



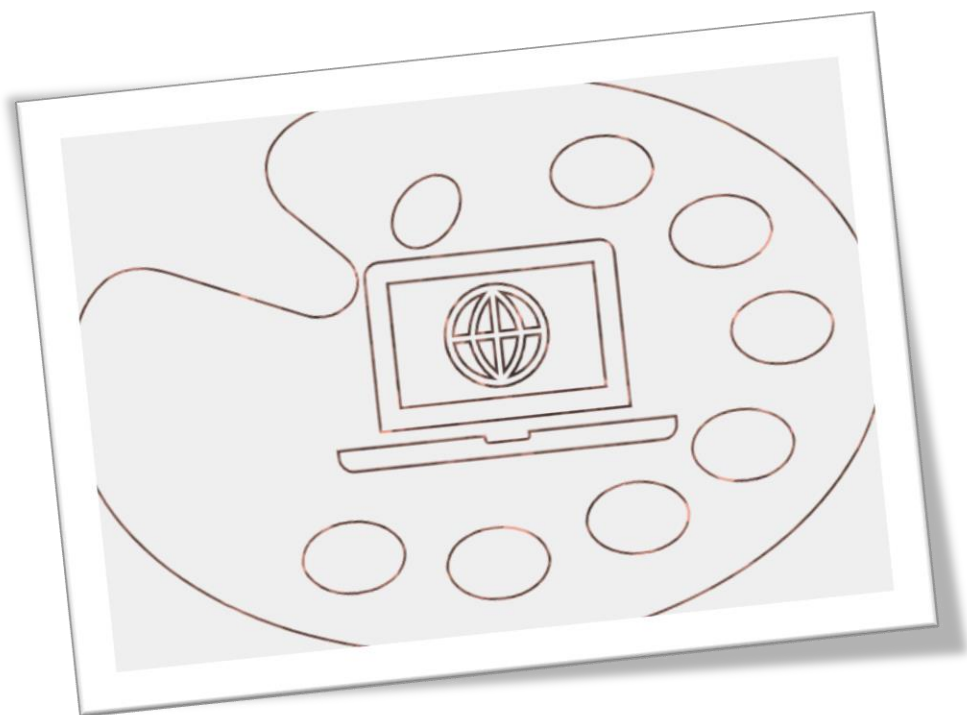
Central Asia
CLIMATE PORTAL

CACIP Platform

System concepts design

Back-end and interoperability structure

Graphical interface



Project: Central Asia Regional Climate Information Platform



The main objective is the development a Central Asia Regional Information Platform which will help stakeholders to access, analyze, and visualize public-domain data to support improved awareness, assessment, and decision support. This is expected to make available comprehensive and up-to-date relevant data and information, linking with high-quality datasets (including time series and spatial information) from global, regional, and local sources, provide analytical tools and interfaces for the visualization and interpretation of data and information (e.g. mapping tools to layer data and map hotspots and areas at risk, screening tools, etc.).

For more information, please visit:

<https://mel.cgiar.org/projects/cacip>
www.CentralAsiaClimatePortal.org

AUTHOR:

Simone Maffei

CO-AUTHORS:

Jim Jaspe, Aya Mousa, Budi Hermansyah, Samuel Stacey, Enrico Bonaiuti, Chandrashekhar Biradar

SUGGESTED CITATION

Simone Maffei, Jim Jaspe, Aya Mousa, Budi Hermansyah, Samuel Stacey, Enrico Bonaiuti and Chandrashekhar Biradar (21/11/2019). CACIP Platform – System concepts design, back-end and interoperability structure, graphical interface. International Center for Agricultural Research in Dry Areas (ICARDA): Beirut, Lebanon.

DISCLAIMER



This document is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit

<http://creativecommons.org/licenses/by-sa/4.0/>.

Unless otherwise noted, you are free to copy, duplicate, or reproduce and distribute, display, or transmit any part of this publication or portions thereof without permission and to make translations, adaptations, or other derivative works under the following conditions:



ATTRIBUTION. The work must be attributed, but not in any way that suggests endorsement by the publisher or the author(s)



SHARE ALIKE. If this work is altered, transformed, or built upon, the resulting work must be distributed only under the same or similar license to this one.

Photo Credit: ICARDA

Contents

INTRODUCTION	4
GENERAL OVERVIEW AND APPROACH	5
OVERVIEW OF THE SYSTEM	5
SYSTEM REQUIREMENTS	6
BUSINESS PROCESSES	6
ASSUMPTIONS	7
CONSTRAINTS	7
RISKS	8
DESIGN CONSIDERATIONS	10
<i>User heterogeneity</i>	<i>10</i>
<i>Heterogeneous technological environment</i>	<i>11</i>
<i>Moving from a local to a regional perspective</i>	<i>11</i>
GOALS AND GUIDELINES	12
Goals	12
Guidelines	13
DEVELOPMENTS METHODS AND CONTINGENCIES	13
SYSTEM ARCHITECTURE	15
THE SPATIAL COMPONENT (GEOPORTAL)	17
Architecture	17
Web framework	18
GeoNode basic external components and software	18
HARDWARE ARCHITECTURE	20
SOFTWARE ARCHITECTURE	20
The MetaData Harvester (MDH)	21
The front-end of the spatial component (GeoPortal)	22
The back-end of the spatial component (GeoPortal)	24
Special tools	28
INFORMATION ARCHITECTURE	29
Proposal for processing of new data (MODIS derived maps)	31
INTERNAL COMMUNICATION ARCHITECTURE	33
Geo-Portal	33
INTEROPERABILITY	33
GeoNode component	33
INTERNAL AND EXTERNAL RELATIONSHIPS	34
SYSTEM ARCHITECTURE DIAGRAM	34
SYSTEM DESIGN	35
DATABASES DESIGN	35
FILES AND DATABASES STRUCTURE	36
CACIP database	36
GeoNode and the Spatial component	37
GRAPHICAL INTERFACE DESIGN	38
SITE MAP	40



CACIP Platform

System concepts design

Back-end and interoperability structure

Graphical interface

Introduction

The document describes in details of the Central Asia Climate Information Platform.

A summary and main objectives of the initiative, and the surrounding general context introduce the platform, and provide the basis to understand the architectural and design aspects.

The general approach to the development methodologies used is explained.

System architecture is the skeleton and the high level infrastructure of the software, and describe how to translate in a software solution the design principles of the project (sustainable, long term provision of public domain information and data: accessible on digital devices, maximizing the use of existing, accessible information, and available institutional infrastructure, facilitation of in-country, regional and international cooperation, easily accessible by end-users). In this section the hardware infrastructure is described; the architecture is divided in back-end and front-end components. Also, the relationships between the system and other external systems are explained, and the **interoperability** solutions are described.

System design is a code level description of how each software module works. It includes all technical solutions adopted in the code and details each module.

