian relief operation. In this operation governmental and non-governmental partners must understand the environment and infrastructure in the 'Daraa' area. The partners must also be able to quickly share that understanding between diverse mapping systems with different rendering environments.

To support this humanitarian relief operation, simple maps with styles for day and night operations must be rapidly customized and shared between partnering organizations. The most recent satellite imagery for Daraa must also be added to the 'common picture'.

The scenario begins with users accessing geospatial data and exploring corresponding styles and metadata that are made available at multiple style servers, maintained by members of the coalition. The styles, following the conceptual model developed in this testbed [http://www.opengis.net/doc/PER/t15-D011], can use different models for the actual stylesheet definitions, such as OGC SLD/SE, Mapbox GL, CSS, or others. Style discovery and style metadata review are shown in the following figures:



GeoSolutions

Styles API Service

Tiles API Service (needed for Visual Style Editor)

Filter styles...

Edit Selected Style

Point of Intersest

The style supports datasets based on the TDS 6.1 specification.

by OGC

Nigth

This topographic basemap style is designed to be used in situations with low ambient light. The style supports datasets based on the TDS 6.1 specification.

by OGC

Topographic

A traditional topographic basemap style. The style supports datasets based on the TDS 6.1 specification

by OGC

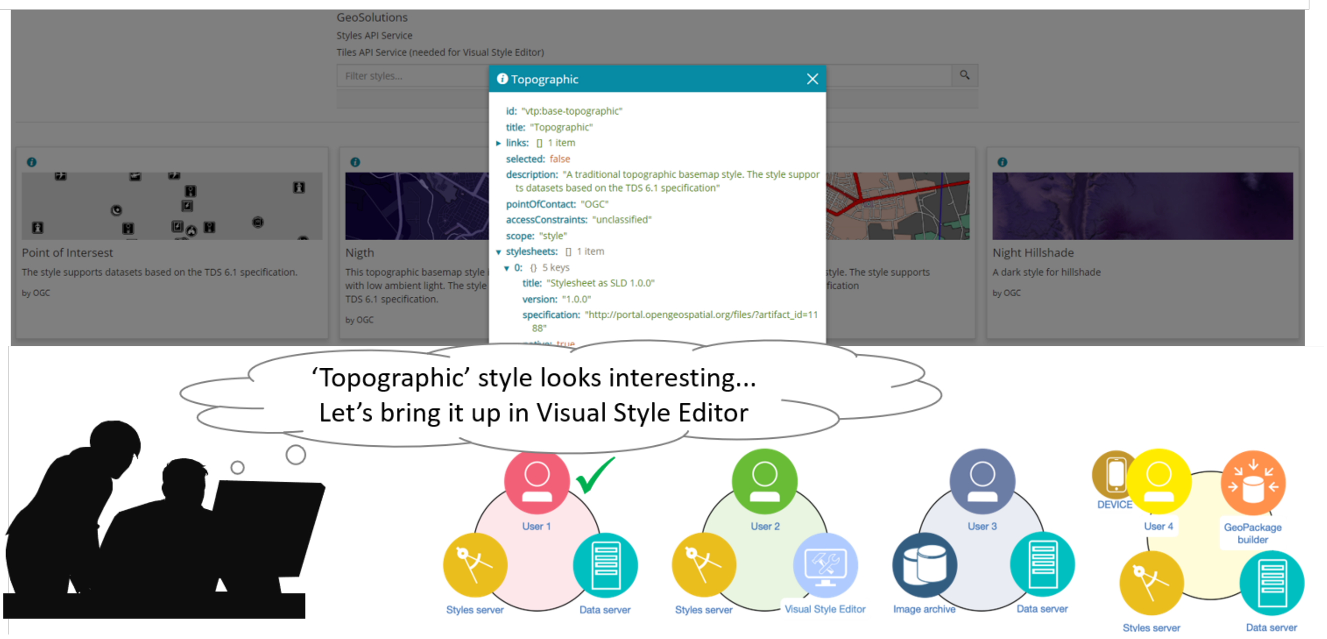
Night Hillshade

A dark style for hillshade

by OGC

Figure 5. Users access styles server to find the latest map styles they can reuse, update and share with other coalition partners

and



GeoSolutions
Styles API Service
Tiles API Service (needed for Visual Style Editor)

```
Topographic
  id: "vtp:base-topographic"
  title: "Topographic"
▸ links: [] 1 item
  selected: false
  description: "A traditional topographic basemap style. The style suppor
    ts datasets based on the TDS 6.1 specification"
  pointOfContact: "OGC"
  accessConstraints: "unclassified"
  scope: "style"
▾ stylesheets: [] 1 item
  ▾ 0: {} 5 keys
      title: "Stylesheet as SLD 1.0.0"
      version: "1.0.0"
      specification: "http://portal.opengeospatial.org/files/?artifact_id=11
        88"
```

