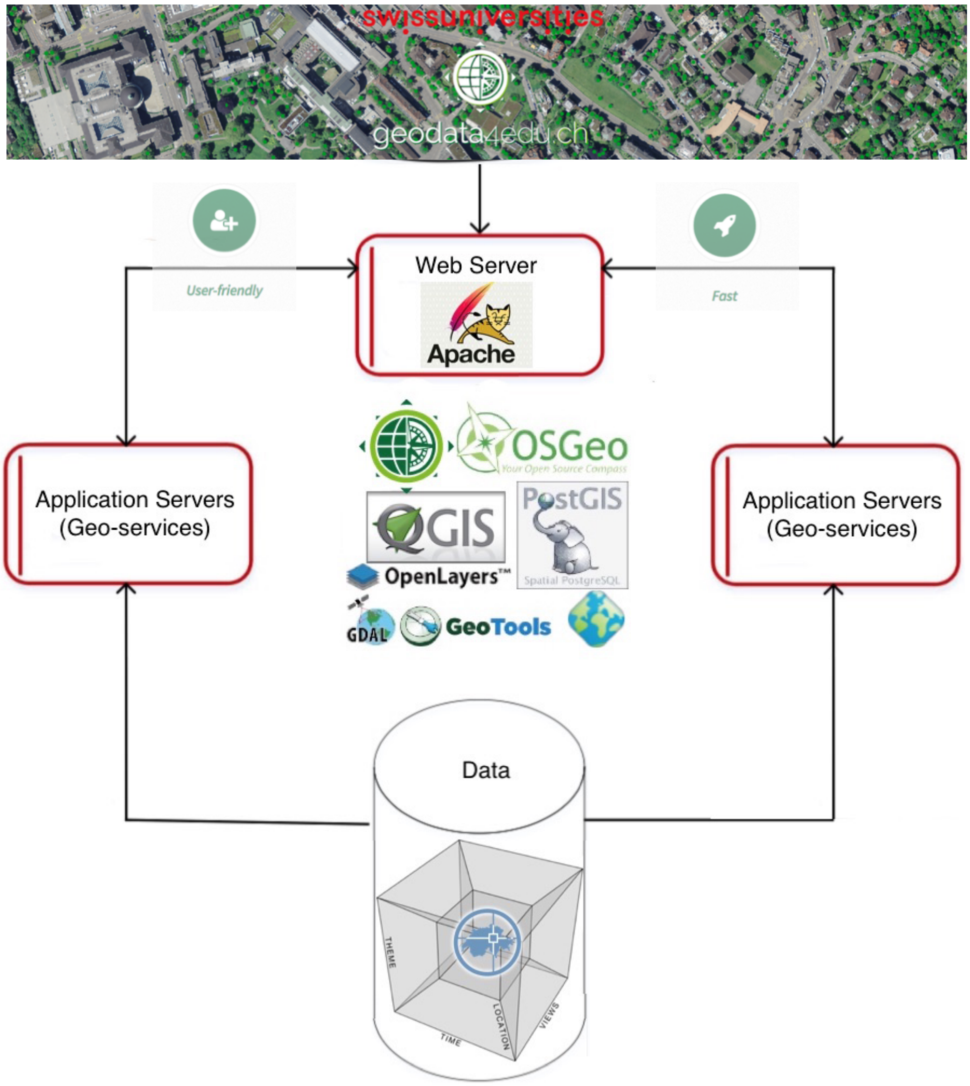
swissuniversities

geodata4edu.ch

Geodata Download Service

Technology Overview

(updated: January 2016)



Ionuț Iosifescu-Enescu

Technical Project Manager Geodata4SwissEDU

Institute of Cartography and Geoinformation

ETH zürich

Executive Summary

GeoVITe (Geodata Versatile Information Transfer environment), which is providing the download functionality of the Geodata4EDU national service, is based on a three-tier architecture, with a clear separation between the data (managed by a back-end data management layer), the application (consisting of server-side geo-services) and the presentation (front-end; client-side user interface).

The available vector and raster datasets are managed in back-end systems such as PostgreSQL geodatabases and Network Attached Storage (NAS) shares. The server-side application layer was developed around servers hosting geoprocessing and view services based on well-known open source software and libraries such as QGIS Server, GDAL/OGR and GeoTools, and enhanced by several Java servlets creating an Application Programming Interface (API) around the data management layer. The portal's Graphical User Interface (GUI) is built using well-known Web technologies such as HyperText Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript (JS) on top of GeoAdmin3 and OpenLayers3 frameworks. The entire GeoVITe GUI is also wrapped by Java server technologies such as Java servlets and Java Server Pages (JSP) in order to enforce authentication and proper communication with the server-side application layer. As a consequence, the "thin" Web-based interface in GeoVITe handles the majority of user interactions by sending requests and listening to responses from the server-side application layer. In turn, the geo-services access the data in the back-end, do the necessary data processing (for viewing or downloading) and return the corresponding responses back to the user interface.

The GeoVITe technologies support the implementation of academic spatial data infrastructures (SDIs) and geoportals that are interoperable with commonly used open source and proprietary GIS software such as QGIS or ESRI ArcGIS. The GeoVITe portal and services also serve as the corporate SDI of ETH Zurich, which is now part of the comprehensive national service offered to the Geodata4EDU consortium members.

Zusammenfassung[1]

Die GeoVITe (Geodata Versatile Information Transfer environment) Plattform, welche Downloadmöglichkeiten für den nationalen Dienst Geodata4EDU bereitstellt, basiert auf einer dreischichtigen Architektur mit einer klaren Trennung zwischen Daten (verwaltet von einer Datenhaltungsebene im Backend), Applikationslogik (bestehend aus serverseitigen Geodiensten) und einer Präsentationschicht (realisiert durch eine clientseitige Benutzeroberfläche im Frontend).

Die vorhandenen Vektor- und Rasterdatensätze werden von mehreren Systemen im Backend verwaltet, wie PostgreSQL Geodatenbanken und Network Attached Storage (NAS) Systemen. Die serverseitige Applikationslogik stellt Verarbeitungs- und Darstellungsdienste für Geodaten bereit, die auf bekannten Open Source Softwares und Bibliotheken wie QGIS Server, GDAL/OGR und GeoTools basieren, und erweitert diese mit mehreren Java Servlets, die eine Programmierschnittstelle (API) über der Datenhaltungsebene bilden. Die Benutzeroberfläche des Portals nutzt Webtechnologien wie die HyperText Markup Language (HTML), Cascading Style Sheets (CSS) und JavaScript (JS) innerhalb der GeoAdmin3 und OpenLayers3 Frameworks. Die komplette GeoVITe Benutzeroberfläche ist auf Java Server Technologien wie Java Servlets und Java Server Pages (JSP) aufgebaut, um Authentifizierung und die Kommunikation mit der serverseitigen Applikationslogik sicherzustellen. Diese webbasierte Schicht in GeoVITe regelt die Mehrzahl der Benutzeraktionen durch Senden von Anfragen und Empfangen von Antworten von der serverseitigen Applikationslogik. Die Geodienste greifen auf die Daten im Backend zu, verarbeiten sie zur Darstellung oder für den Download und liefern schliesslich der Benutzeroberfläche die entsprechende Antwort.

Die verwendeten Technologien in GeoVITe unterstützen die Umsetzung von akademischen Geodateninfrastrukturen (GDIs) und Geoportalen, die sowohl client- als serverseitig mit gängigen Open Source und proprietären GIS-Softwares wie QGIS oder ESRI ArcGIS kompatibel sind. Das Portal und die Dienste von GeoVITe dienen als überbetriebliches GDI der ETH Zürich, welches den Geodata4EDU Konsortialmitgliedern nun als ein nationaler Dienst zur Verfügung gestellt wird.

[1] Many thanks to R. Schnürer for supervising the German translation

Résumé[2]

GeoVITe (Geodata Versatile Information Transfer environment), qui fournit la fonction de téléchargement du service national Geodata4EDU, est basé sur une architecture 3-tiers avec une séparation claire entre les données (administrées par un tier de gestion de données en back-end), l'application (géo-services du côté serveur) et la présentation (front-end; interface utilisateur du côté client).

Les sets de données vectorielles et raster disponibles sont administrés dans des systèmes back-end tels que des bases de données spatiales PostgreSQL et des baies de stockage en réseau (NAS). Le côté serveur de l'application a été développé sur plusieurs serveurs qui hébergent des services de traitement géospatial et de consultation basés sur des logiciels libres bien connus tels que QGIS Server, GDAL/OGR et GeoTools. Leurs fonctionnalités ont été étendues grâce à plusieurs servlets Java qui créent une API autour du tier de gestion de données. L'interface graphique utilisateur du portail GeoVITe est construite avec des technologies Web bien connues telles que l'HyperText Markup Language (HTML), Cascading Style Sheets (CSS) et JavaScript (JS) superposées aux environnements GeoAdmin3 et OpenLayers3. L'interface utilisateur est également encapsulé dans des technologies serveur Java, tels que des servlets Java et des Java Server Pages (JSP), afin d'assurer l'authentification et la bonne communication avec les géo-services du côté serveur. En conséquence, l'interface de GeoVITe gouverne la majorité des interactions de l'utilisateur en envoyant des requêtes et en recevant les réponses des géo-services du côté serveur. De leur côté, les géo-services accèdent aux bases de données, font le traitement géospatial (pour l'affichage ou le téléchargement) et renvoient les réponses à l'interface utilisateur.