The graphical solutions adopted for the information platform are described in the **Graphical interface** section. Some sample images provide a realistic overview of how the platform looks like



General overview and approach

Instructions: Briefly introduce the system context and organization. Provide a brief overview of the system and software architectures and the design goals. Include the high-level context diagram(s) for the system and subsystems

The Climate Adaptation and Mitigation Program for Aral Sea Basin (CAMP4ASB) aims to enhance regionally coordinated access to improved climate change knowledge services for key stakeholders (e.g., policy makers, communities, and civil society) in participating Central Asian countries as well as to increased investments and capacity building that, combined, will address climate challenges common to these countries.

In this context the main objective of the CACIP project is the development a Central Asia Regional Information Platform which will help stakeholders to access, analyze, and visualize public-domain data to support improved awareness, assessment, and decision support. The information platform is expected to make available comprehensive and up-to-date relevant data and information, linking with high-quality datasets (including time series and spatial information) from global, regional, and local sources, provide analytical tools and interfaces for the visualization and interpretation of data and information (e.g. mapping tools to layer data and map hotspots and areas at risk, screening tools, etc.).

Overview of the system

Instructions: A brief functional description with key concepts. Provide a top-level description of the system and its major external interfaces to aid the reader in understanding what the software is to accomplish. Reference appropriate graphics, illustrations, tables etc. to show functions.

- In what environment it works
- Who the users are
- What it is for
- The main functions
- The main interfaces, inputs and outputs

Climate-related platforms are becoming abundant globally, climate databases are already becoming available for each location in the world, including Central Asia, containing historical observation and projections of future climate. The approach guiding this particular information platform is to translate those databases into information and knowledge through facilitating appropriate interpretation by designing user-friendly and high-resolution suite of tools. The proposed climate information platform will focus on designing and developing an integrated tool primarily based on multi-source open access data and satellite earth observations.

The target users span several sectors:

- decision makers (both in public and private sectors) need updated and reliable information for planning and administration purposes
- national meteorological organizations can benefit from a portal where they can share and compare their data with the agencies of the other countries
- scientists and researchers can benefit from finding, in one place, the collection of all resources and data related to climate in Central Asia.
- Individuals who suffer the effects of climate change and need information, technologies, procedures to mitigate the impact on their activities and to improve the results of their initiatives



- donors need data and statistics for their initiatives
- international and national organizations, working on projects in support of Central Asia
- citizens, always more involved in climate change related phenomena

The system is a multi-scale data inventory, especially for knowledge about climate, biophysical, socioecological, land-use and land cover, and includes near real-time spatiotemporal information collected by the satellites.

All information will be available through an information platform including advanced repository features, with full standardized management of the metadata, with harvesting capability and interoperability.

System requirements

Instructions: Include a description of each major requirement in narrative and include a table of the mapping of requirements to modules and current status: These should include significant functional workload and functional performance, operational, technical, security and any other special requirements.

Project definition

The system is expected to harvest information and from external websites, locally store information and data that is otherwise not harvestable, display them for comprehensive knowledge and allow printing, downloading and accessibility from external systems.

Information and data

The system is expected to be able to collect already existing information and data, freely distributed and in the public domain.

Accessibility

The system is expected to follow the responsive design principles and to be accessible on all digital devices (computers, tablets, smartphones). Where possible, the information must be printable to allow a dissemination of the contents using non-digital means.

Languages

The system is expected to be developed in Russian and English.

Knowledge Sharing

The System has a built-in discussion forum linked with research outputs and deliverables to share opinion and improve quality.

Business processes

Instructions: Briefly describe the various business processes at a system overview level; descriptions of the specific business process will be tackled later in this document.

The overall business process of the system is collecting, structuring and sharing information and data (geographical and non-geographical). The system includes monitoring component that tracks the amount of available information/data, and produces statistics on the uses, interactions and performances of the platform.

Assumptions

Instructions: Describe any assumptions or dependencies regarding the system and its use. These may concern such issues as related software or hardware, operating systems, end-user characteristics, and possible and/or probable changes in functionality

Beside of the usual assumptions on user, system administrators and developers' prerequisite knowledge to be able to operate with the platform, other specific assumptions are listed below.

Information and data availability

The system is not supposed to produce new data, but it assumes that institutions, agencies and other subjects provide information and data to the platform. Accessibility based on interoperable interfaces such as APIs or web services is a priority.

System hosting

It is assumed that a local institution, acting at regional level, and able to ensure the needed technical activities to keep it operational and maintain the platform would take charge of it. This institution must be able to guarantee a suitable informatic infrastructure, according with the system requirements, and suitable system admin and administrative skills and commitments.

Constraints

Instructions: Describe any limitations or constraints that have a significant impact on the system, software and/or communications, and describe the associated impact. Such constraints may be imposed by any of the following (the list is not exhaustive):

- a) Hardware or software environment
- b) End-user environment
- c) Availability or volatility of resources
- d) Standards compliance
- e) Interoperability requirements
- f) Interface/protocol requirements
- g) Licensing requirements
- h) Data repository and distribution requirements
- i) Security requirements (or other such regulations)
- j) Memory or other capacity limitations
- k) Performance requirements
- l) Network communications
- m) Verification and validation requirements (testing)
- n) Other means of addressing quality goals
- o) Other requirements described in the Requirements Document

Hardware or software environment

The system run on a server with a stable and secure operating system (such as one of the hardened Linux distributions) and 64 bits hardware is strongly recommended.

The system architecture is based on widely used, stable and tested Open Source software, thus no specific constraints are found.

End-user environment

A basic compatible browser with JavaScript is needed.



Availability or volatility of resources

No limitations, nor constraints are found.

Standards compliance

No limitations, nor constraints are found.

Interoperability requirements

No limitations, nor constraints are found.

Interface/protocol requirements

No limitations, nor constraints are found.

Licensing requirements

No limitations, the platform is based on Open Source software and it does not require any commercial license.

Data repository and distribution requirements

No limitations, nor constraints are found.

Security requirements (or other such regulations)

No limitations, nor constraints are found.

Memory or other capacity limitations

No limitations, nor constraints are found.

Performance requirements

No limitations, nor constraints are found.

Network communications

No limitations, nor constraints are found.

Verification and validation requirements (testing)

No limitations, nor constraints are found.

Risks

Instructions: Describe any risks associated with the system design and proposed mitigation strategies

Data Sharing Resistance

The system relies on the availability of the data from partners such as stockholders, universities, governments etc. The system does not include data creation but processes external data. The resistance to data sharing might put the system in a scenario where it has functionality with no data to serve.

The support for Central Asia community (governments, universities, stockholders, etc.) is one of the success factors for this project.

No control on the accuracy of collected information and data

During loading activities (manual upload and/or harvesting) the system collects additional metadata, useful to categorize the information. If the minimum set of metadata is not provided, the information is not added to the system database,



however no quality control is performed on the content and accuracy of the information.

