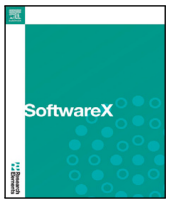
SoftwareX 28 (2024) 101942

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx



Original software publication

GIS-Publisher: Simplifying web-based GIS application development for enhanced data dissemination

Victor Lamas *, David de Castro, Alejandro Cortiñas, Miguel R. Luaces

Database Lab, University of a Coruña, A Coruña, Spain

ARTICLE INFO

Keywords:
Geographic information systems
Software product lines
Domain specific language
Automatic deployment

ABSTRACT

Geographic Information Systems (GIS) are complex systems that store, organize, process, and present geographically referenced data. Developing GIS requires specialized knowledge of algorithms, data structures, and geospatial concepts, along with the ability to implement scalable and efficient solutions for managing massive volumes of spatial data from various sources and providing user-friendly interfaces. This article introduces GIS-Publisher, a tool built using the Software Product Line (SPL) approach, which is a method of systematically creating a family of software products from shared core assets managing similarities and controlling variability. With GIS-Publisher, users without software development expertise can quickly and easily create web applications from directories containing *shapefiles*, a popular format for geographic data. The tool automates system deployment across various environments, including local computers, Secure Shell (SSH) remote servers, and Amazon Web Services (AWS) instances. Additionally, GIS-Publisher enables users to specify different styles using Styled Layer Descriptions (SLDs) for each shapefile, providing complete control over the visual representation of geographic data. This study details the features, benefits, and implementation of GIS-Publisher, demonstrating how it can accelerate GIS development and deployment.

Code metadata

Current code version	v1.0.3
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-24-00402
Permanent link to Reproducible Capsule	
Legal Code License	MIT
Code versioning system used	Git
Software code languages, tools, and services used	Javascript, Nodejs
Compilation requirements, operating environments & dependencies	Docker (optional, only required for local deployment)
If available Link to developer documentation/manual	https://github.com/lbdudc/gis-publisher#readme
Support email for questions	victor.lamas@udc.es

1. Motivation and significance

In recent years, many research fields have seen an increase in the production of geographic information. Disciplines such as environmental science, urban planning, public health, and disaster management rely heavily on spatial data to analyze patterns, make predictions, and inform decision-making [1,2]. Researchers and practitioners have many technologies, such as satellite imagery, remote sensing, Global Positioning System (GPS) devices, and mobile data collection apps, designed to collect geographic information efficiently. The challenge lies not in collecting geographic information but in its dissemination. Researchers and practitioners need and want to publish their findings to share with colleagues, stakeholders, and the broader public. Web-based Geographic Information Systems (GIS) offer an ideal solution for this need, providing interactive platforms where users can visualize, analyze, and share geographic data seamlessly, but building a web-based GIS is challenging because managing complex spatial data

* Corresponding author.
E-mail addresses: victor.lamas@udc.es (Victor Lamas), david.decastro@udc.es (David de Castro), alejandro.cortinas@udc.es (Alejandro Cortiñas), miguel.luaces@udc.es (Miguel R. Luaces).

https://doi.org/10.1016/j.softx.2024.101942
Received 30 July 2024; Received in revised form 14 October 2024; Accepted 16 October 2024
Available online 30 October 2024

across several components is frequently necessary [3]. One approach is to store geographic data in a spatial database (like PostGIS), serve it through a web map server (like GeoServer) on the backend, and visualize it on the front end using a web map library (like Leaflet). It takes knowledge of multiple technologies to develop this software and make these integrations. An easier approach would be to load spatial data from files using only a web map library on the front end. However, file-based data retrieval and processing can be slower and less scalable than a database-driven approach, which can cause issues with efficiency, particularly when working with large datasets.