Les technologies de l'environnement GeoVITe soutiennent l'implémentation d'infrastructures de données spatiales et de géoportails au sein du milieu académique. Elles sont compatibles avec des logiciels SIG populaires comme QGIS ou ESRI ArcGIS. Le géoportail et les services GeoVITe représentent également l'infrastructure de données spatiales de l'EPF Zurich et font maintenant partie du service national offert aux membres du consortium Geodata4EDU.

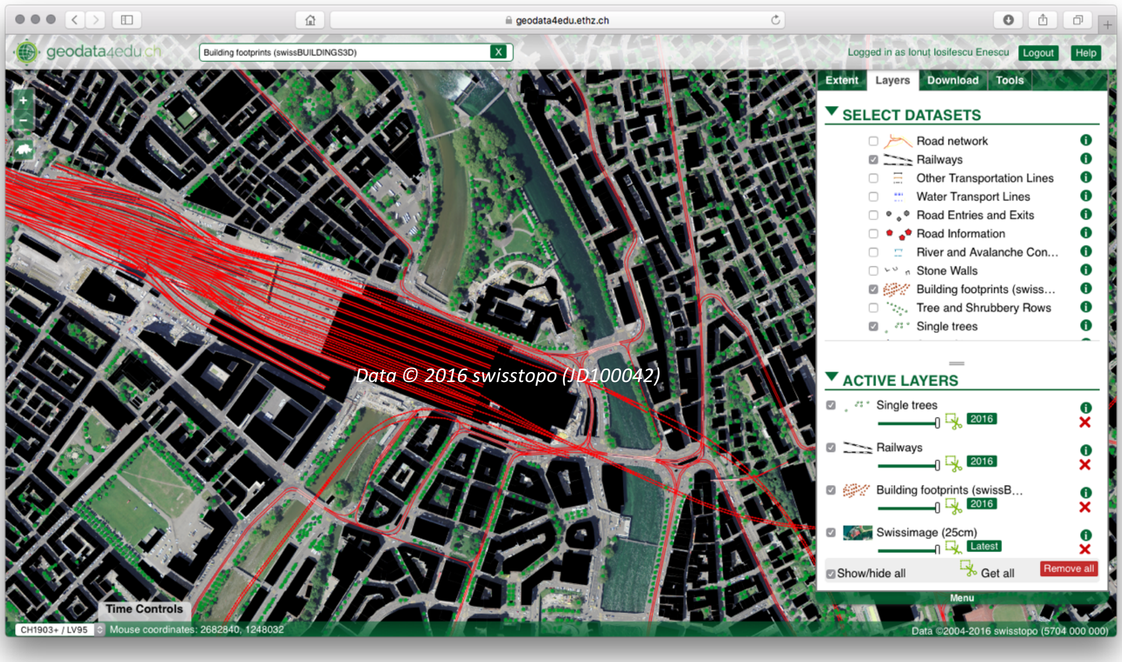
[2] Many thanks to N. H. Panchaud for supervising the French translation

Contents

1 Introduction and History of GeoVITe

Geospatial information is not only necessary for taking many daily decisions, but also for teaching and scientific research. Hence, reliable access to (spatial) data is paramount for higher education institutions. However, while such data have been available in some form at Swiss universities, so far the access to geospatial datasets has been cumbersome and difficult for any non-expert in the field of geospatial information.

GeoVITe (Geodata Versatile Information Transfer environment), which is delivering the download functionality of the Geodata4EDU national service, aims to redefine the way researchers and students are accessing geospatial information. It provides an intuitive and easy to use web interface for fast access to the most important spatial datasets provided by the Federal Office of Topography swisstopo. Any user can visually browse the spatial data, select the desired area and download the necessary data in a straightforward manner through a standard Web browser as illustrated in the next figure.



GeoVITe, as the corporate SDI of ETH Zurich, has been in development since 2004 and the GeoVITe portal has been actively used for the download of geodata since 2010. Between 2004 and 2015, GeoVITe has experienced a long history of implementation technologies, migrating from proprietary technologies to open software and standards. The first GeoVITe prototype, finished in 2008, was mainly based on technologies such as IBM's DB2 (database), ESRI ArcSDE (database middleware) and ESRI ArcIMS (services). The subsequent GeoVITe platforms, from 2009 to 2012, also continued to be mainly powered by newer versions of ESRI ArcGIS Server technology, until the version operating between 2013 and 2015, harmoniously blended open source alongside proprietary technologies.

In 2015 - 2016, GeoVITe was completely redesigned in the frame of the Geodata4SwissEDU project, a project substantially co-financed by swissuniversities.ch. GeoVITe's new version is a fundamentally different (graphical as well as a structural) redesign of the previous, ETH-internal GeoVITe platform, extending the provided geodata download service to all universities of Switzerland while at the same time offering improved scalability, usability and versatility over its predecessor.

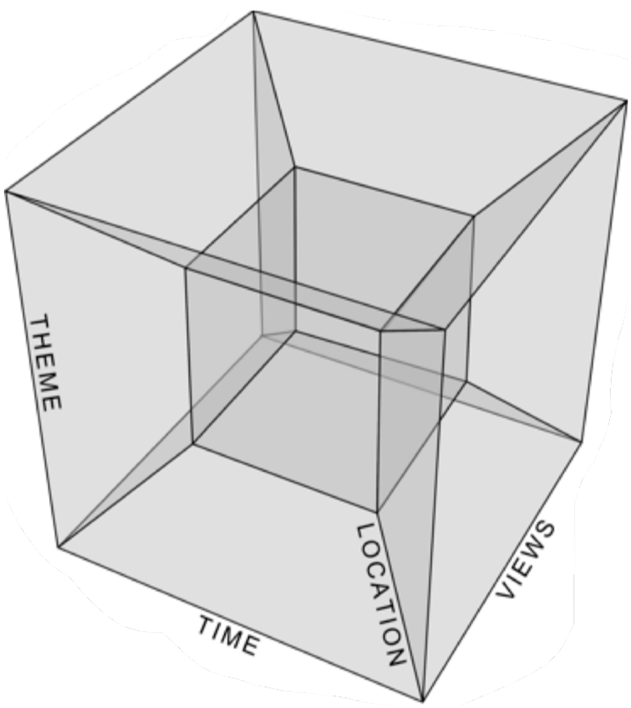
2 Fundamental Architectural Concepts

The GeoVITe Architecture, as redesigned in the frame of the Geodata4SwissEDU project, is centred around three fundamental guidelines: (1) assurance of a coherent user experience through a hypercube-based model, (2) standards-based, service-oriented architecture and (3) the consideration of non-functional requirements.

2.1 The Hypercube-based Model

The assurance of a coherent user experience for different types of products was the most important guideline that influenced the architectural elements involved in the structuring and presentation of geospatial information. This is due to the fact that the heterogeneity and complexity of the data requires a generic data and visualization model that supports the navigation, exploration, and visualization inside a geoportal. As a consequence, the Hypercube-based model was created for providing users with a coherent, efficient and conceptually sound method for exploring and accessing GeoVITe datasets, users who may not know in advance what data is available and may want to visually explore the entire geoinformation space offered through GeoVITe.

The Hypercube-based model represents an extension of the space-time cube model, in order to deal with the different representation of spatio-temporal data. The first conceptual dimension in the hypercube-based visualization model is the location, namely a region of interest on the geographic surface of the Earth, defined by extent coordinates in a specific coordinate reference system. The next dimensions are given through the selection of the themes that are available at a location followed by their availability on the time axis. If a theme (e.g. swissimage, swissALTI3D, DHM25) is available at a location, it must have a cardinality of at least one, even though a longer time-range may not necessary be available (as for example in the case of DHM25).



The final dimension, namely the views, originates from the fact that it is possible to have multiple topics on specific slices/datasets in the spatio-temporal hypercube. For example, the swisssALTI3D dataset from 2015 covering the entire Switzerland can be viewed as a digital elevation model, as its derived terrain slope or as a shaded relief, representing different topics derived from the same data; similarly, for swissimage, we have different scales (from 0.25m to 25m), determining the permitted maximal size of the download extent.

It is also worth mentioning that this model is compatible with many future functionality extensions in the area of visual analysis or user-defined symbology customization, because it theoretically supports an infinite number of custom data classifications, colour schemes and similar extensions of data visualization.