The system includes a "social" evaluation method based on user rating. Users can provide a rate to each content item, and a summarized score is calculated by the system: no check on the rating is performed.

Possible loss of harvested information/data

Harvested data are not duplicated on the system. The system stores only some metadata and the link to the original data. If the external source should fail, information and data harvested from that source will not be available anymore.

Avoiding the duplication of information is one of the requirements of the project, thus no mitigation measures are feasible.

Sustainability of the system

The system relies on the capability of the hosting institution to collect the funds to keep operational and maintain the system itself. The system does not include self-financing mechanism, thus relying on utility and providing useful services to the community to raise interest around itself and promote financing solutions.

Design considerations

Instructions: Describe issues, which were addressed or resolved when designing the system.

The objective of the system is to develop a platform useful to a heterogeneous community of users located in a heterogeneous social environment.

User heterogeneity

Issue

Target users are:

- government policy makers, ministries
- public agencies (hydro-meteorological observations/services, water management, agriculture, disaster prevention and response, natural resources management, etc.)
- state and public research institutions and agencies
- academia, researchers, training institutions, information dissemination centers
- public and private subjects implementing and financing Climate Change mitigation and adaptation projects:
- resource user associations
- farmer associations, individual farmers, shepherds cooperatives, commercial farms, and smallholdings
- commercial sector, including insurance companies, design organizations, and banks
- civil society organizations
- local communities (provincial/district/village administrations and individual households, water user associations)
- relevant regional organizations
- international donors and expert community

Target users have **different needs**, and they are interested in **different types of information and data**.

Design action proposed: user categorization based on users profiles

From the analysis of the stakeholders and of the potential users of CACIP, the following main profiles have been Identified:

- citizens
- trainers (advisers)
- researchers (scientists, experts, ad also students)
- farmers
- insurances, energy companies and other private companies
- investors
- policy makers
- data/knowledge providers

Taking into account the contents, data, and tools provided through the CACIP, these profiles have been aggregated in the following categories:

- citizen



- trainer
- researchers
- farmers
- decision makers (including policy makers, private companies, investors)

These five categories are the ones used to group the portal users: the “role” is one of the attributes saved for each registered user, and the previous categories are the values allowed for this attribute. This is an important attribute and it can be used in the portal to tailor the provided contents.

From the user profiling point of view, “data/knowledge providers” are not a special category, and they can belong to each listed category.

Heterogeneous technological environment

Issue

The geographical area covered by the information provided by the system is also characterized by a heterogeneous distribution of communication infrastructure:

- internet coverage is generally reasonable, however could be limited in some areas and speed could be low
- smartphones are common in the region, usage for information access varies depending for young/old generation, internet cost/access
- simpler communication channels have a heterogeneous coverage (WhatsApp in Kazakhstan, Telegram in Uzbekistan, IMO in Turkmenistan, IMO and Viber in Tajikistan)

Moving from a local to a regional perspective

Issue

One of the priorities of the system is to facilitate the creation of a network to:

- increase the collaboration of local, national, regional and international subjects
- support moving from a local to a regional perspective
- encourage a regional awareness

This is a challenge, because of the simplified tagline “think regionally” and a resistive attitude to sharing information and data outside the traditional silos (organization, county etc.): some previous initiatives have had difficulties at this point, and have been curtailed by this problem.

Goals and guidelines

Instructions: Describe any goals, guidelines, principles, or priorities, which dominate or embody the design of the system and its software. Examples of such goals might be an emphasis on speed versus memory use; or working, looking, or “feeling” like an existing product. Guidelines include coding guidelines and conventions. For each such goal or guideline, describe the reason for its desirability unless it is implicitly obvious.

Goals

The goals of the system derive from the initial project requirements, integrated by the suggestions and feedback of the stakeholders, collected during the regional and national official workshops, and during several informal face to face meetings.

Sustainability and long-term services

- To be able to guarantee the sustainability and durability of the system, the following principles have to be taken into account while designing the system itself:
- **long-term provision of free, public-domain climate information** (without information and data continuously updated, the system loses interest and cannot survive)
- **minimizing cost of O&M** (an expensive infrastructure has minimal chances to survive)
- **involvement of the community** (the more people involved in the system, as user, as contributor, as supporter etc., the more the opportunities to stimulate a continuous interest and funding of the initiative).

Re-use

- **maximize use of existing information, knowledge, data** (the system is a container fed with already existing data; the creation of new content is not one of the goal of the system, but the grouping of all available data in one structured place, featuring tools to easily browse, search, display, download, print, analyze and share)
- **maximize use of existing infrastructure** (the system must be hosted in an existing local structure working at regional level)

Network approach

- facilitation of in-country, regional, international cooperation and information sharing: this goal can be satisfied by
 - designing tools to integrate information and data at a regional level
 - making easy exchanging data and information
 - ensuring the security of shared data
 - prioritizing the guarantee of the ownership of shared data
 - prioritizing the visibility of the sources of the data

Accessibility

- **accessibility from various digital devices** (a **responsive interface design** ensures the correct visualization of the information and data in screens of different size, in computers, tablets, smartphones; **development based on**



standards and widely used libraries guarantees a coherent browsing experience on every device)

- **easy linkage to modern decision support systems** (interoperability based on standards is one of the pillars of the system)
- **delivery of information in analysis-ready format** (the system prioritizes the provision of “interpreted” information)
- **support for off-line knowledge products** (e.g. the system includes contents already easy-printable as well as friendly printing tools are included)
- **information contents:** the portal is mainly based on harvested contents and data; from the analysis of available information derived from the websites and stakeholders inventories, the main topics:
 - Climate Change, Adaptation, Mitigation
 - Water Management
 - Risk Assessment
 - Food Security
 - Sustainable Agroecosystems
 - Land Degradation

Guidelines

The logical architecture of the system has been structured in three main modules:

Website

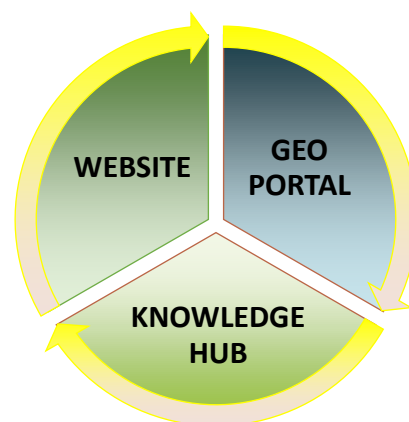
The **website** is the entry point of the system, from where the user can login, select the language, search information, visualize news, tweets, update, documents, dashboards, access the specific sections of the GeoPortal and of the Knowledge Hub.

GeoPortal

The **GeoPortal** collects, manages and displays geographical data and includes analysis tools.