Point of Intersest
The style supports datasets based on the TDS 6.1 specification.
by OGC

Nigth
This topographic basemap style with low ambient light. The style TDS 6.1 specification.
by OGC

style. The style supports fication

Night Hillshade
A dark style for hillshade
by OGC

Figure 6. Users review styles metadata for the 'Topographic' style shared on the styles server by other coalition partners

Styles and style metadata are managed and modified by users through the Web API *OGC API - Styles*. This draft specification enhances reusability of styles, which can be made available to others in a well-defined way.

Once users find a style that meets their purpose, they can customize that style with *Visual Style Editors*. In this scenario, they select the Topographic style and access it for update in a Visual Style Editor as shown in the following figures:

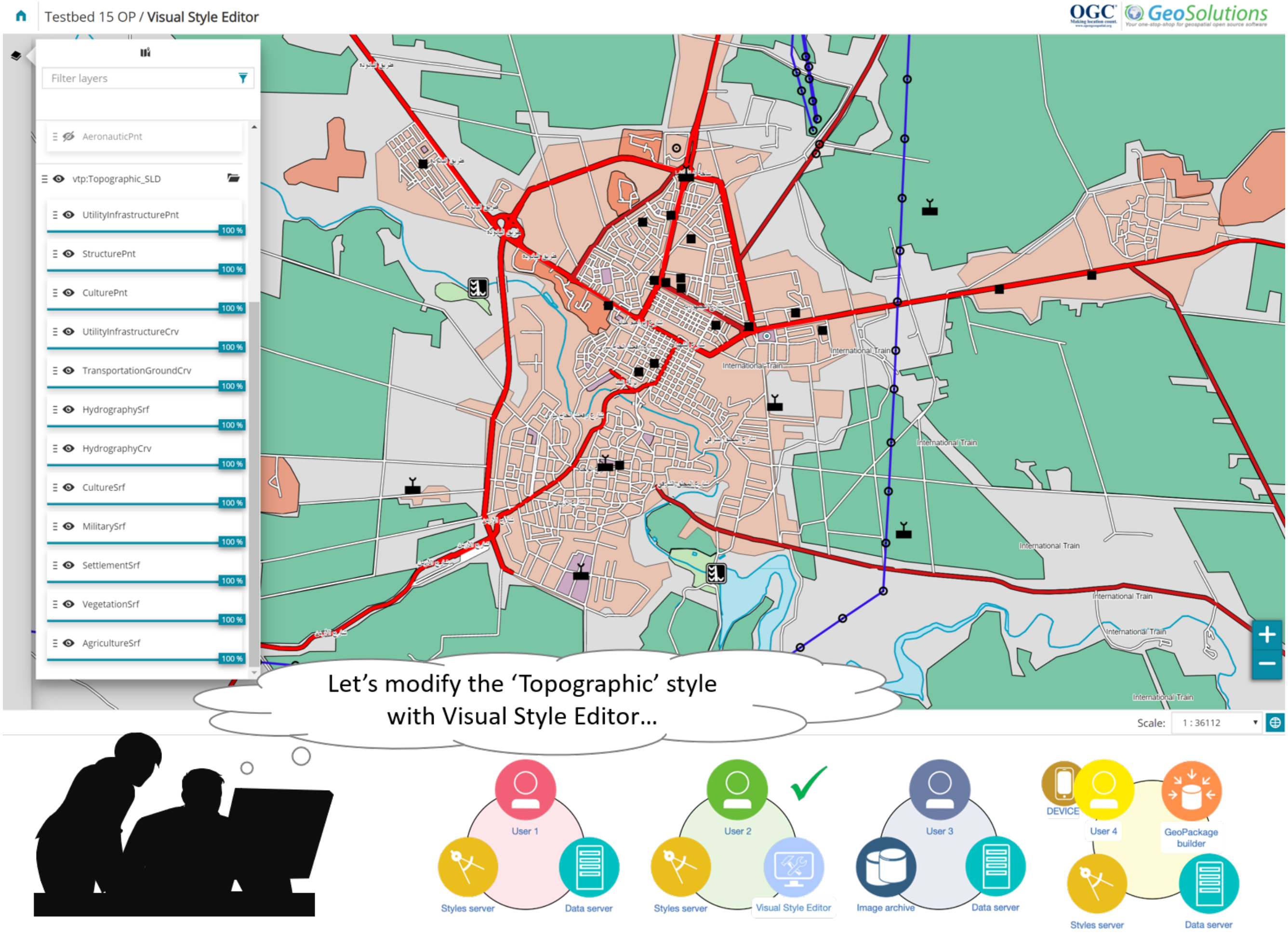


Figure 7. Users begin modifying Topographic style in Visual Style Editor.

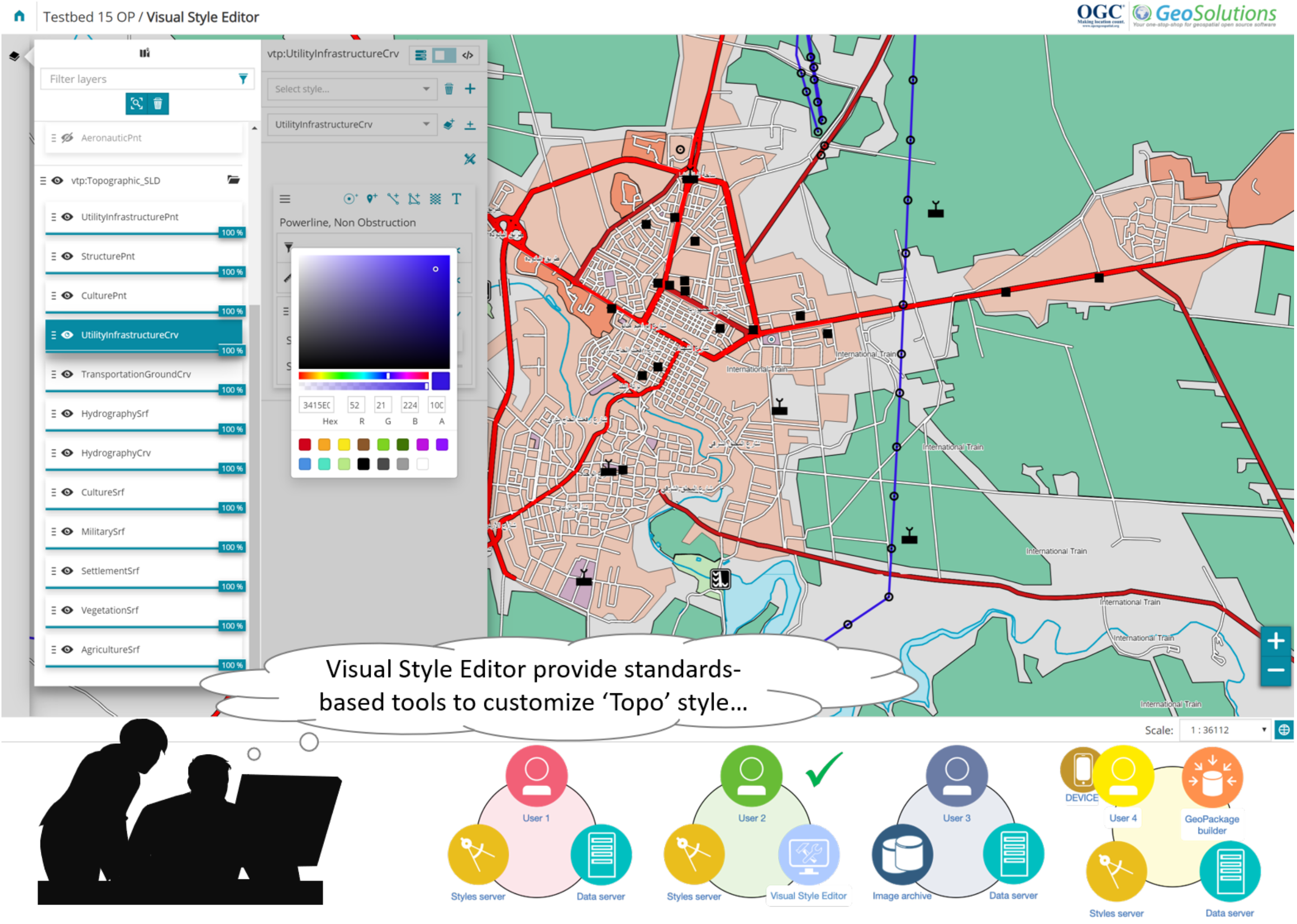


Figure 8. Visual symbolizers component include point, mark, line, polygon and text symbolizers.