Alternatives such as ArcGIS[1] and Carto[2] provide sophisticated functionalities like managing raster and image data, but they are pricey and require a monthly or annual subscription. Furthermore, some of these applications restrict the number of maps that can be made based on the subscription that the user chooses to pay. Even though our tool lacks certain features compared to these alternatives, our application provides an open-source, free alternative with an unlimited number of shared maps. Additionally, our tool gives the final user more control over how much they want to spend on having these maps running by making it simple to deploy the final systems on a machine to self-host the final maps.

A different option would be to use qgis2web,[3] an open-source QGIS plugin that enables users to export and create web maps on the client-side [4]. However, it is limited in handling large data files and does not support automatic cloud deployment, unlike our tool, which can handle large geographic files in addition to enabling cloud product deployment.

Recent advances in software engineering, such as SPLs, domain-specific languages, and model-driven development, offer promising avenues to simplify the process of building and deploying code. We propose a tool that generates web-based GIS applications from collections of geographic information files. This tool produces a complete software project that can be deployed to a local computer, an Secure Shell (SSH) remote server, or a cloud server. Additionally, the generated project serves as a starting point, facilitating the creation of more sophisticated and tailored GIS applications. Furthermore, our tool is built upon a software product line framework. This approach allows researchers to select functionalities from a feature model, which represents the information of all possible products of a SPL in terms of features and their relationships, tailoring the resulting product to specific requirements and preferences [5].

Our tool's value comes from its capacity to facilitate access to GIS technology, helping researchers and institutions to quickly develop and exchange geographic data without depending on sophisticated software or costly subscription services. Our tool promotes effective data publication and collaboration by facilitating a smooth deployment process on multiple server platforms, such as cloud instances and dedicated servers. This, consequently, aims to assist the scientific community in publishing results with greater efficiency and effectiveness.

2. Software description

2.1. Software architecture

GIS-Publisher is divided into separate components, each responsible for completing a specific task. The left part of Fig. 1 shows the GIS-Publisher main component and the artifacts it depends on: a folder with the geographic information, the configuration of the product to be generated, and the annotated code of the SPL. The right part of the figure shows the components responsible for the different steps in the process: **shapefile-reader** extracts metadata from the shapefiles in the folder with the geographic information, **gis-dsl** uses the metadata to create a product specification for the SPL, **spl-js-engine** creates the product from the product specification and the annotated code, **code-uploader** deploys the product, and *gis-publisher* uploads the shapefile data into the final deployed product. All components are implemented in JavaScript[4] with NodeJS.[5] We will now describe each component in more detail.

- **shapefile-reader**[6]: The *shapefile* format is the most common and widely accepted standard for handling geographic data in the GIS community. Each shapefile contains both geometric information and associated data attributes. The geometric data defines the shapes of geographic features, such as points, lines, or polygons, while the attribute data provides additional information about each feature, including names, IDs, and other relevant details. This component processes the shapefiles within a folder, extracting metadata about their data structure and geometry type. The main component of GIS-Publisher uses the metadata to specify the data model and the visualization model of the web-GIS product creating an instance of a domain-specific language (DSL) [6]. Users can define the visual appearance of geographic data layers by associating Styled Layer Descriptor (SLD) files with each shapefile using the same file names. This allows for customization of how geographic data is visually displayed.
- **gis-dsl**[7]: This component parses a product definition provided as an instance of the DSL, generating a JavaScript Object Notation (JSON) file usable by the software product line's derivation engine. Domain-specific languages bridge the gap between feature model-based configuration and general-purpose programming, maintaining clarity between problem and solution spaces [7]. A default feature set is pre-selected in the `config.json` configuration file including standard map-viewing functions such as layer management, zooming, panning, and map measurement. Users can modify this file to customize feature selection and thereby create a product with functionalities tailored to their specific needs selected from those provided by the SPL.
- **spl-js-engine**[8]: The products are generated by the tool using a SPL derivation engine called *spl-js-engine* [8]. This component takes the product specification created from the SPl, and the annotated code of the SPL, and it generates a finished product using the annotative approach. The annotated code of the SPL can be accessed as publicly available.[9]
- **code-uploader**[10]: This component deploys the final product into a local computer, a remote computer accessible via SSH, or an Amazon Web Service (AWS) instance. In addition to managing pre-machine setup tasks, it uses Docker to launch the product when required. Regardless of the deployment destination, users may specify the configuration in the `config.json` file (e.g., host, port, the authentication certificate, etc.). Depending on the destination, the component performs additional tasks:
  - **Local Deployment**: The tool deploys the final product on *localhost* for rapid setup and testing. This feature is beneficial for testing and debugging generated products before deployment to ensure they meet user requirements. Users only need Docker and docker-compose installed on their computers. The component starts docker containers on the machine after finishing the product generation.