Knowledge Hub

The **Knowledge Hub** collects, stores and provides docs, papers, training materials, media, contacts, and all relevant information.



Developments methods and contingencies

Instructions: Briefly describe the method or approach used for the system and software design (e.g., structured, object-oriented, prototyping, J2EE, UML, XML, etc.). If one or more formal/ published methods were adopted or adapted, then include a reference to a more detailed description of these methods. Describe any contingencies that might arise in the design of the system and software that may change the development direction. Possibilities include lack of interface agreements with outside agencies or unstable architectures at the time. Address any possible workarounds or alternative plans.

The system is oriented to harvest information and data from multiple sources. For documents and knowledge base this approach is already applied by several portals,



and the exchange interfaces (interoperability) are widely used. For spatial and technical data, the task is not as common.

Spatial and technical data have an **intrinsic complexity** that sometimes make a full interoperability a hard task. The “amount” of information potentially included in a spatial dataset can be huge, the use of data may require the application of geographical algorithms (for example to synchronize different coordinate systems), graphical style (to let information be more understandable), numerical aggregation (for example to translate the row numbers collected by a sensor and stored in a geo dataset, into categories or classes). Thanks to diverse application of standard formats and structures, data are exchangeable, but sometimes not understandable.

For these reasons, the approach to such a spatial and technical data requires **the development of specific interface and procedures**. These procedures are very often valid only to “one” source, and they are not applicable in general; then for each kind of source and type of data, specific routines must be developed. This issue has to be taken into account and can make some functionalities work only for specific set of data: the extension of applicability could require in the future additional development activities.

Even the most carefully planned project can run into trouble. No matter how well planned, projects can always encounter unexpected problems. In regards to CACIP, we are trying to fortify the solution by using standard and widely used engines and applications:

- **GEONODE** for the management of spatial data, an open source geospatial product that provides out-of-the-box components and functionalities to create a complete Spatial Data Infrastructure to ingest, style, browse and search, visualize, disseminate, geospatial data; GeoNode plays a key role for implementing interoperable Spatial Data Infrastructures as it provide data and metadata management as well as rapid mapping capabilities to complement GeoServer functionalities

System architecture

Instructions: Describe the system architecture, how the application interacts with other applications. Not necessarily how the application itself works but, how the appropriate data is correctly passed between applications. Provide an overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Don't go into too much detail about the individual components themselves in this section. A subsequent section of the System Design Document (SDD) will provide the detailed component descriptions. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

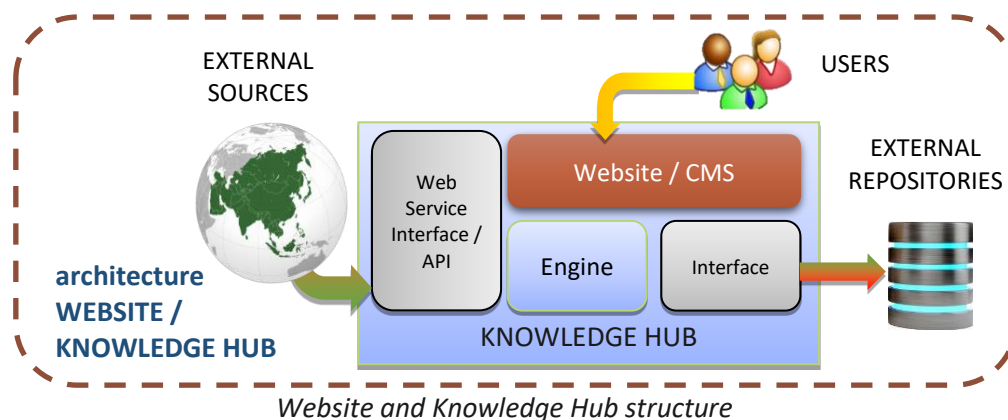
At the top-most level, describe the major responsibilities that the software must undertake and the various roles that the system (or portion of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem and the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results. Provide some sort of rationale for choosing this particular decomposition of the system.

Make use of design patterns whenever possible, either in describing parts of the architecture (in pattern format), or for referring to elements of the architecture that employ them. Provide rationale for choosing a particular algorithm or programming idiom (or design pattern) to implement portions of the system's functionality.

The system is based on three main components: Website/CMS, Knowledge Hub, GeoPortal.

These components are linked each other, but there are also links from and to external systems.

The next graphic shows the internal structure of the two components **Website** and **Knowledge Hub**, and the links to the external entities:



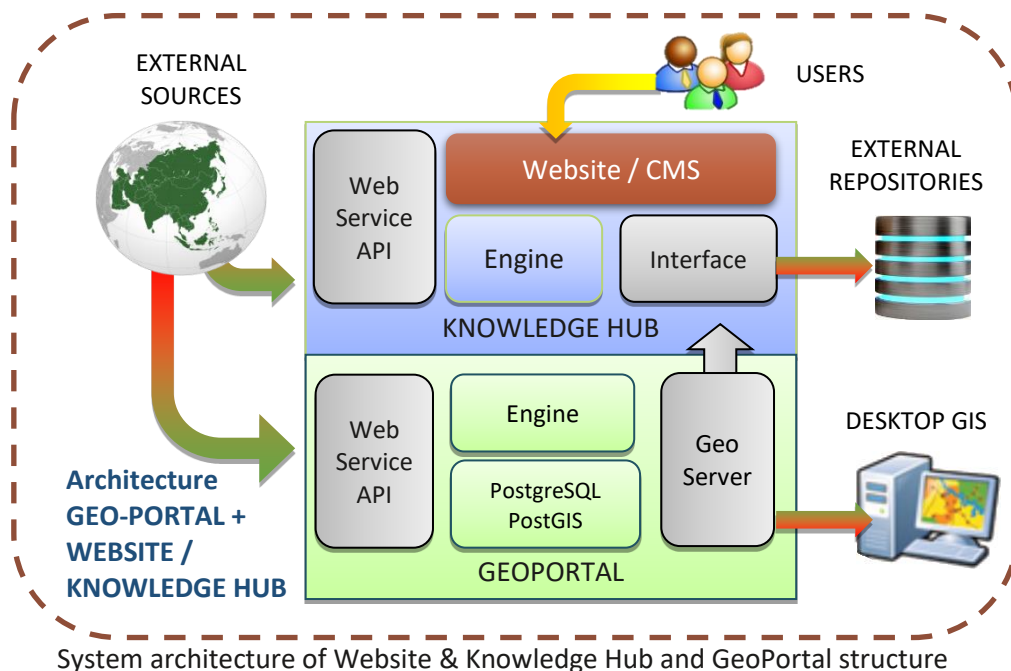
- the **external sources of information**, through specific procedures based on the interoperability facilities provided by these external sources: during the websites inventory, this topic has been checked in detail
- the **external repositories** are the external systems that potentially could harvest data from the CACIP system: data exchange interfaces are a core sub-component of the system both for input and output

- the **users** access the system through the website, and from here they are able to navigate all the contents

The Geo-Portal is the third component of the system.