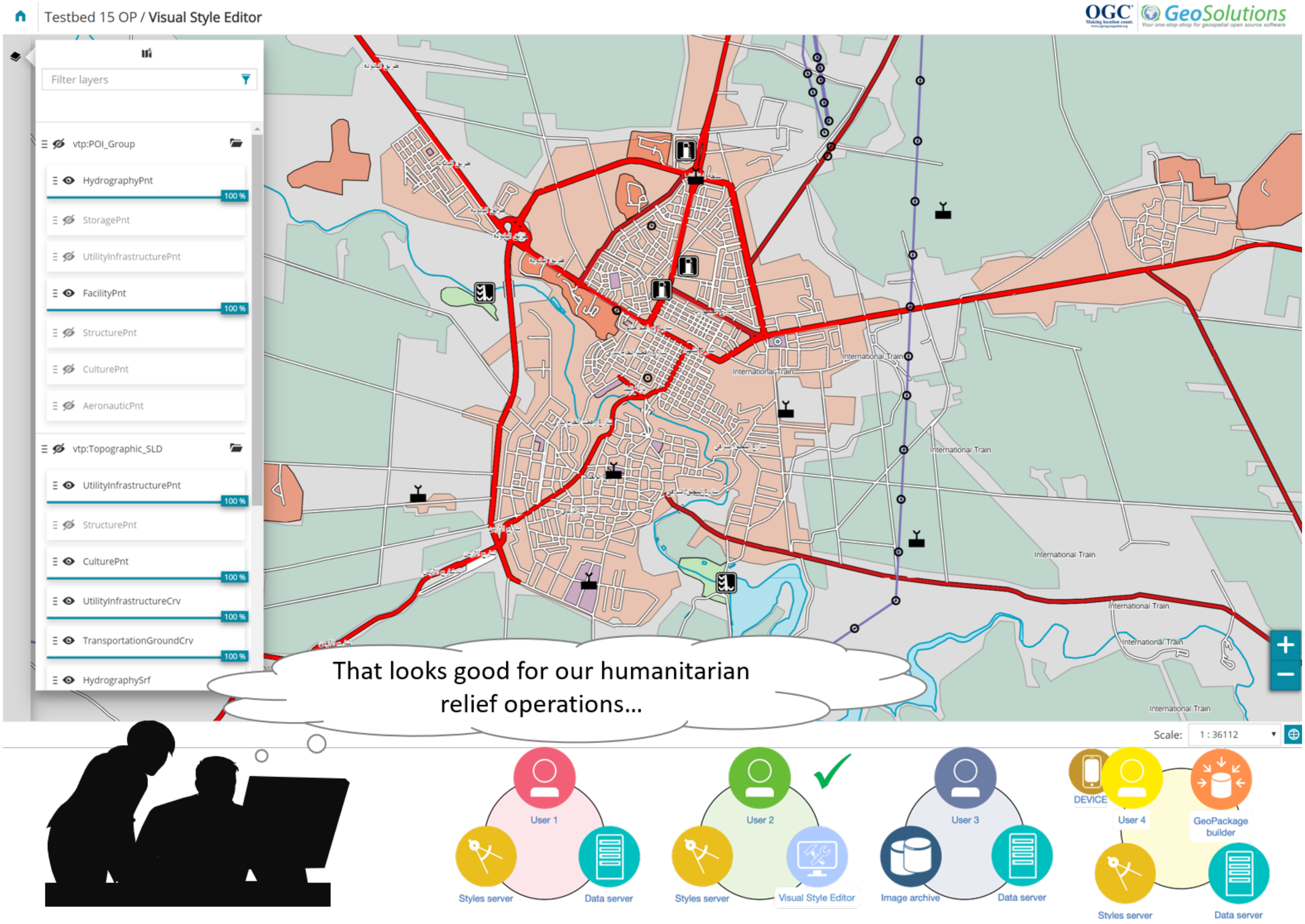