[1] https://arcgis.com/
[2] https://carto.com/
[3] https://plugins.qgis.org/plugins/qgis2web/
[4] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[5] https://nodejs.org/en
[6] https://github.com/lbdudc/shapefile-reader
[7] https://github.com/lbdudc/gis-dsl
[8] https://github.com/AlexCortinas/spl-js-engine
[9] https://github.com/lbdudc/mini-lps
[10] https://github.com/lbdudc/code-uploader

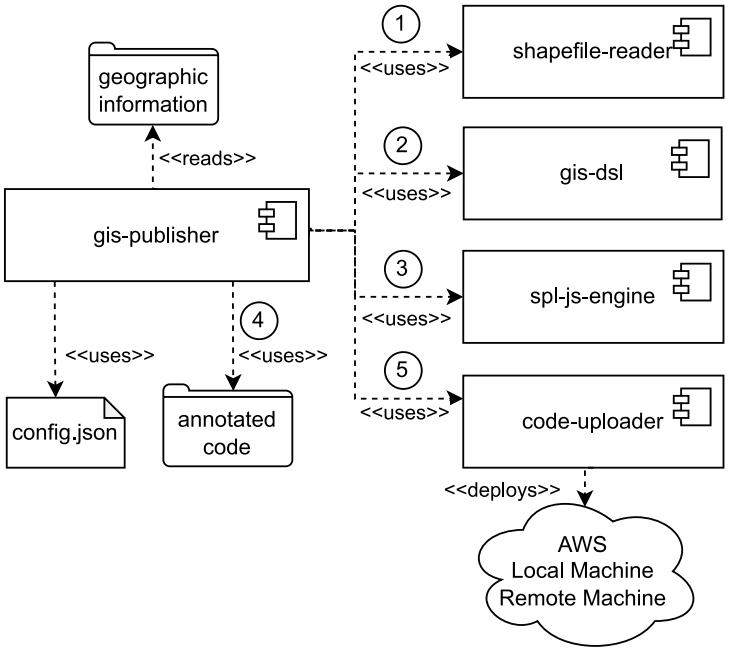


Fig. 1. Component diagram of the tool architecture.

- **SSH Deployment**: Users can install final products via SSH on a remote computer. The tool ensures all installation requirements are met and automatically sets up Docker and docker-compose if needed. It handles source code transfer and starts Docker containers on the remote machine.
- **AWS Deployment**: This is handled as an SSH deployment, but the tool first creates an AWS instance for cloud-based deployment.

2.2. Software features

In this section, we describe some of the most relevant features of the tool. We will first describe the features of GIS-Publisher, and then we will describe the features of the products that it generates.

2.2.1. GIS-Publisher features

GIS-Publisher is designed to create, customize, and deploy web-based GIS applications starting from a folder of geographic information. It is a command-line utility that offers flexibility by allowing users to select specific features for inclusion and generates comprehensive, customizable products tailored to meet diverse project needs. Thus, we believe it can be used by researchers who generate geographic information to easily create applications to share their information with other stakeholders.