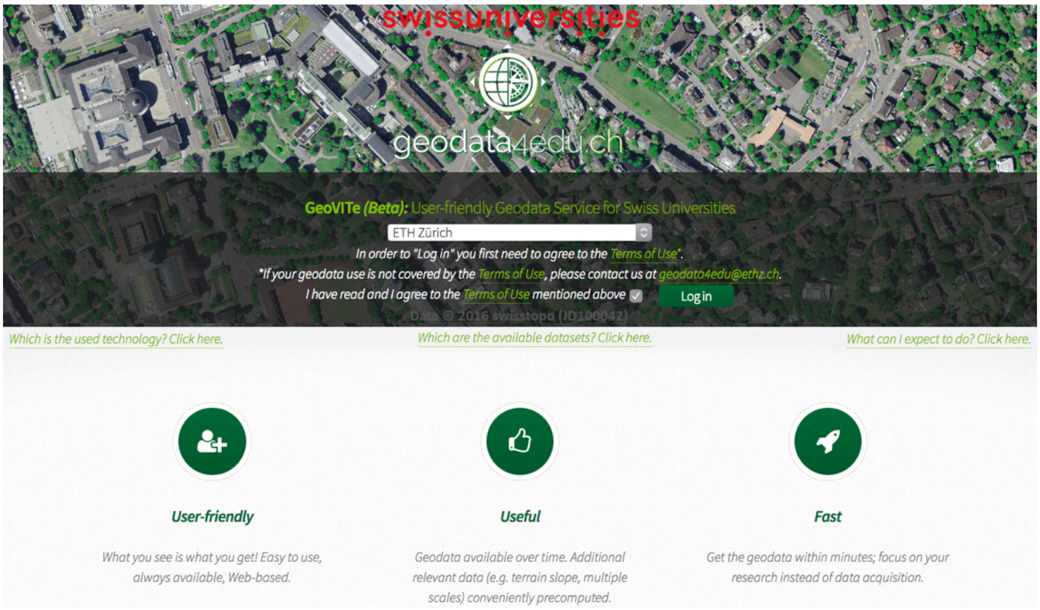
2.2 Standards-based Service-oriented Architecture

The second guideline for the redesign of the GeoVITe architecture was to be based on open standards that enable interoperability and reusability of the underlying geospatial services. In this context, there are available a number of high quality open source software for the geospatial domain, as for example PostgreSQL[3]/PostGIS[4], QGIS Server[5], GDAL/OGR[6], GeoTools[7] and OpenLayers3[8], which enables GeoVITe to conform with various Open Geospatial Consortium (OGC) and World Wide Web Consortium (W3C) standards, as for example the Simple Features for SQL, Web Map Service (WMS), Web Feature Service (WFS) and Geographic Mark-up Language (GML), Web Coverage Service (WCS), Styled Layer Descriptor (SLD) and Feature Encoding Standard (FE), GeoJSON (Geographic JavaScript Object Notation), GeoPackage, HTML 5 or ECMAScript 5.

Furthermore, the OGC Web services and standards allows GeoVITe to be organized in a generic three-tier architecture, containing a Data Management, an Application and a Presentation tier (please refer to chapter 3 – "Technical Overview" for details).

2.3 Coverage of Non-Functional Requirements

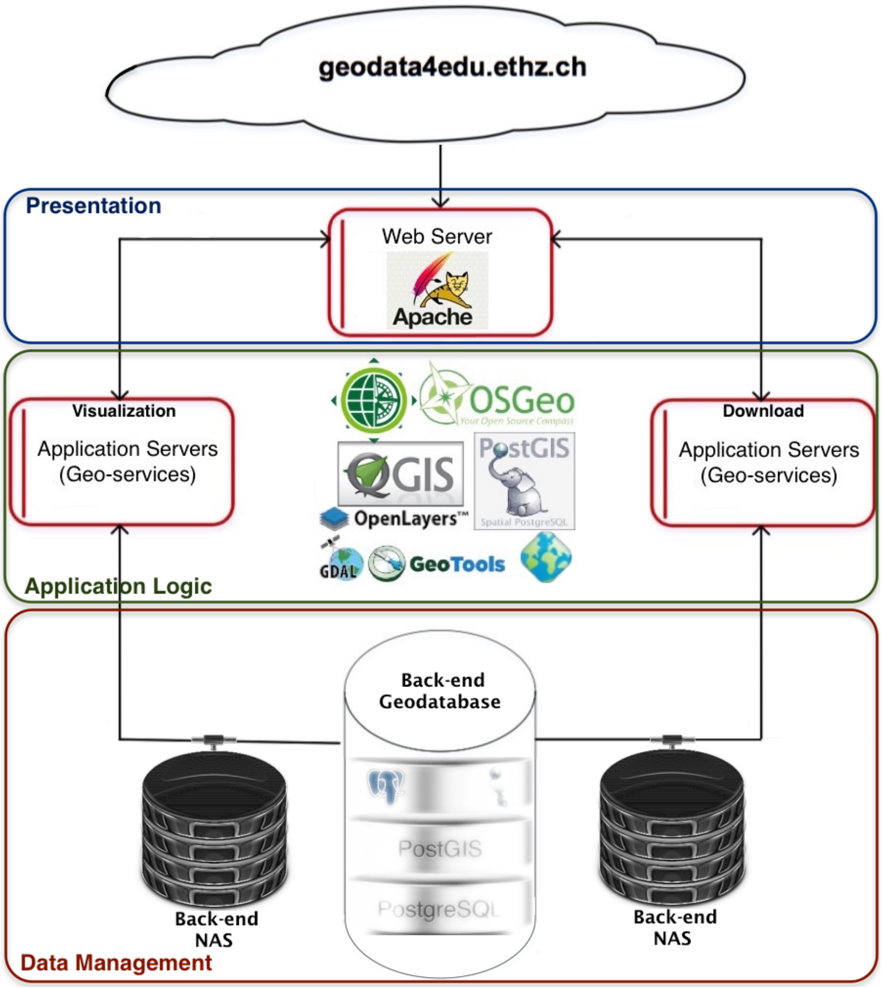
The third guideline introduced the consideration of non-functional requirements (NFRs) in the architectural design of GeoVITe. We considered that NFRs are having a direct influence on the quality attributes of a system. In GeoVITe we have considered the following generic NFR categories based on the taxonomies of Roman (1985) and Rashwan (2013), more specifically usability, security, reliability, maintainability, life-cycle, and compatibility with the economic and political environment. Due to the fact that not all NFRs could be optimally addressed, key design decisions required complex trade-offs between usability, availability, modularity, security, confidentiality, stability, performance and cost-effectiveness, with the goal of designing a versatile system that is user-friendly, useful and fast.



[3] https://www.postgresql.org/
[4] http://www.postgis.net/
[5] http://docs.qgis.org/2.14/en/docs/user_manual/working_with_ogc/ogc_server_support.html
[6] http://www.gdal.org/
[7] http://www.geotools.org/
[8] https://openlayers.org/

3 Technical Overview

GeoVITe is based on a three-tier architecture, with a clear separation between the data (managed by a back-end data management layer), the application (consisting of server-side geo-services) and the presentation (front-end; client-side user interface), as illustrated in the following figure:



3.1 The Presentation Layer

The portal's Graphical User Interface (GUI) is built using well-known Web standards such as HTML, CSS and JavaScript on top of GeoAdmin3[9] and OpenLayers3[10] frameworks. The main use of the swisstopo's GeoAdmin framework, which is also based on OpenLayers 3, was to allow a simple integration of swisstopo's map services into the platform via simple identifiers. Although the GeoAdmin framework is currently disabled in GeoVITe (because of compatibilities with our VectorTiles scheme), a switch between the Geodata4SwissEDU WMS servers and swisstopo services is possible.

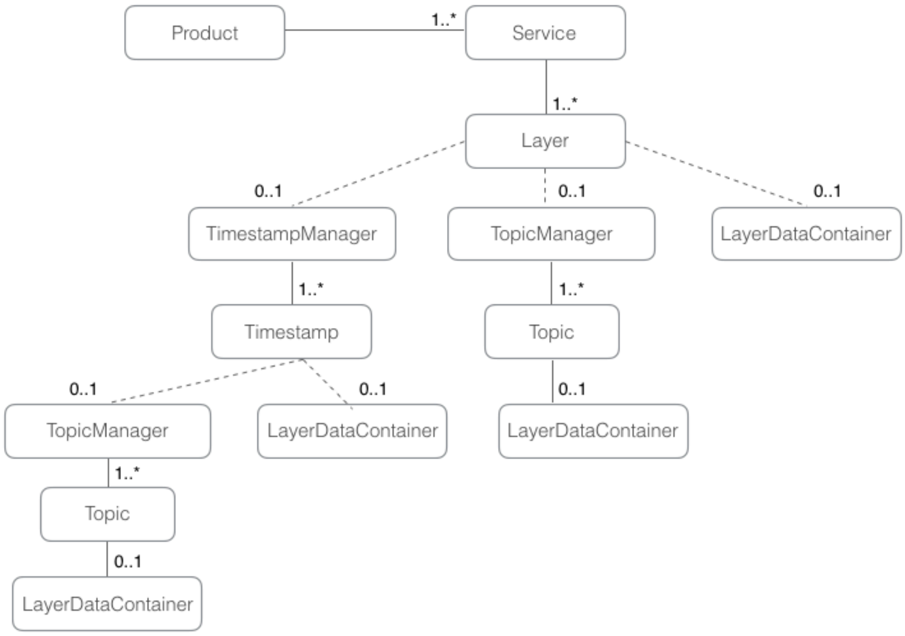
The entire GeoVITe GUI is wrapped by Java server technologies such as Java servlets and Java Server Pages (JSP) served by an Apache Tomcat 8.x container and ensuring authentication and proper communication with the server-side application layer.