It is designed as a separate component, but it is linked to the website/CMS/Knowledge Hub to allow the user a “transparent” navigation of the platform. Due to its complexity, it has its own CMS and its own interfaces.

The modular design of the full system can make easier the activity of maintenance and upgrade, by giving the opportunity to involve in the development very specialized developers (to whom a full skill on all the modules is not required).



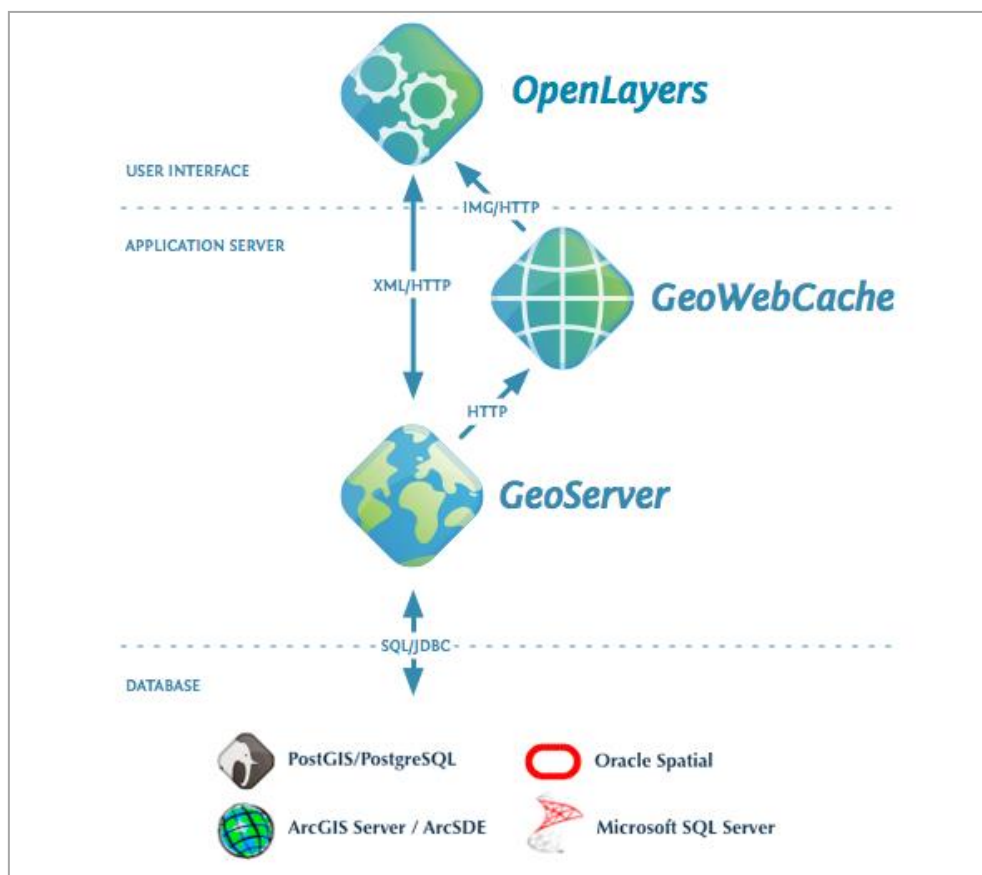
The graphic above shows the full internal structure of the components **Website**, **Knowledge Hub** and **GeoPortal**, and the links to the external entities. In addition to the ones listed before:

- the **desktop GIS** can embed directly the data coming from the system: indeed GeoNode (the core of the GeoPortal, implements many Open Geospatial Consortium (OGC) standards, including Web Map Service (WMS), Web Feature Service (WFS), Web Coverage Service (WCS), KML, and Catalogue Service for Web (CSW)

The spatial component (GeoPortal)

The GeoPortal core is built on top of GeoNode (more detailed information is available [here](#)): a geospatial content management system, a platform for the management and publication of geospatial data.

Architecture



GeoNode is an open source framework designed to build geospatial content management systems (GeoCMS) and spatial data infrastructure (SDI) nodes. Its development was initiated by the Global Facility for Disaster Reduction and Recovery (GFDRR) in 2009 and adopted by large number of organizations in the following years.

GeoNode is built upon a platform of proven open source components including Django, GeoServer, pycsw, OpenLayers and GeoExt. GeoNode implements many Open Geospatial Consortium (OGC) standards, including Web Map Service (WMS), Web Feature Service (WFS), Web Coverage Service (WCS), KML, and Catalogue Service for Web (CSW). GeoNode development also contributes to the underlying open source projects and software libraries. Features include:

- PostGIS Spatial Databases
- GeoServer OGC Services
- pycsw CSW Metadata Catalogue
- Geospatial Python libraries
- OpenLayers and GeoExt Web Mapping Libraries



Web framework

GeoNode leverages the Django web framework, offering a friendly and extensible environment for web developers accustomed to using any modern MVC web framework. In contrast with many geospatial tools which force web developers to adapt to a GIS context, GeoNode and Django offer a framework on which developers can build geospatial web applications in a familiar environment.

Django has a large user base and ecosystem of apps, making it easy to develop websites focused on users and collaboration. Features Include:

- Modern web framework
- Pluggable GeoNode Django apps
- Deployable template projects
- Integrated admin interface
- Large ecosystem of pluggable apps
- Spatial GeoDjango object-relational mapping available

Using an open source stack based on mature and robust frameworks and software like those on which GeoNode is based (Django, OpenLayers, PostGIS, GeoServer and pycsw) an organization can build on top of GeoNode its SDI or geospatial open data portal.

GeoNode basic external components and software

GeoNode Architecture is based on the cited set of core tools and libraries that provide the building blocks on which the GeoNode application is built. The following sections explain each of these components.

Django

The user interface of a GeoNode site is built on top of the Django web framework, which is a high level Python web development framework that facilitates rapid development and clean design. GeoNode includes a few “apps” (reusable Django modules) to support development of those user interfaces. While these apps have reasonable default configurations, which can be adjusted to suit specific needs. The apps provide HTTP proxies for accessing data from remote servers, to overcome restrictions imposed by the same-origin policy used by browsers. This helps the GeoExt applications in a GeoNode site to access various XML documents from OGC-compliant data services.