Each of the components that are used by GIS-Publisher can also be used on their own:

- **shapefile-reader:** This component can analyze folders containing geographic information, extracting metadata from the shapefiles and enabling various actions based on this metadata.
- **gis-dsl:** This component defines a DSL for describing the data model and visualization model of a GIS. The DSL allows for the configuration of various aspects of a GIS application. Additionally, it includes an extendable parser that can be customized to perform additional actions.
- **spl-js-engine:** This JavaScript library uses an annotative approach to handle variation points, supporting three types of annotations. The *basic annotation* indicates whether a piece of code is included in the product source code. The *interpolation annotation* allows specific values from the specification to be included in the source code using JavaScript syntax. The *code generation annotation* is used to generate new files based on product specification data. According to the classification by [9], the tool characteristics are as follows:
  - ***Domain Analysis:*** Supports basic features and constraints.
  - ***Requirement Analysis:*** Validates product specifications against the feature model.
  - ***Domain Implementation:*** Uses an annotation-based approach with a low abstraction level and supports multi-language artifacts.
  - ***Product Derivation:*** Generates products but lacks validation beyond feature selection and traceability.
- **code-uploader:** Facilitates the deployment of software products in various environments, including local machines, remote servers via SSH, and AWS instances, accommodating diverse deployment scenarios.

2.2.2. Web-based GIS features

Fig. 2 shows a screenshot of a product generated by GIS-Publisher to illustrate the features offered by a product where the user can view the geometries and data of each shapefile (each shapefile corresponds to a layer on the map), with the SLD styles applied to each layer. The main part of the user interface is a map viewer with common features such as zoom (1), coordinate display and graphical scale (2), a reference map (3), and a tool to retrieve information about elements in the map (4). The product line allows for the support of multiple maps (5), but GIS-Publisher does not support this feature. The current map can be exported (6) as a PDF document or a permanent URL can be generated to be shared.

The application includes a powerful layer manager (7), shown in Fig. 3, where users can hide, center, rearrange, and change the opacity of individual layers. Alternative visualization styles can be added to layers, but they are not persisted for later sessions. Users can also add external Web Map Service (WMS) layers to the map (8 in Fig. 2).