[9] http://api3.geo.admin.ch/
[10] https://openlayers.org/

The GeoVITe Graphical User Interface (GUI) is organized, as much as the JavaScript language allows, in an object-oriented manner. The initialization of the portal is performed by the **map.js** script, instantiating the following important GeoVITe elements:

- GeoVITe Map Pane (including control add-ons and add-on projections)
- GeoVITe Available Layers, Products and Services (see Diagram below)
- Active Layers from (1) a portal API call (e.g. from a metadata Link of a search result, providing a layer name in the portal URL), (2) the previously "selectedLayers" cookie (layers that were selected the last time by the user), or (3) a set of default layers (in the absence of (1) or (2)).
- Saved Download Lists and Extents

The GeoVITe Available Layers, Products and Services are organized as in the following diagram:



The following cardinalities hold true:

- A **Product** (e.g. "Topographic Vector Maps") contains one or more **Services** (e.g. "swissTLM3D")
- A **Service** contains one or more **Layers** (e.g. "Primary Surfaces, Landcover"). In the GUI, Layer entries are grouped in a drop-down menu below the parent Service. If the Service contains exactly one Layer, there will be no drop-down menu; instead the Service itself will be styled to look like a selectable Layer.
- A **Layer** may contain only one of the following three items: **TimestampManager**, **TopicManager**, or **LayerDataContainer**. The TimestampManager is used if a Layer has multiple **Timestamps** (e.g. "National Map 1:100'000" > "1978"). Likewise, the TopicManager provides control over all available **Topics** per Layer (e.g. "swissALTI3D": topics "Relief", "Slope", "Aspect" and "Elevation").
- A **LayerDataContainer**, used by every other class requiring layer management, as it provides the most important data for viewing and downloading for each layer, timestamp and topic.

Furthermore, the download functionality is combined into the file **downloadCart.js**, handling both saved and shared download lists, but also provides functionality of initiating a download, polling the status and linking to download URLs.

3.2 The Server-side Application Layer

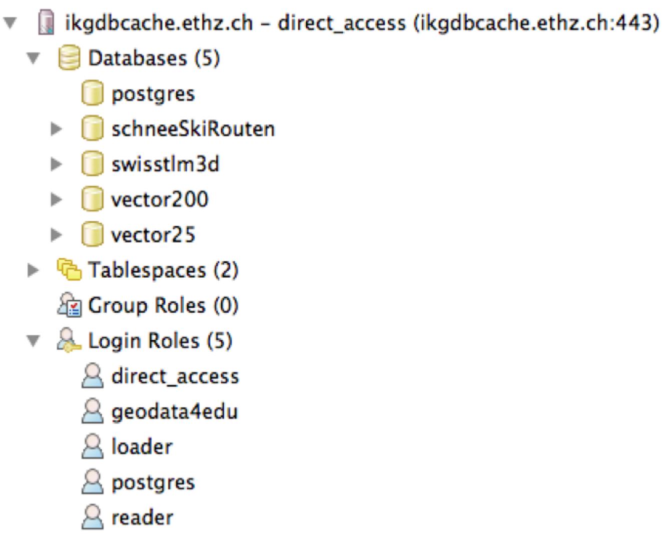
The server-side application layer was developed around servers hosting geoprocessing and view services based on well-known open source software and libraries such as QGIS Server, GDAL/OGR and GeoTools. Furthermore, the application layer is extended by several Java servlets creating an Application Programming Interface (API) around the data management layer and handling API calls to the geoprocessing and view services. The main components of the Application Layer are presented in the following, with servlets and standard Java classes highlighted in bold:

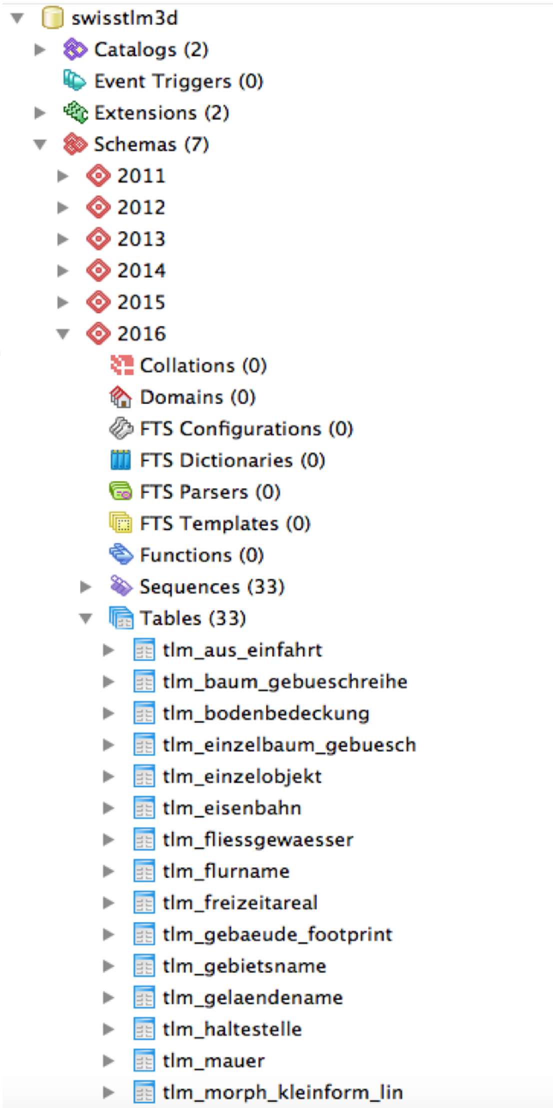
- **ServerProperties:** singleton providing all servlets and instances with globally valid metadata.
- logincheck:
  - **AAILoginCheck:** called when the user logs in via the "Log In" button of the main portal page.
  - **Logout:** clears all session and user data, then sends a redirect to the index page of the portal.
- portalRequests:
  These fulfil specific tasks initiated by the GUI.
  - **LayerTimestampSearch:** if a user enters a search term into the text area in the GUI that corresponds to a layer name, and if a bounding box is drawn, this servlet receives the layer name and bounding box and evaluates the timestamps of the given layer for which there is data in the bounding box.
  - **Legend:** returns the legend image, embedded in HTML formatting, for the specified layer (if present).
  - **QGISservlet**: composes a direct WMS access QGIS file for the given layer, timestamp, and topic.
- proxy
  - **AjaxProxyServlet:** general-purpose proxy servlet
  - **ServiceProxyServlet:** gatekeeper for securing servers hosting generic GeoVITe services that should not be accessed directly by users
  - **TileCache:** similar functionality with the ServiceProxyServlet but specialized for serving cached Tiled-WMS requests. The servlet first tries to look in the local tile cache directory whether that tile has already been requested and sends back a potentially found match directly without querying the WMS server. If the tile has not (yet) been requested, it forwards the request to the back-end WMS server, sends back its answer and at the same time stores the tile to the local tile cache.
  - **VectorTileCache:** closely related to the "TileCache" servlet, also performs an authentication for each request, but is responsible for the caching of *vector* tiles instead. In addition, this servlet acts as the tile server itself by querying a PostGIS database, composing GeoJSON files, caching them and sending them back to the client.
  - **VectorDBConnector:** singleton providing a database connection to the vector database. Parameters are retrieved from the "ServerProperties" instance.
- userContent
  - **BBoxServlet:** provides functionality to store, retrieve, delete, import and share the bounding boxes / extents for the users.
  - **CartServlet:** stores, retrieves, deletes, imports and shares the download carts / lists.
  - **Extents:** stores, retrieves, deletes, imports and shares the extents / bounding boxes
  - **UserDBConnector:** singleton providing a database connection to the database storing all user-generated content (carts, bounding boxes, extents).

3.3 The Data Management Layer

The available vector and raster datasets are managed in back-end systems such as PostgreSQL geodatabases and Network Attached Storage (NAS) shares. The data management layer of GeoVITe was also heavily influenced by the conceptual hypercube model previously presented, with folder structures and databases organized accordingly.