GeoServer

GeoServer is an open-source software for servers written in Java that provides OGC compliant services, which publish data from many spatial data sources. GeoServer is used as the core GIS component inside GeoNode that renders the layers, creates map tiles from the layers, downloads those layers in various formats, and allows for transactional editing of those layers. GeoNode utilizes GeoServer in following ways:

- GeoNode configures GeoServer via the REST API
- GeoNode retrieves and caches spatial information from GeoServer. This includes relevant OGC service links, spatial metadata, and attribute information
- GeoNode can discover existing layers published in a GeoServer via the WMS capabilities document
- GeoServer delegates authentication and authorization to GeoNode

- Data uploaded to GeoNode is first processed in GeoNode and finally published to GeoServer (or ingested into the spatial database)

GeoExplorer

GeoExplorer is a web application, based on the GeoExt framework, for composing and publishing web maps with OGC and other web-based GIS Services. GeoExplorer is used inside GeoNode to provide many of the GIS and cartography functions that are a core part of the application.

PostgreSQL and PostGIS

PostgreSQL is an SQL-supporting successor to the Postgres DBMS, which was developed up until 1993 at the University of Berkeley. It is a free, open-source, object-relational DBMS and provides some geometrical data types as well as the R-tree as spatial index. However, this implementation does not fulfill the OGC specification and full spatial database functionality is achieved by the PostGIS extension, which is built on top of PostgreSQL.

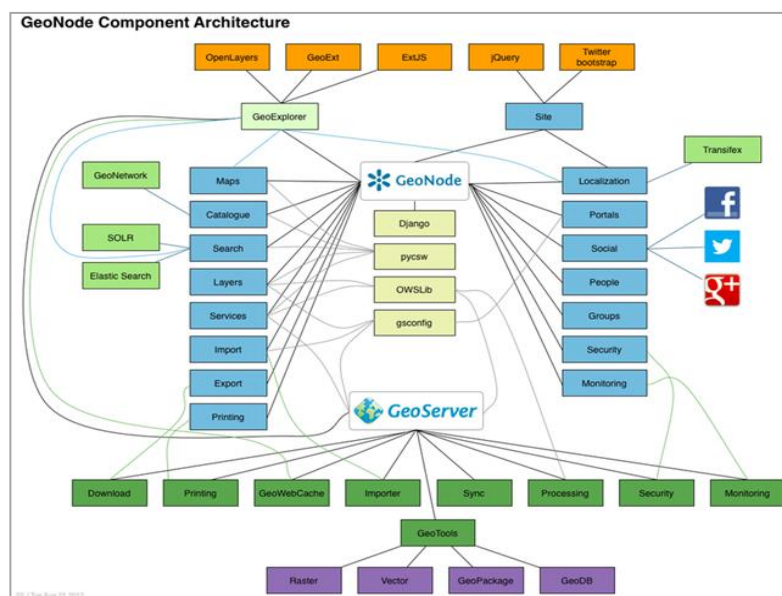
GeoNode uses PostgreSQL and PostGIS for storing and managing spatial data and information. All the tables and data are stored within a GeoNode database in PostgreSQL, whereas GeoServer uses PostGIS to store and manage spatial vector data for each layer.

pycsw

pycsw is an OGC Catalogue Services for the Web (CSW) server implementation written in Python. GeoNode uses pycsw to provide an OGC compliant standards-based CSW metadata and catalogue component of spatial data infrastructures, supporting popular geospatial metadata standards such as Dublin Core, ISO 19115, FGDC and DIF.

Geospatial Python Libraries

GeoNode leverages several geospatial python libraries including gsconfig and OWSLib. gsconfig is used to communicate with GeoServer's REST Configuration API to configure GeoNode layers in GeoServer. OWSLib is used to communicate with GeoServer's OGC services and can be used to communicate with other OGC services.





Django Pluggables

GeoNode uses a set of Django plugins, which are usually referred to as pluggables. Each of these pluggables provides a particular set of functionalities inside the application from things like Registration and Profiles to interactivity with external sites. Being based on Django enables GeoNode to take advantage of the large ecosystem of these pluggables out there, and while a specific set is included in GeoNode itself, many more are available for use in applications based on GeoNode.

The development of custom Django plugins in GeoNode will enable the software to extend the functionalities to meet with requirements.

jQuery

jQuery is a feature-rich JavaScript library that is used within GeoNode to provide an interactive and responsive user interface as part of the application. GeoNode uses several jQuery plugins to provide specific pieces of functionality, and the GeoNode development team often adds new features to the interface by adding additional plugins.

Bootstrap

Bootstrap is a front-end framework for layout and styling the pages that make up the GeoNode application. It is designed to ensure that the pages render and look and behave the same across all browsers. GeoNode customizes bootstrap's default style and it is relatively easy for developers to customize their own GeoNode based site using existing Bootstrap themes or by customizing the styles directly.

Hardware architecture

Instructions: Describe the overall system hardware and organization, indicating whether the processing system is distributed or centralized. Identify the type, number, and location of all hardware components including the presentation, application, development and data servers and any peripheral devices (e.g., load balancers, SSL accelerator, CDN with a brief description of each item and diagrams showing the connectivity between the components along with required firewalls, ports. Include resource estimates for processor capacity, memory, on-line storage, and auxiliary storage.

The minimum server requirement for this software :

- Linux operating system (up-to-date and hardened)
- 16 GB of RAM.
- 2.2 GHz processor with 2 cores. (Additional processing power may be required for multiple concurrent styling renderings)
- 100 GB software disk usage.
- Additional disk space for any data hosted with GeoNode and tiles cached with GeoWebCache. For spatial data, cached tiles, and “scratch space” useful for administration, a decent baseline size for GeoNode deployments is between 300GB and 600GB.
- 64-bit hardware strongly recommended.

Software architecture

Instructions: Describe all software that is needed to support the system, and specify the physical location of all software systems. List such things as logical components (e.g., CSS in presentation layer, Controllers in application layer, MySQL connectors in data layer), database platforms, computer languages, utilities, operating systems, communications



software, commercial off-the-shelf (COTS) software, open source frameworks, etc., with a brief description of the function of each item and any identifying information such as manufacturer, version number, number and types of licenses needed, etc., as appropriate. Identify all Computer Software Configuration Items (CSCIs), Computer Software Components (CSCs) and Application Programming Interfaces (APIs) to include name, type, purpose and function for each; the interfaces, messaging, and protocols for those elements; and rationale for the software architectural design. Include software modules that are functions, subroutines, or classes. Use functional hierarchy diagrams, structured organization diagrams (e.g., structure charts), or object-oriented diagrams that show the various segmentation levels down to the lowest level. All features on the diagrams should have reference numbers and names.

Include a narrative that expands on and enhances the understanding of the functional breakdown. If necessary, describe how a component was further divided into subcomponents, and the relationships and interactions between the subcomponents. Proceed into as many levels/subsections of discussion as needed in order to provide a high-level understanding of the entire system or subsystem, leaving the details for inclusion in a later section of the SDD. Include data flow diagrams that conform to appropriate standards (e.g., Yourdon-Demarco conventions) and provide the physical process and data flow related to the logical process and data flow decomposed to the primitive process level (describing how each input is processed/transformed into the resulting output).