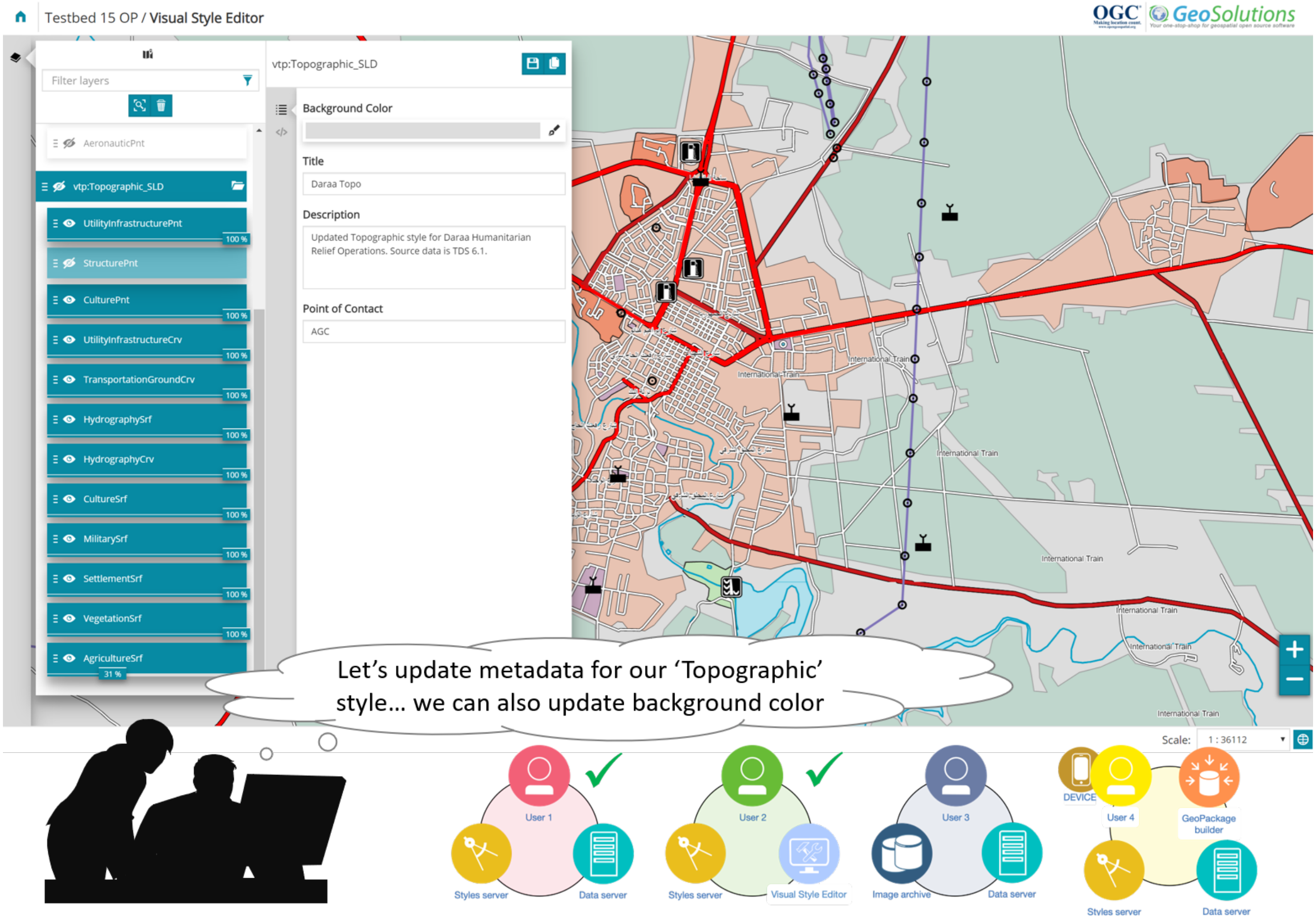