In PostgreSQL geodatabases, each available vector product (such as swissTLM3D, vector25 or vector200) is structured in its own database, with the GeoVITe application logic accessing the data only in read-only mode. For illustration purposes, the following visual of a development database server is provided, presenting the structuring of data according to products, years, data layers and topics.





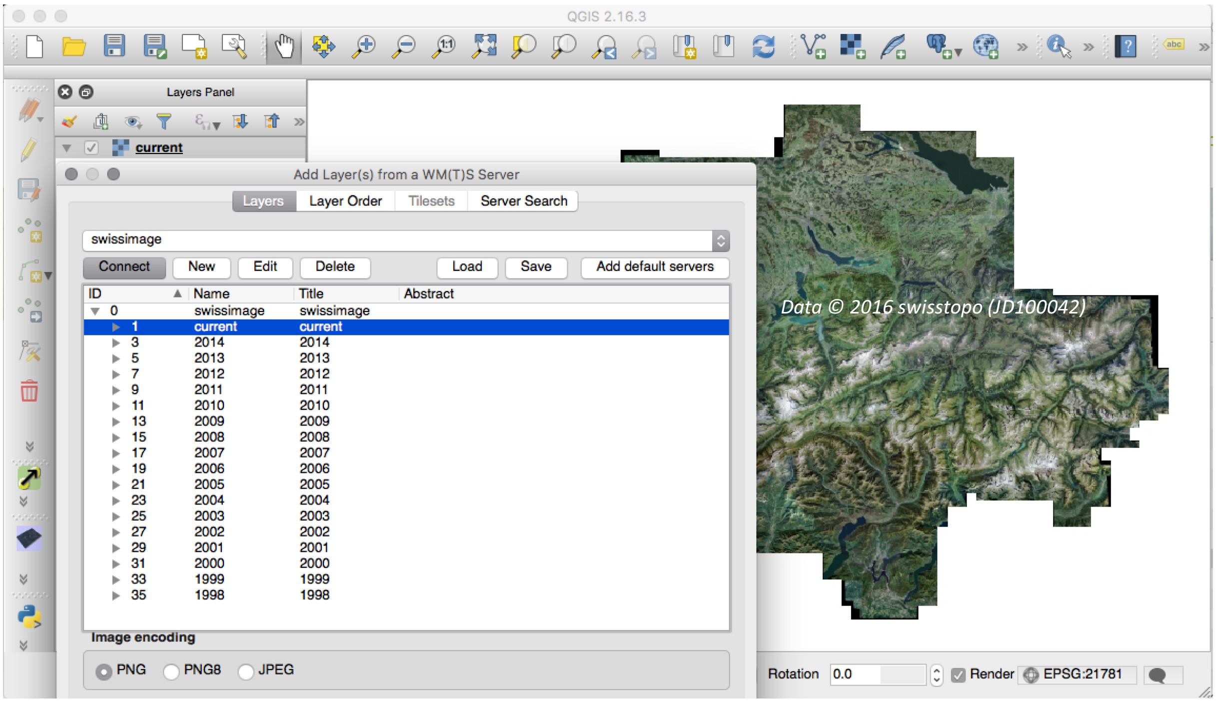
Further raster geospatial data is stored in NAS shares administered by the IT Services of ETH Library and ETH Zurich. For obvious security reasons, also these NAS shares are accessed by the GeoVITe application layer only using only read-only credentials.

3.4 Testing back-end GeoVITe Databases and Services

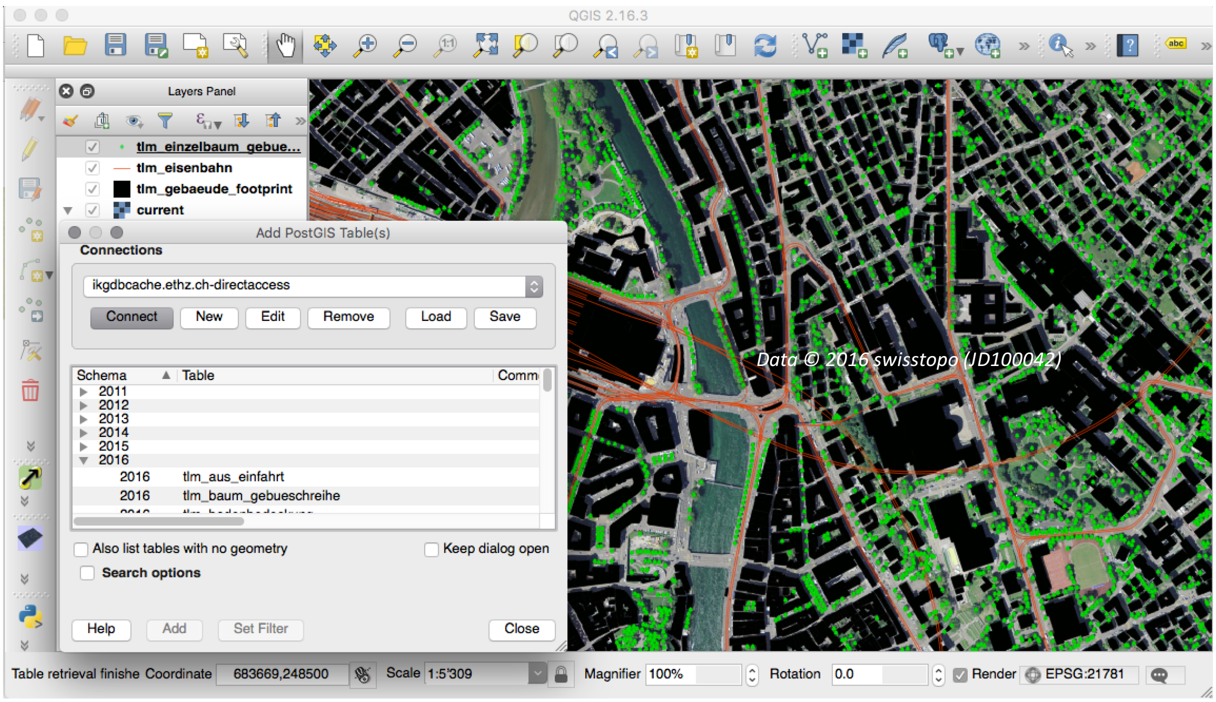
The GeoVITe databases and services can be managed and tested in a straight forward manner using open source GIS software such as QGIS Desktop[11] or proprietary software such as ESRI ArcGIS. The following two images illustrate how available WMS raster layers (e.g. swissimage) and vector PostGIS tables can be added as layers in QGIS, a procedure that also demonstrates how the back-end GeoVITe services could also be directly accessed by GIS professionals in the future.

[11] http://www.qgis.org/

Illustrative usage of GeoVITe WMS servers in QGIS (similar to ESRI ArcGIS):



Directly accessing GeoVITe PostGIS vector layers in QGIS (similar in ESRI ArcGIS):



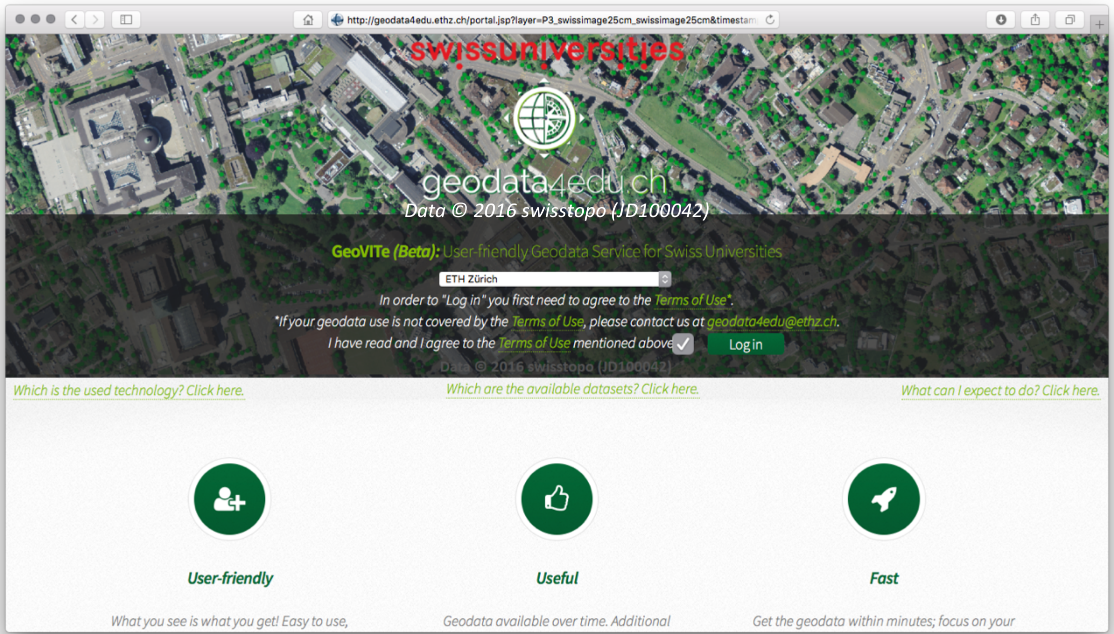
4 Conclusions

From a technical implementation point of view, GeoVITe is entirely service-driven. We use a three-tier architecture, with a clear separation between presentation, services and data. The geoportal user has access to a Web-based interface, which handles the majority of user interactions by sending requests and listening to responses from visualization and geoprocessing services. The services access the corresponding data in the database, do the necessary processing and send the responses back to the user interface. From a software engineering viewpoint, these background services are entirely transparent for the users: they only see that the interface reacts to their commands such as navigating through the different products available, zooming or panning the map, selecting the right area for download, and enabling them to straightforwardly download the desired data.