The MetaData Harvester (MDH)

Instructions: Describe the functionalities available to the user (search tools, language selector, customization tools, downloading and uploading features, etc.) and the software “not visible” to the user that makes the system working: back-end (for example scheduled functions to harvest information, procedures to pre-process data and make them available, etc.)

This module collects and stores the metadata of publications (journal, book, thesis, paper, report, book review, book chapter, newspaper article, educational material, presentation, video, audio), blogs, news, events, and spatial/statistical data. An example of the metadata collected and stored is: title, author, date published, thumbnail, blog content, article link, download link.

The availability of metadata from the external portals (sources) depends on the API provided by the portals themselves. Some APIs may provide comprehensive metadata (all above), while others may provide only a title and a direct link to the original source. MDH stores what is available from these APIs.

Product metadata are stored in then **CACIP Database**. Each product record include also additional inforamtion:

- topic as: a. Water Management, b. Climate Change, c. Risk Assessment, d. Food Security, e. Sustainable Agroecosystems, f. Land Degradation, g. other
- geo-coverage as: Kazakhstan, Kyrgyzstan, Tajikistan, Turkmenistan, Uzbekistan, and International
- type as: map layers, datasets, publications, vizualization, news, blogs

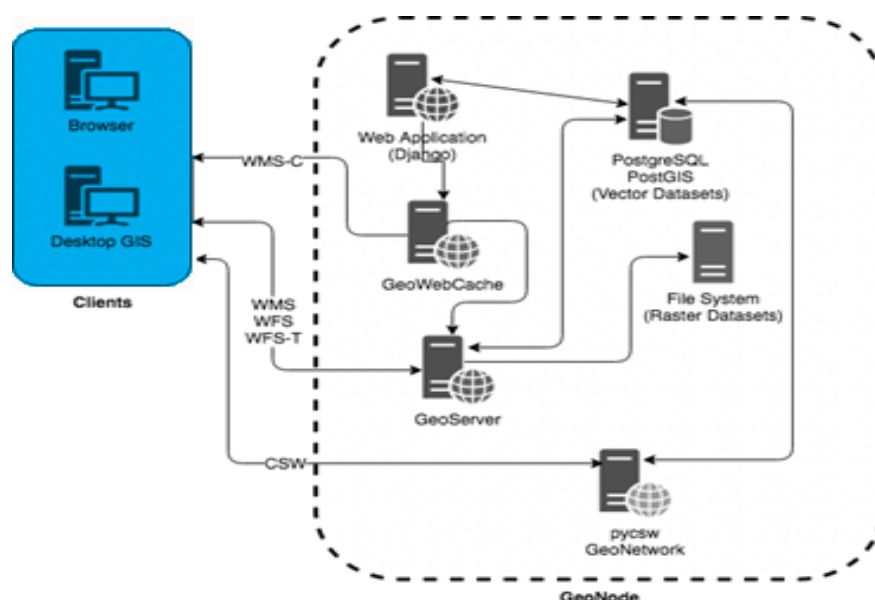
Storing metadata in this way will give the CACIP users the search-ability to filter and download the related contents (document, data, etc.) from its external resources (if needed, for example for paid contents, users will be re-directed to the original product site).

The front-end of the spatial component (GeoPortal)

Instructions: Describe the functionalities available to the user (search tools, language selector, customization tools, downloading and uploading features, etc.)

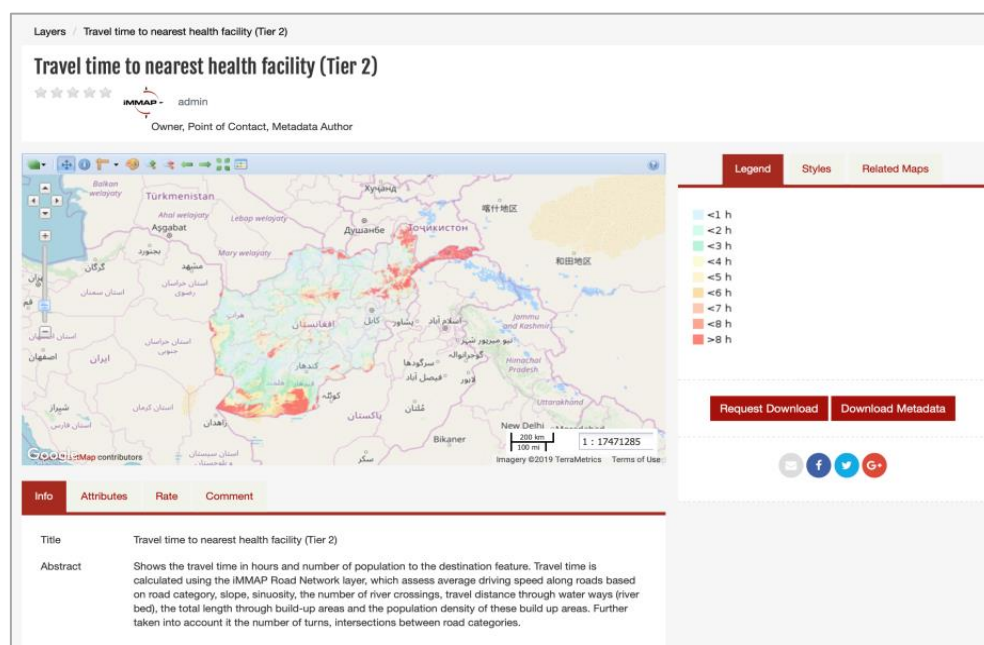
The GeoPortal is fully integrated in the global system, and specific tools are available for the user to access the spatial data contained in the system:

- from a **browser**
- from a **desktop GIS application**



Access through a browser

The **GeoExplorer** is a web application, based on the GeoExt framework, for composing and publishing web maps with OGC and other web-based GIS Services. GeoExplorer is used inside GeoNode to provide many of the GIS and cartography functions that are a core part of the application.