Figure 9. User create a new Topographic style for the operations.



Testbed 15 OP / Visual Style Editor

vtp:Topographic_SLD

Background Color

Title

Daraa Topo

Description

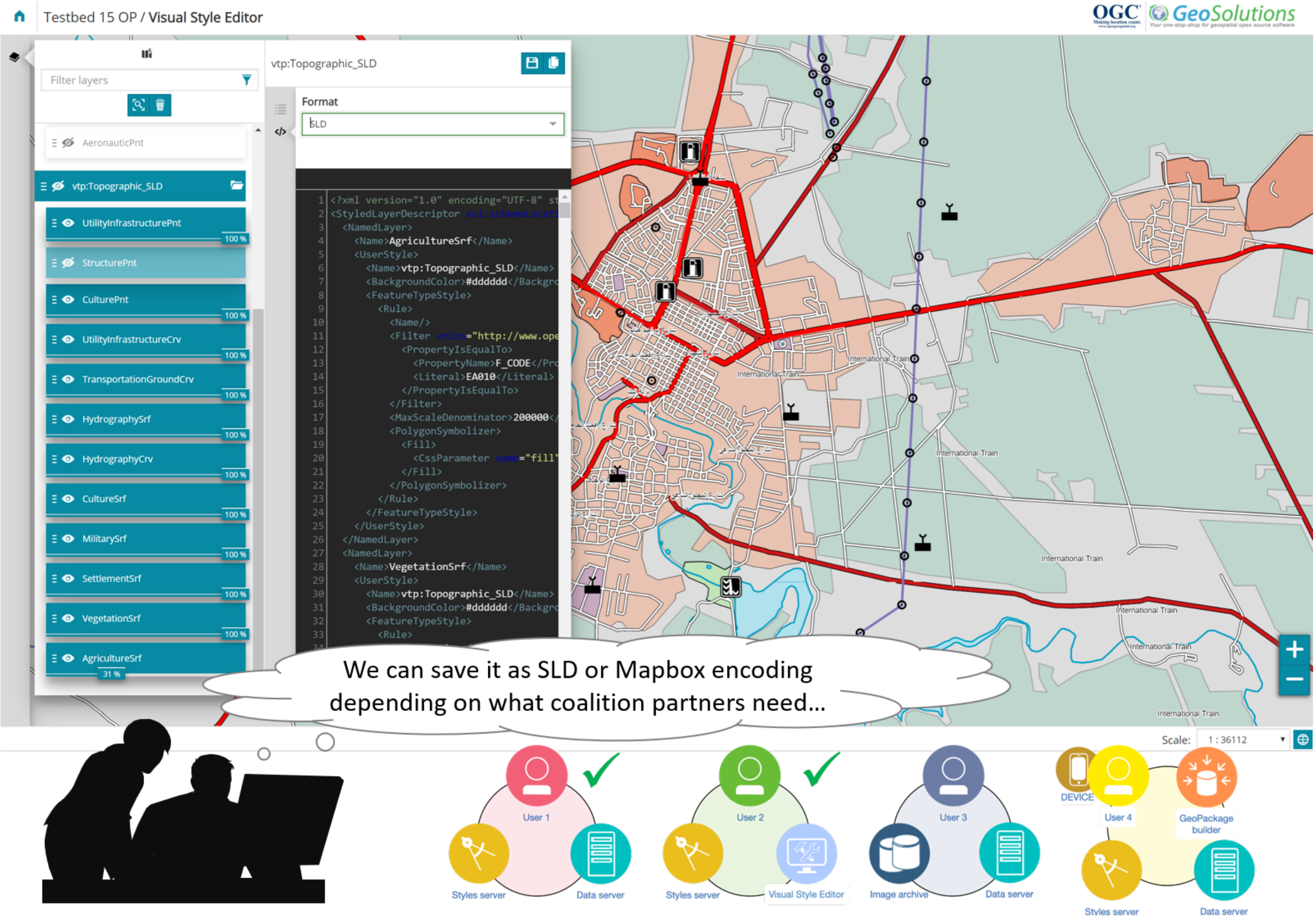
Updated Topographic style for Daraa Humanitarian Relief Operations. Source data is TDS 6.1.

Point of Contact

AGC

Let's update metadata for our 'Topographic' style... we can also update background color

Figure 10. User updating the style metadata, the background color picker is visible on the panel.



```
<?xml version="1.0" encoding="UTF-8" st
<StyledLayerDescriptor xsi:schemaLocati
  <NamedLayer>
    <Name>AgricultureSrf</Name>
    <UserStyle>
      <Name>vtp:Topographic_SLD</Name>
      <BackgroundColor>#dddddd</Backgro
      <FeatureTypeStyle>
        <Rule>
          <Name/>
          <Filter xmlns="http://www.ope
            <PropertyIsEqualTo>
              <PropertyName>F_CODE</Pro
              <Literal>EA010</Literal>
            </PropertyIsEqualTo>
          </Filter>
          <MaxScaleDenominator>200000</
          <PolygonSymbolizer>
            <Fill>
              <CssParameter name="fill"
            </Fill>
          </PolygonSymbolizer>
        </Rule>
      </FeatureTypeStyle>
    </UserStyle>
  </NamedLayer>
  <NamedLayer>
    <Name>VegetationSrf</Name>
    <UserStyle>
      <Name>vtp:Topographic_SLD</Name>
      <BackgroundColor>#dddddd</Backgro
      <FeatureTypeStyle>
        <Rule>
```

Figure 11. User saving style as SLD or Mapbox styles, depending on rendering environment needs of coalition partners.