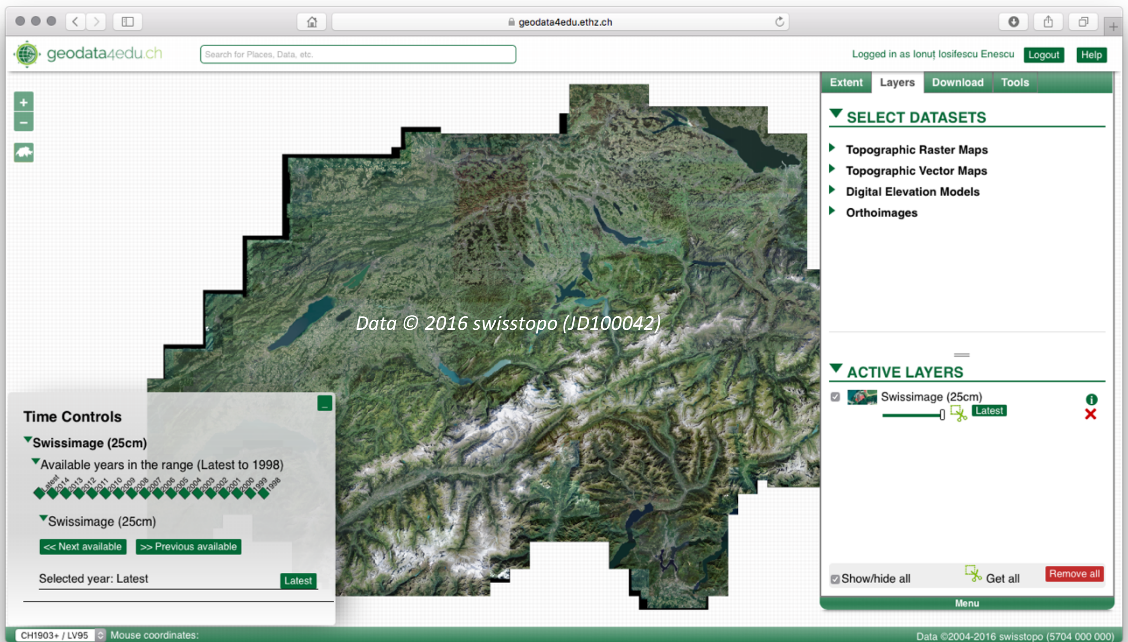
The GeoVITe technologies support the implementation of academic spatial data infrastructures (SDIs) and geoportals that are interoperable with commonly used open source and proprietary GIS software such as QGIS or ESRI ArcGIS. The GeoVITe portal and services also serve as the corporate SDI of ETH Zurich, which is now part of the comprehensive national service offered to the Geodata4EDU consortium members.

Appendix: Use Case for GeoVITe Portal Usage

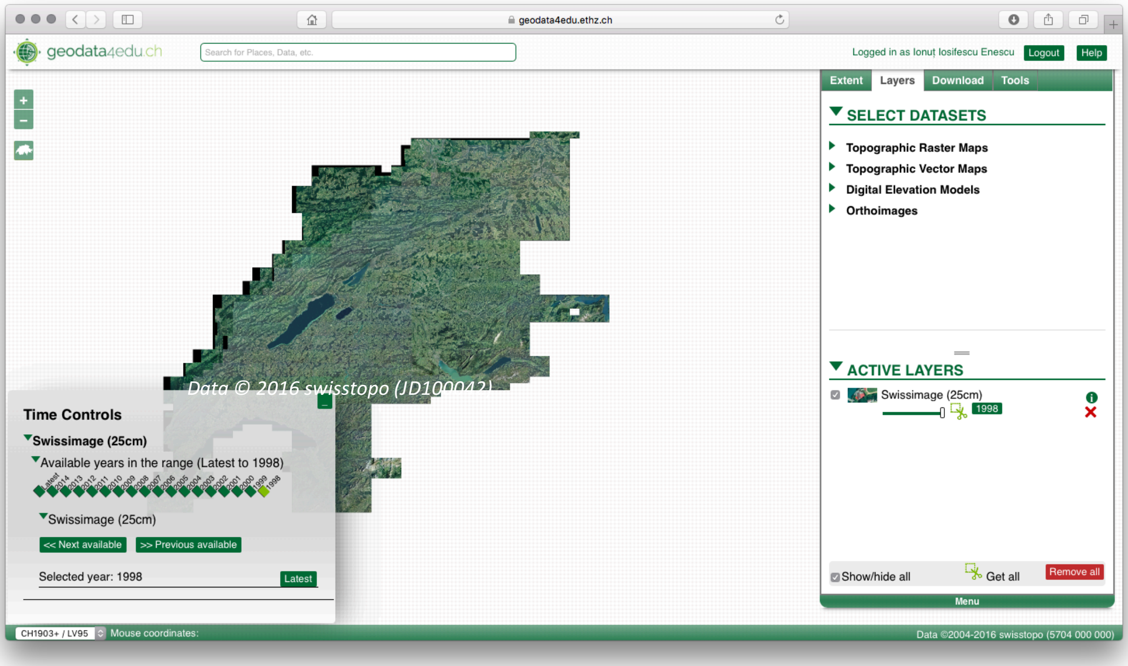
After a search for “swissimage” in the main geodata4edu.ch Website, a user might follow the corresponding product link for the latest swissimage to the login page of the GeoVITe portal: e.g.http://geodata4edu.ethz.ch/portal.jsp?layer=P3_swissimage25cm_swissimage&timestamp=Latest&topic=25cm



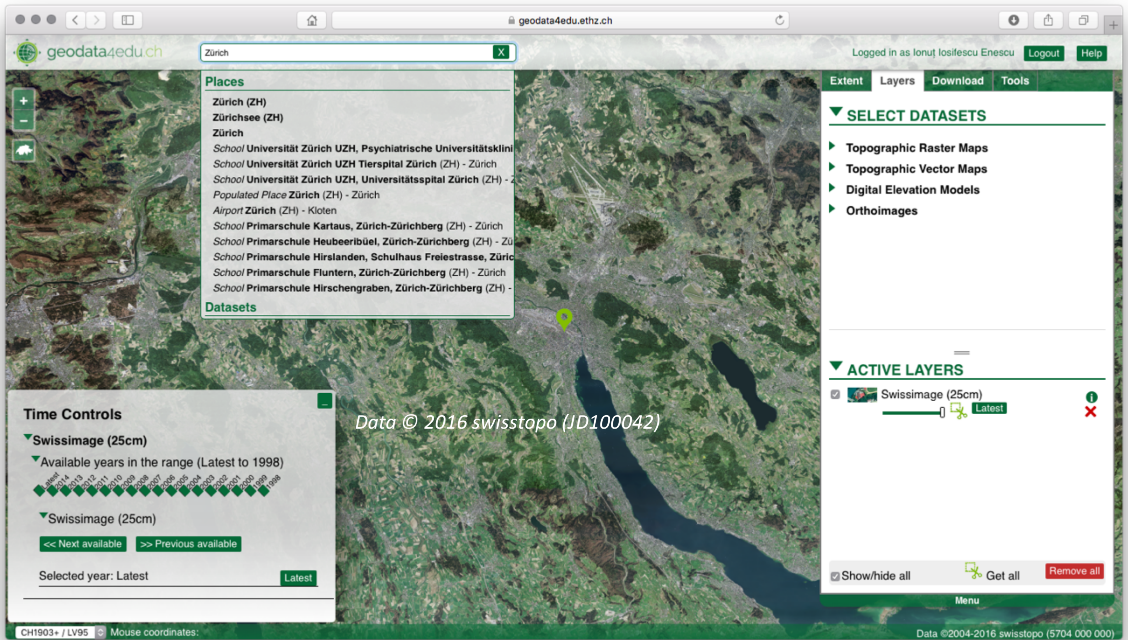
After the acceptance of the “Terms of Use”, which restricts the swisstopo geodata usage to only teaching and research purposes, and the mandatory AAI login, the user will be permitted to access GeoVITe having the desired dataset – swissimage – already preselected in the Active Layers.