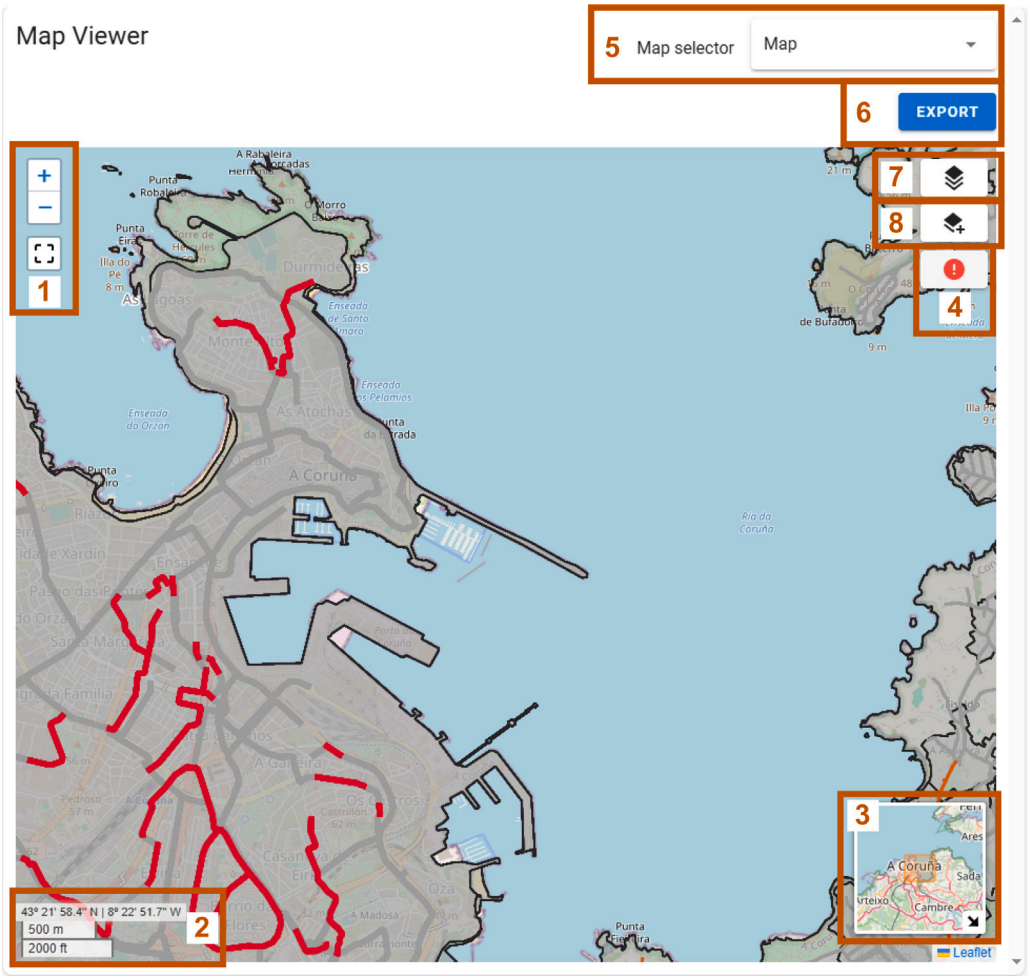


Fig. 2. Screenshot of the WaterSupply example generated by GIS-Publisher.

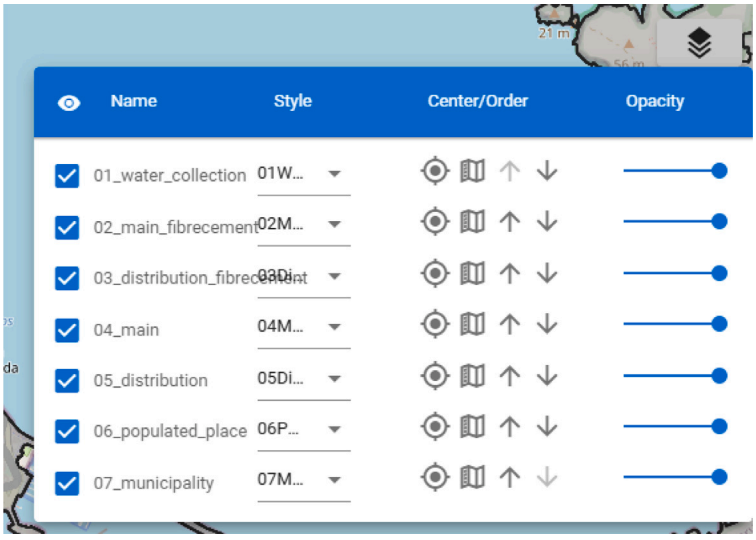


Fig. 3. Screenshot of the product layer manager with WaterSupply example layers.