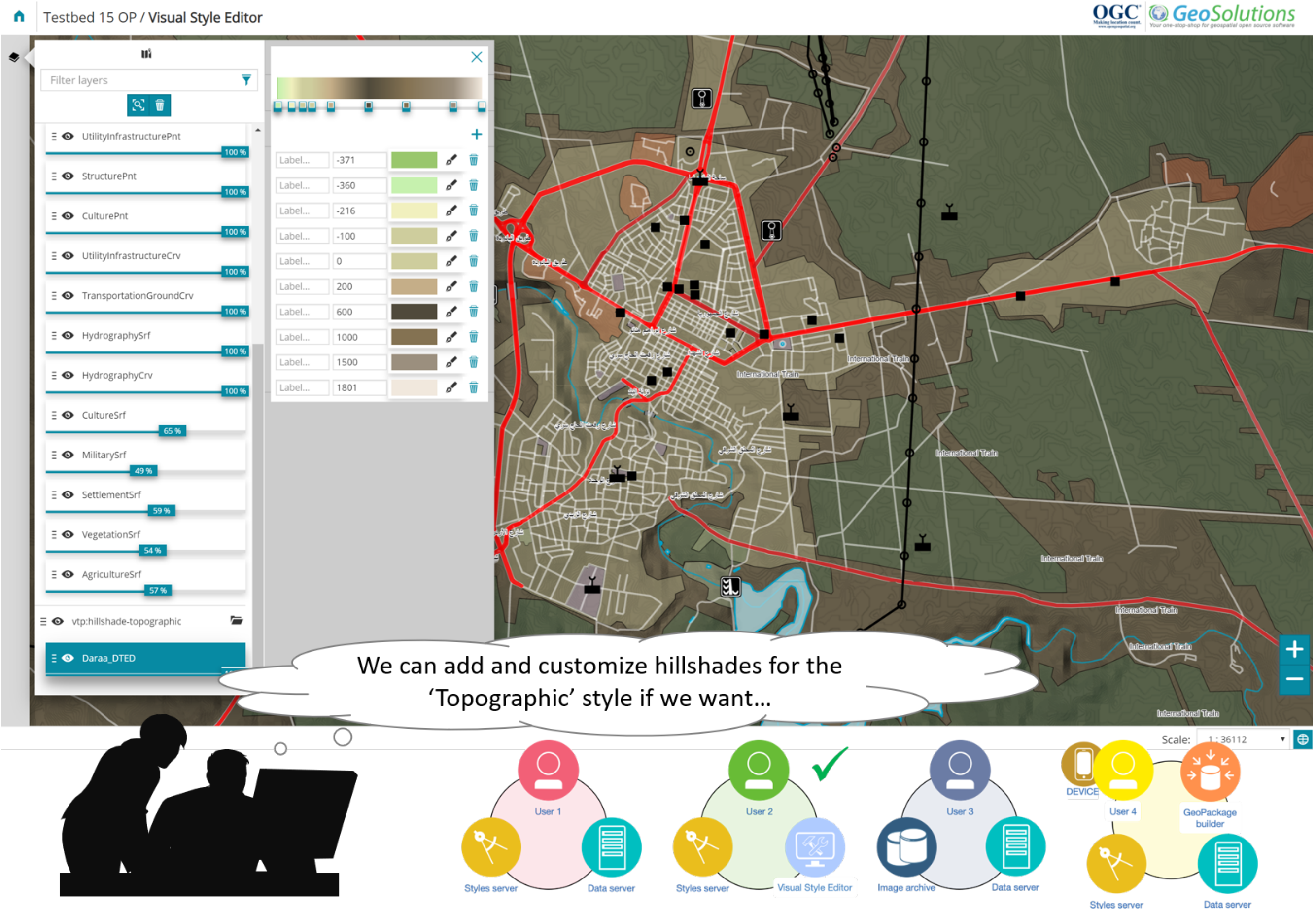


Figure 12. User can customize hillshade backgrounds, and edit them with raster symbolizer tools in the Visual Style Editor.