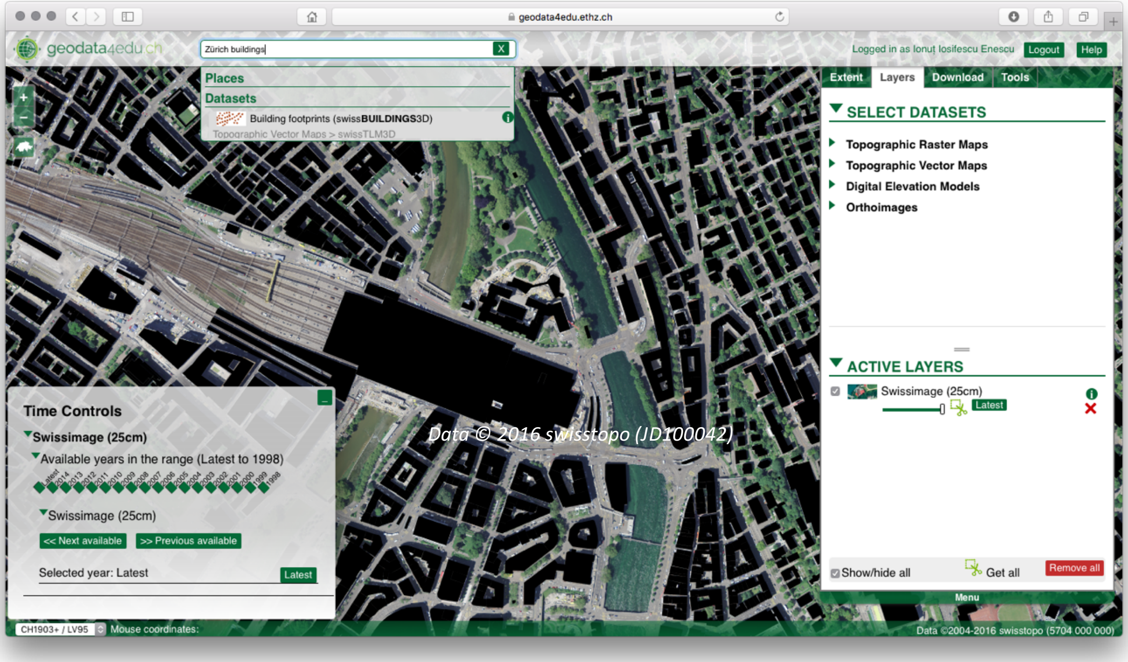
“Time Controls“ are available for every active layer with more than one temporal version and allows the user to quickly access previous product versions.



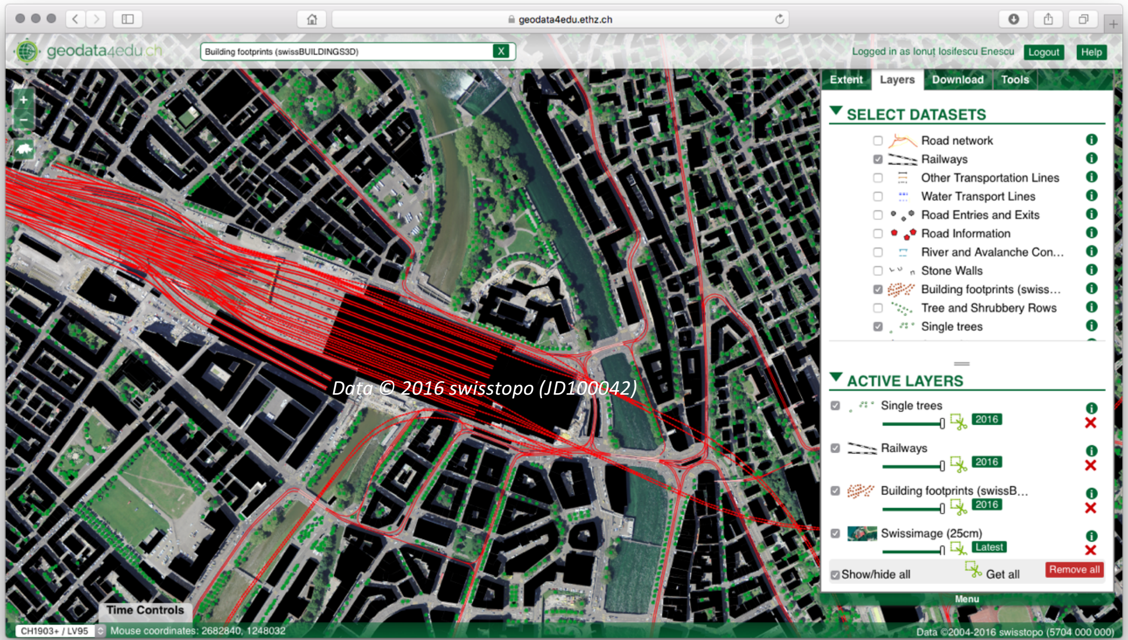
Assuming that the user is interested in the latest swissimage for an area around the city centre of Zurich, the user can either zoom in by intuitively using the mouse or by searching for “Zurich” in the universal search bar available in the upper left side of the user interface and selecting the first result.



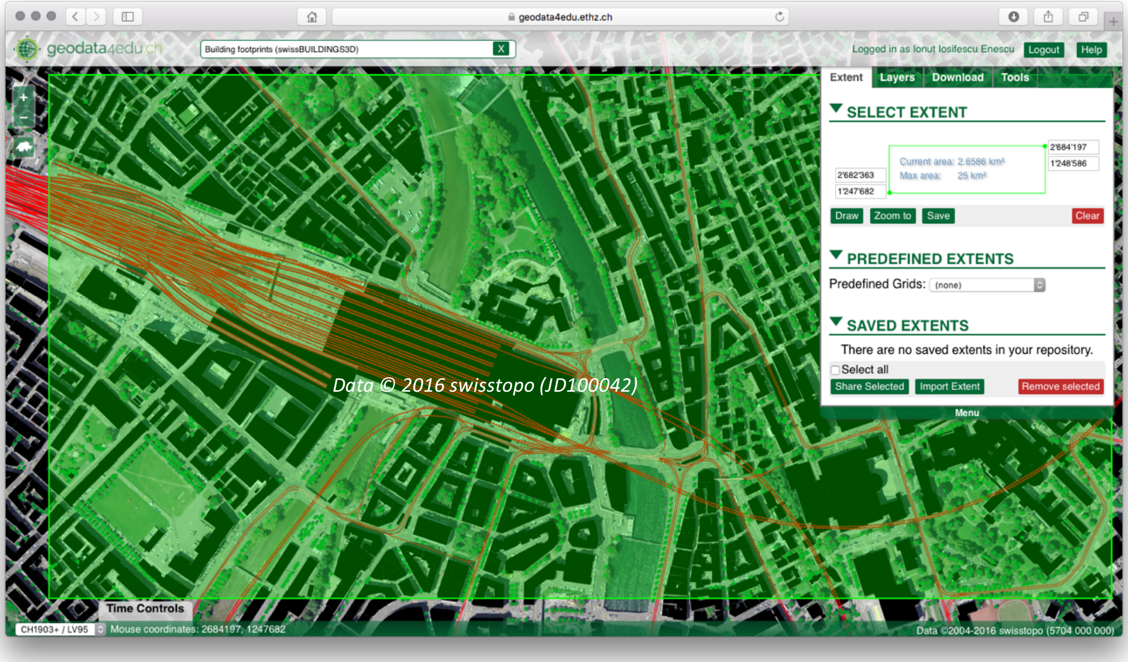
The universal search bar is also useful if the user wants to navigate to specific coordinates in the map or if other datasets (e.g. buildings) want to be quickly added to the active layers.