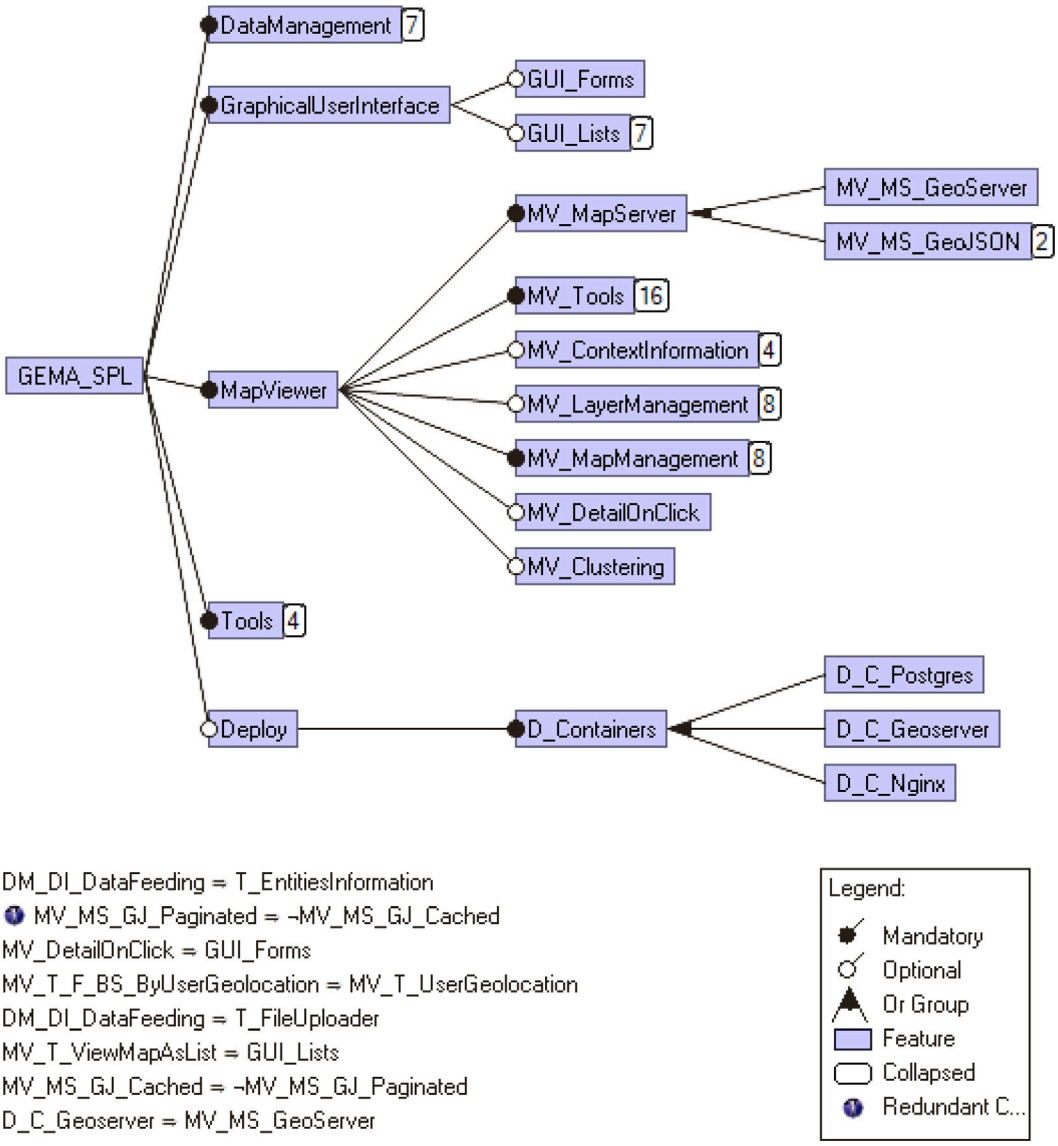


Fig. 4. Software product line feature model for customizing GisPublisher.

PostGIS,[11] a spatial database extension to PostgreSQL, is used for storing and managing geographic data, enabling efficient data retrieval. By default, map rendering is carried out by Geoserver via its WMS service. When the product is deployed, Docker will automatically configure and run the PostGIS and Geoserver services. The SPL can generate products that render data on the client side using GeoJSON as an alternative. Furthermore, to avoid visual clutter, the SPL supports geographic information clustering, which groups multiple geometries into a single shape. Additionally, users can view layer information through lists and forms, although this feature is not enabled by default. Finally, because of its responsive design, the user interface is usable on mobile devices.

The feature model of the SPL is displayed in Fig. 4 to show a glimpse of the alternative features that the user can select when a product is created. A fully expanded version of the figure can be seen in the project repository.[12]

3. Illustrative example

To illustrate the use of our tool, we will present a practical example involving asbestos cement pipes in a Spanish province. The deterioration of these pipes leads to the leakage of hazardous materials into the water system. The regional government of the province of A Coruña is aware of the deteriorating condition of the asbestos cement pipes and the associated hazards. They have identified the locations of these pipes and aim to publish a web-based GIS to inform the municipal governments. This tool will help municipalities know where the asbestos cement pipes are located so they can decide on the necessary actions to remove them.