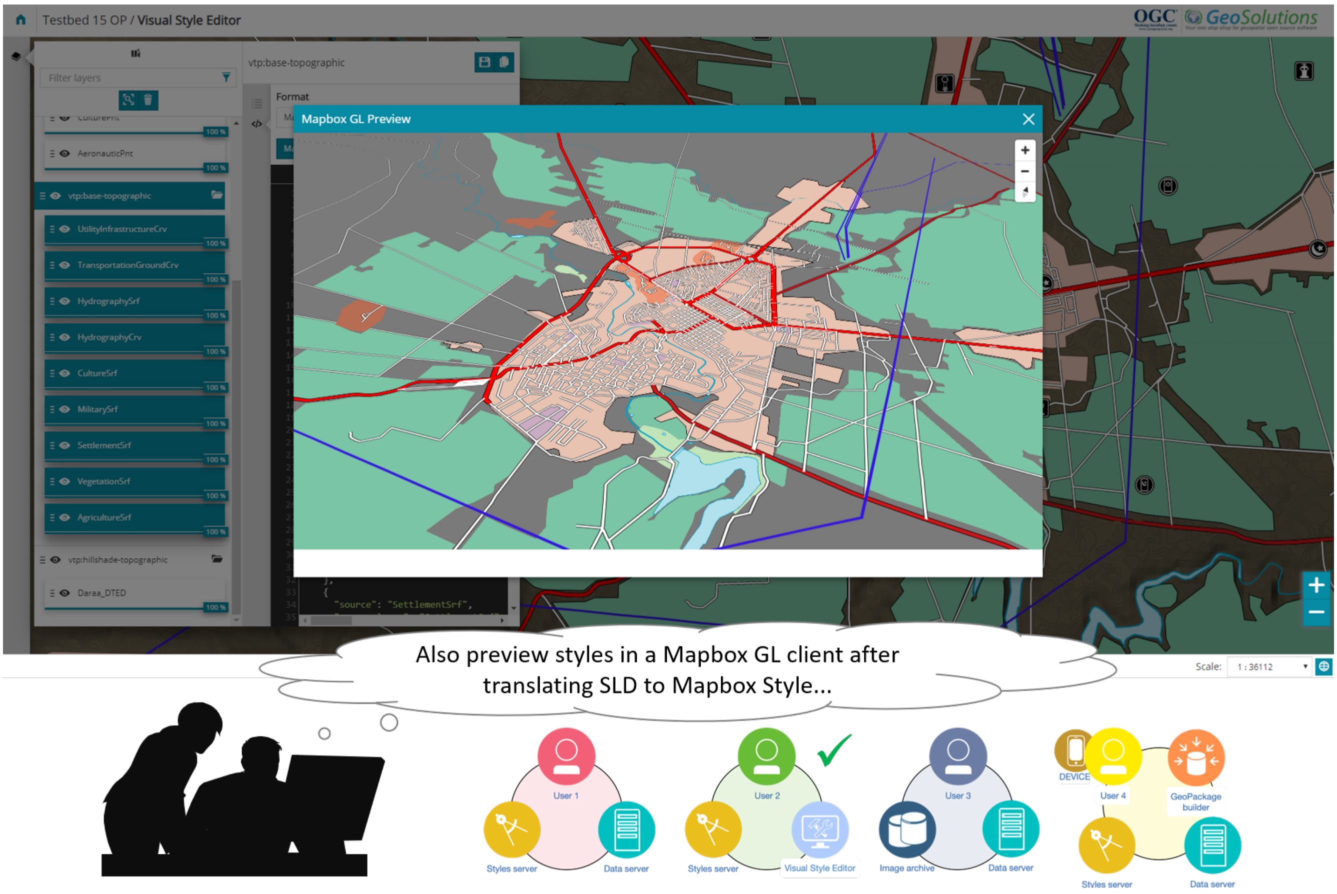


Figure 13. Preview of style in a Mapbox GL client after translating a SLD to a Mapbox Style.

5.4. Further Information and Videos

More details on the overall results of the OGC Testbed-15 project are available on the OGC IP Initiatives - Testbed 15 [https://www.opengeospatial.org/projects/initiatives/testbed15] website.

Demo videos on the OPF scenario and various implementations are available at the OGC YouTube Channel [https://www.youtube.com/playlist?list=PLQsQNjNIDU85HBDZWc8aE7EvQKE5nIedK].

Appendix A: Revision History

NOTE

Example History (Delete this note).

replace below entries as needed

Table 1. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
October 28, 2019	M. Klopfer	.1	all	initial version
October 31, 2019	M. Klopfer	1.0	all	final version