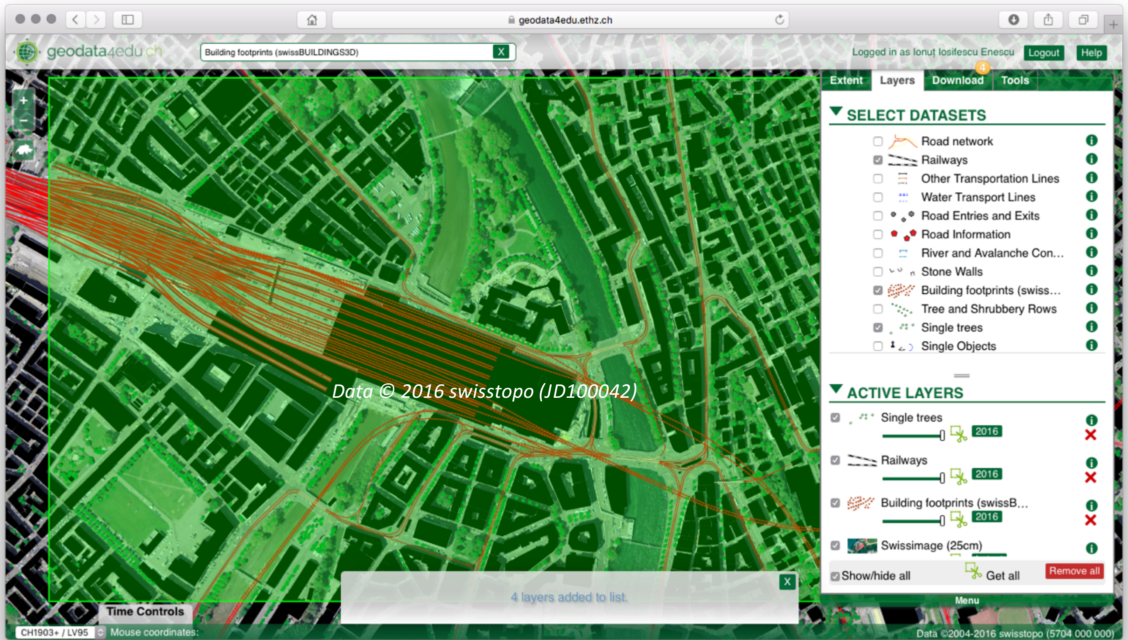
Furthermore, the user can also activate additional datasets by directly selecting them from the "Select Datasets" hierarchical layer tree available in the right-side collapsible menu.



For downloading the datasets, the user must mark the area of interest, by directly drawing the desired “Extent” or by entering the map coordinates.

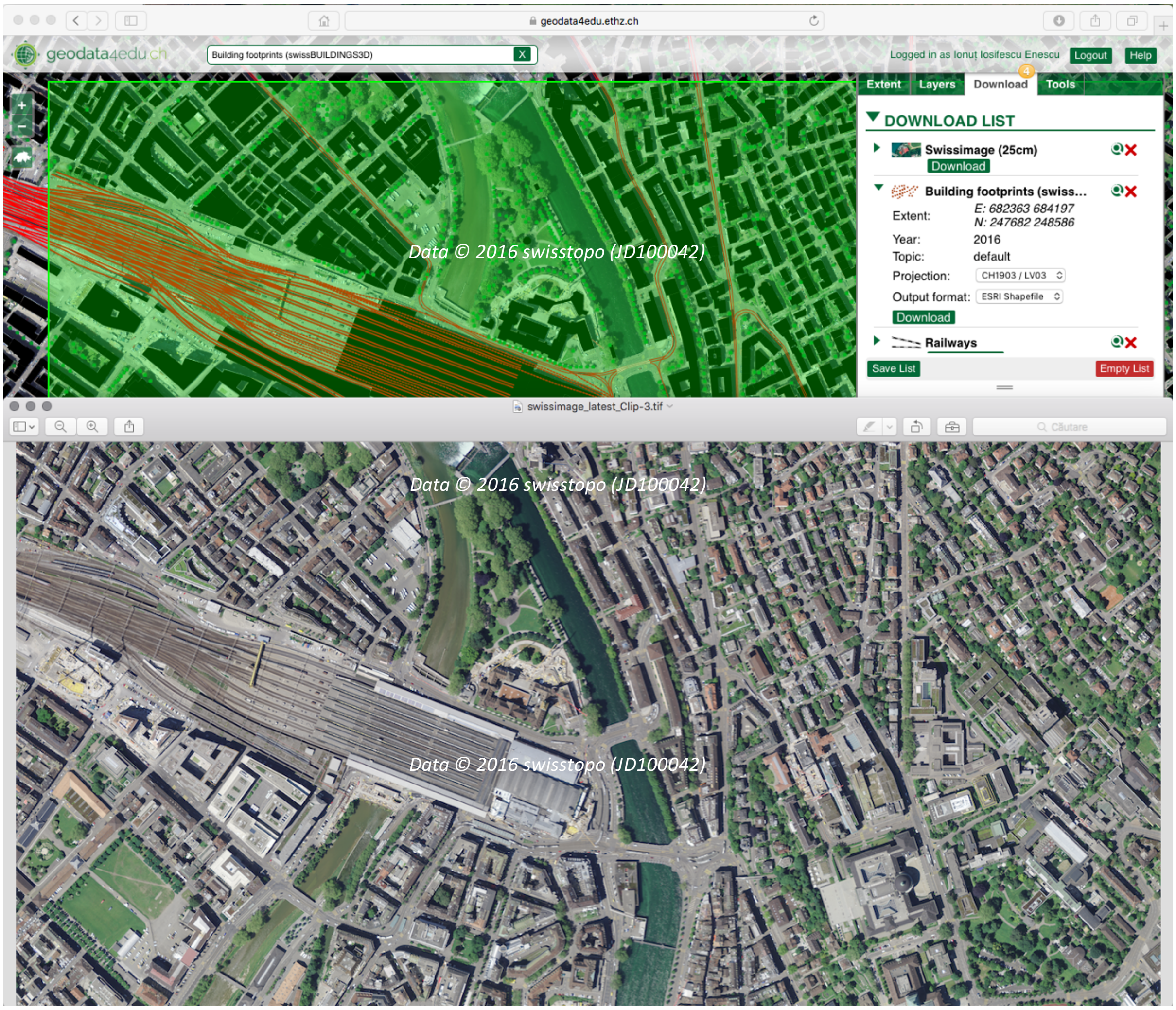


After the download extent has been defined, the user can add the active layers to the “Download List”, either one by one or by pressing the “Get All” button.



In the "Download List" tab, the user can simply accept the default data download settings and simply start the extraction process for each layer. Alternatively, the settings for each individual download can be changed in case the user would prefer a different projection or output format.

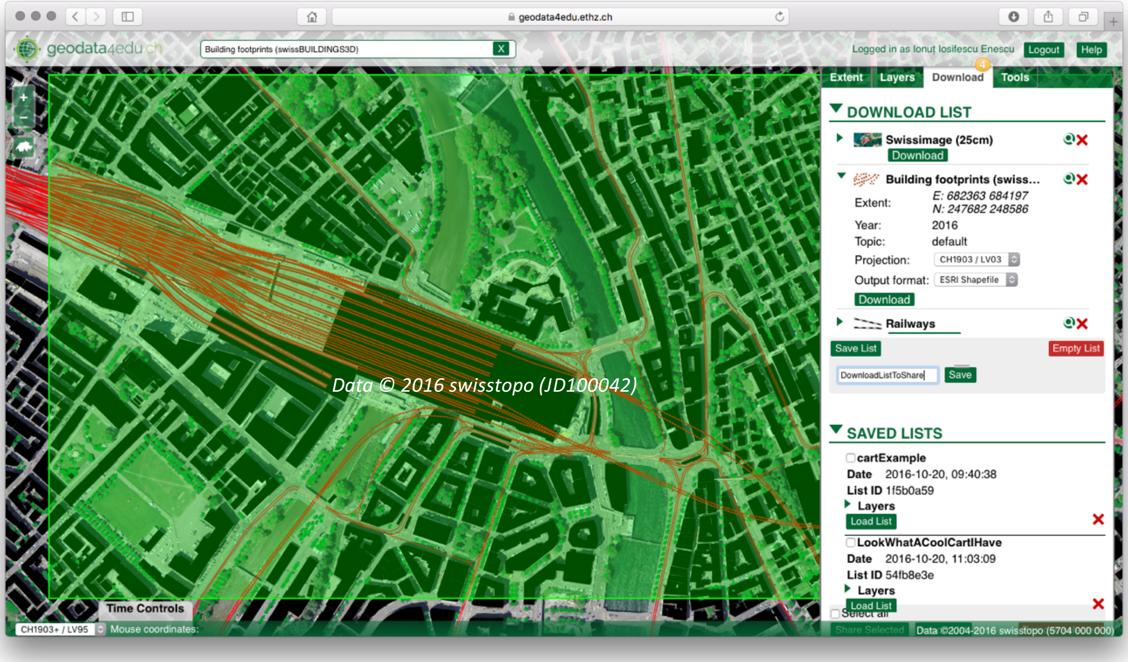
After the download process is complete and the download datasets have been downloaded on the user's computer, the data (e.g. swissimage in the next illustartions) can be opened with a variety of GIS systems (e.g. QGIS, ESRI ArcMap) and non-GIS software (e.g. CAD, graphic design, image editing software).



Note: the illustrated download procedure can be performed only by employees and scientific staff.

Due to legal reasons, students are currently not allowed to download data from GeoVITe by themselves. However, students can view and select the needed data, save the download list and share that list with their supervisor, who assumes the legal responsibility for downloading and disseminating the data to them.

Any GeoVITe user (not just students) have the possibility to save their different download lists so they can be reused later.



The saved download lists can be selected and shared with any other GeoVITe user, by entering their primary AAI email address (the user email address reported by SWITCHaai). These shared lists will automatically become visible to the target users (e.g. supervisors) at their next login.