The first step is to gather all the information about asbestos cement pipes and store it in shapefiles within a designated folder. Additionally, appropriate styles will be defined for these shapefiles using SLD files. The folder containing the files supporting the water supply example is shown in Fig. 5. Furthermore, the user can specify in the `config.json` the information required to deploy the finished system correctly. Listing 1 shows an example configuration to deploy on an SSH remote server. Additional fields will be required for different types

[11] https://postgis.net/
[12] https://github.com/lbdudc/mini-lps/blob/main/src/platform/model.png

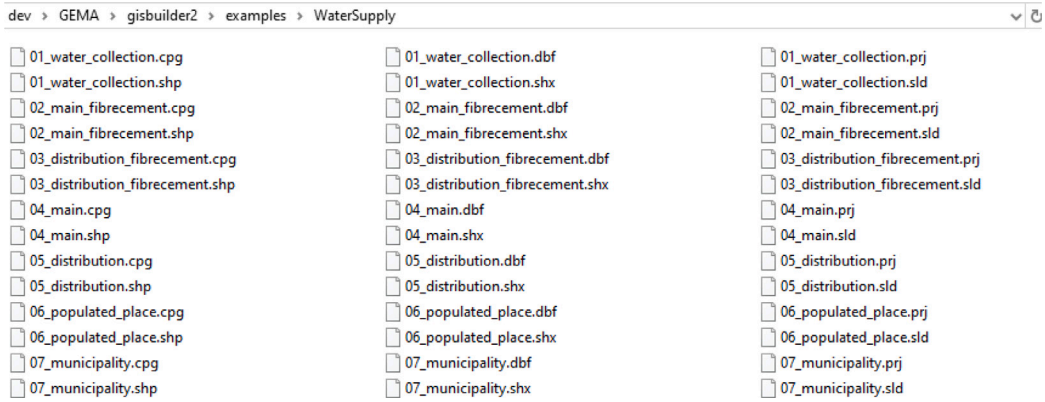


Fig. 5. Folder with the shapefiles needed in the Water Supply example.

of deployments; the tool's `README.md`[13] file contains more details on these deployment options.

```
{
  "deploy": {
    "type": "ssh",
    "host": "remote-host.com␣ || ␣IP",
    "port": 22,
    "username": "username",
    "certRoute": "/path/to/your/cert.pem",
    "remoteRepoPath":
        ↪ "/path/to/remote/repo/code"
  }
}
```

Listing 1 config.json file example for SSH configuration

After setting up a remote server, the user runs the command line shown in Listing 2. The second parameter indicates the location of the folder holding the necessary shapefiles. After the tool generates the product code, deploys the application to the remote server, and imports the shapefiles into the application, the user can view the final system by navigating to the remote server.

```
npx gispublisher ./examples/WaterSupply
```

Listing 2 Running the cli tool example

With the final application running the user can visualize the water network's current status, distinguishing between the asbestos pipes and the modern pipes, as well as displaying population centers and municipalities (see Fig. 2).

4. Impact

Researchers are traditionally given pre-made software tools, which may not satisfy their requirements. Building a new tool from scratch, on the other hand, requires even more time and resources. With the help of our tool, research groups can collaborate more successfully even when domain experts lack software development experience. Our software makes it easier to handle and analyze shapefiles, which reduces the time it takes to market and develop GIS systems. Offering an accessible, user-friendly, and cost-effective substitute for pricey proprietary software, increases researcher productivity. This improvement is particularly significant for projects with short timelines where quick deployment and prototyping are essential. It is no longer necessary for users to spend much time and money learning difficult programming languages or buying expensive software licenses. Alternatively, they may focus on their data analysis assignments and research goals. This change promotes a more inclusive research environment where a range of expertise can contribute to GIS projects.

Additionally, our work raises new research questions. At first, the tool can have advanced capabilities added to it (new functions in the map viewer, or new data formats for the geographic data, for example). Furthermore, there are several alternatives to manage the variability in GIS applications that can be investigated thanks to SPL engineering and variability management. Lastly, domains like environmental monitoring, urban planning, public health, disaster management, bioinformatics, and social sciences have particular needs that could be added to the tool, moving beyond a generic web-based GIS.

5. Conclusions

This paper describes a tool that creates a web-based geographic information system (GIS) from a collection of geographic information represented as shapefiles. The tool is built on a SPL framework that is configured automatically from the datasets. The generated products are centered around an interactive map viewer that allows end users to navigate and analyze geographic information. The tool is user-friendly for field experts and requires no prior web development experience. It also offers seamless automatic deployment on various platforms, including AWS, SSH servers, and local machines.

In future work, we aim to support additional data formats (e.g., Geopackage, OGC web services, or raster file formats) and more complex visualization models (e.g., including multiple maps). Additionally, we plan to provide users with greater flexibility by allowing them to customize visualizations through changes to the DSL or by configuring the product through modified feature selection. Additionally, it would be interesting to expand the services in which the product can be deployed (e.g., Microsoft Azure).

CRediT authorship contribution statement

Victor Lamas: Writing – review & editing, Writing – original draft, Validation, Software, Investigation, Conceptualization. **David de Castro:** Validation, Software, Investigation, Conceptualization. **Alejandro Cortiñas:** Validation, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Miguel R. Luaces:** Writing – review & editing, Writing – original draft, Validation, Supervision, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

[13] https://github.com/lbdudc/gis-publisher/blob/main/README.md

Acknowledgments

CITIC is funded by the Xunta de Galicia through the collaboration agreement between the Department of Culture, Education, Vocational Training and Universities and the Galician universities for the reinforcement of the research centers of the Galician University System (CIGUS); partially funded by MCIN/AEI/10.13039/501100011033 and "NextGenerationEU"/PRTR: [PLAGEMIS: TED2021-129245B-C21]; partially funded by MCIN/AEI/10.13039/501100011033 and EU/ERDF A way of making Europe: [EarthDL: PID2022-141027NB-C21]; partially funded by GAIN/Xunta de Galicia: [GRC:ED431C 2021/53].

Data availability

The source code and example data are available on github. A link is included in the paper.

References

[1] Keenan PB, Jankowski P. Spatial decision support systems: Three decades on. Decis Support Syst 2019;116:64–76.

[2] Kong L, Liu Z, Wu J. A systematic review of big data-based urban sustainability research: State-of-the-science and future directions. J Clean Prod 2020;273:123142.

[3] Duarte L, Teodoro AC, Santos P, Rodrigues de Almeida C, Cardoso-Fernandes J, Flores D. An interactive WebGIS integrating environmental susceptibility mapping in a self-burning waste pile using a multi-criteria decision analysis approach. Geosciences 2022;12(10).

[4] Duarte L, Queirós C, Teodoro AC. Comparative analysis of QGIS plugins for Web Maps creation. La Granja 2021;34(2):8–26.

[5] Batory D. Feature models, grammars, and propositional formulas. In: Proceedings of the 9th international conference on software product lines. Berlin, Heidelberg: Springer-Verlag; 2005, p. 7–20.

[6] Alvarado SH, Cortiñas A, Luaces MR, Pedreira O, Places ÁS. Developing web-based geographic information systems with a DSL: proposal and case study. J Web Eng 2020;19(2):167–93.

[7] Voelter M, Visser E. Product line engineering using domain-specific languages. In: 2011 15th international software product line conference. 2011, p. 70–9.

[8] Cortiñas A, Luaces MR, Pedreira Ó. spl-js-engine: a JavaScript tool to implement software product lines. In: Proceedings of the 26th ACM international systems and software product line conference, vol. B. New York, NY, USA: Association for Computing Machinery; 2022, p. 66–9.

[9] Horcas JM, Pinto M, Fuentes L. Empirical analysis of the tool support for software product lines. Softw Syst Model 2023;22(1):377–414.