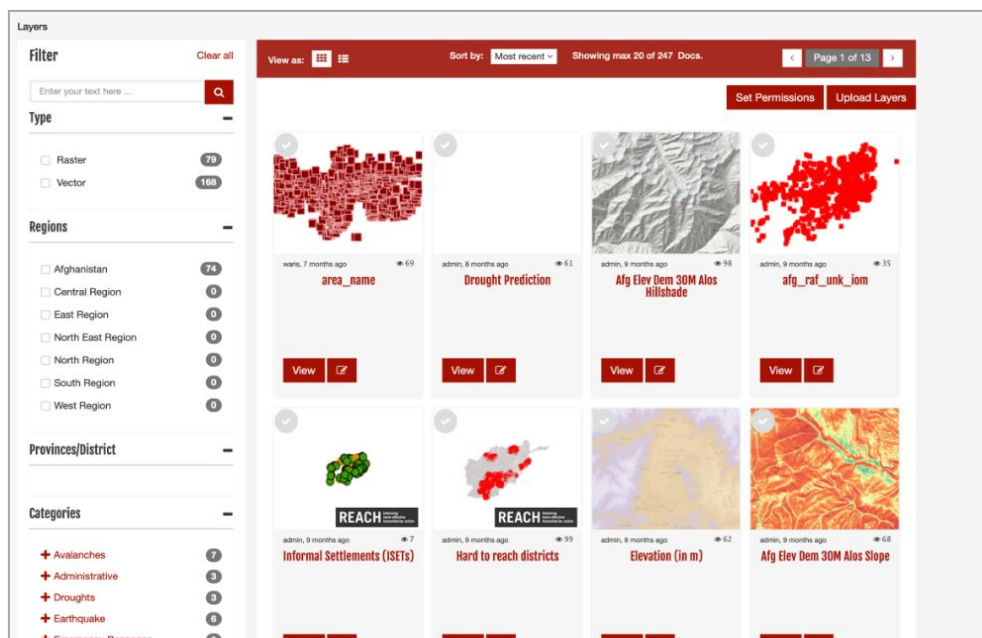


Display, navigate, query

GeoExplorer provides to the user all the functionalities to display, navigate, query the spatial data and the information linked to the geographical features, select different layers, overlap information, etc.

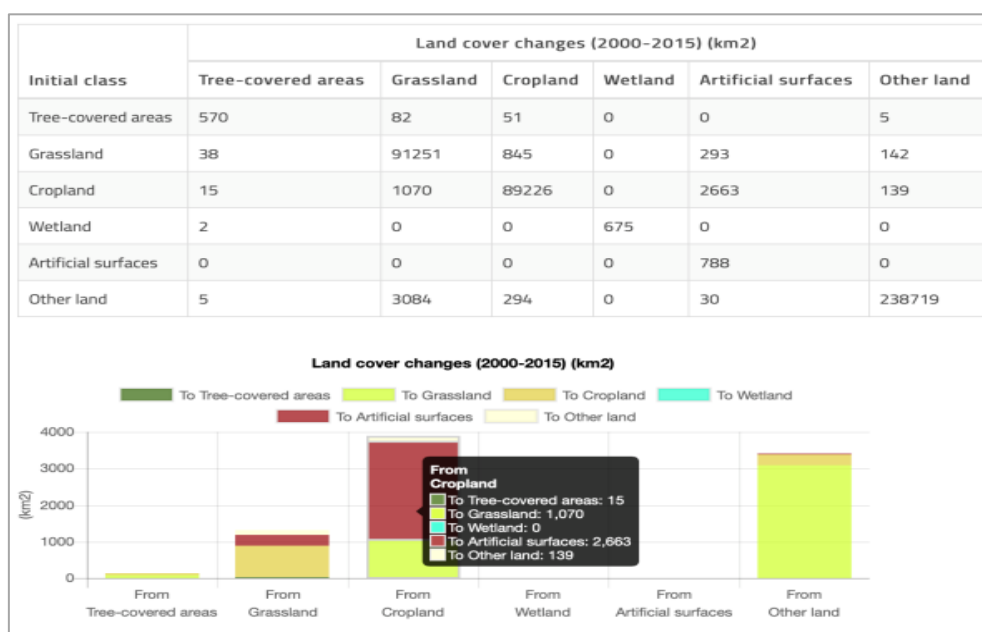
Browse and search

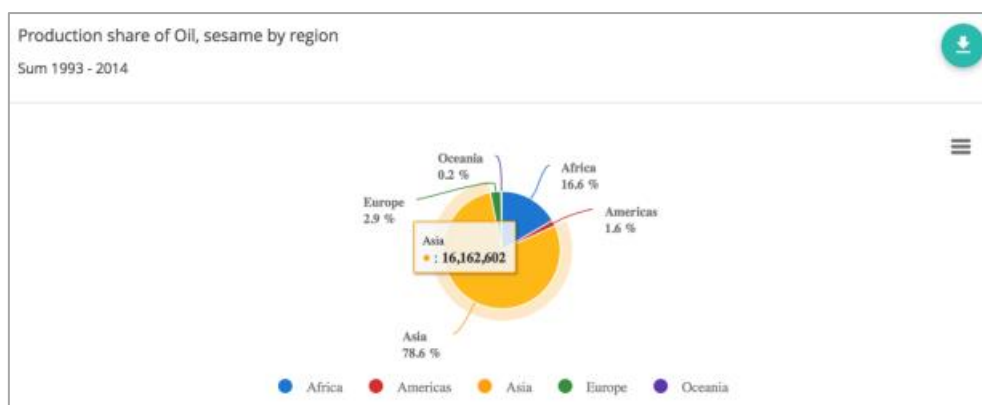
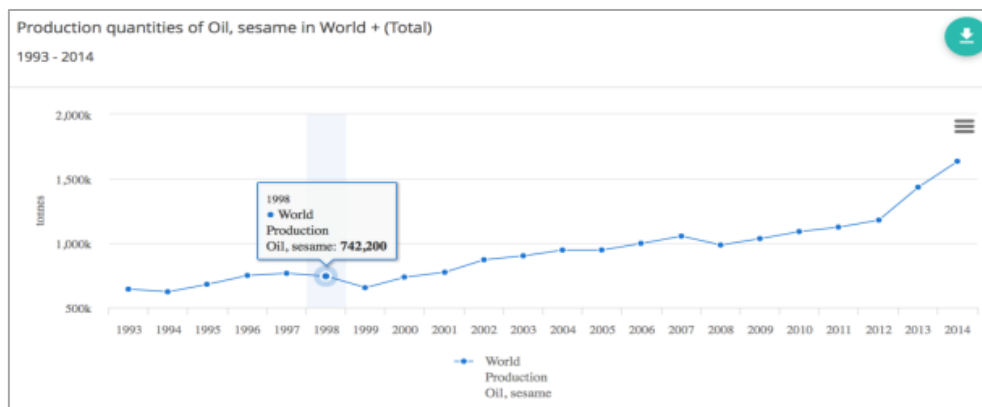
Specific user interfaces allow browsing and searching maps and layers:



Interactive tables

The following graphics show some capabilities of GeoNode allowing the interactive analysis of tabular data (in the example, charts are derived from the data of an attribute table).





How it is developed

GeoNode uses Django templates and a number of JavaScript libraries and frameworks in order to provide the user interface to the clients. Most notably, GeoNode uses AngularJS to let the users to access the Application Programming Interface (API) - which is based on Django Tastypie. The jQuery framework is also used widely in the context of the application, for example to provide words autocompletion in the search text boxes. The user interface includes a mapping client, which can be based on OpenLayers or Leaflet. Mapping clients are GeoExplorer, based on the OpenLayers, GeoExt and ExtJS JavaScript frameworks, which is the default and possibly the most common choice for GeoNode deployments.

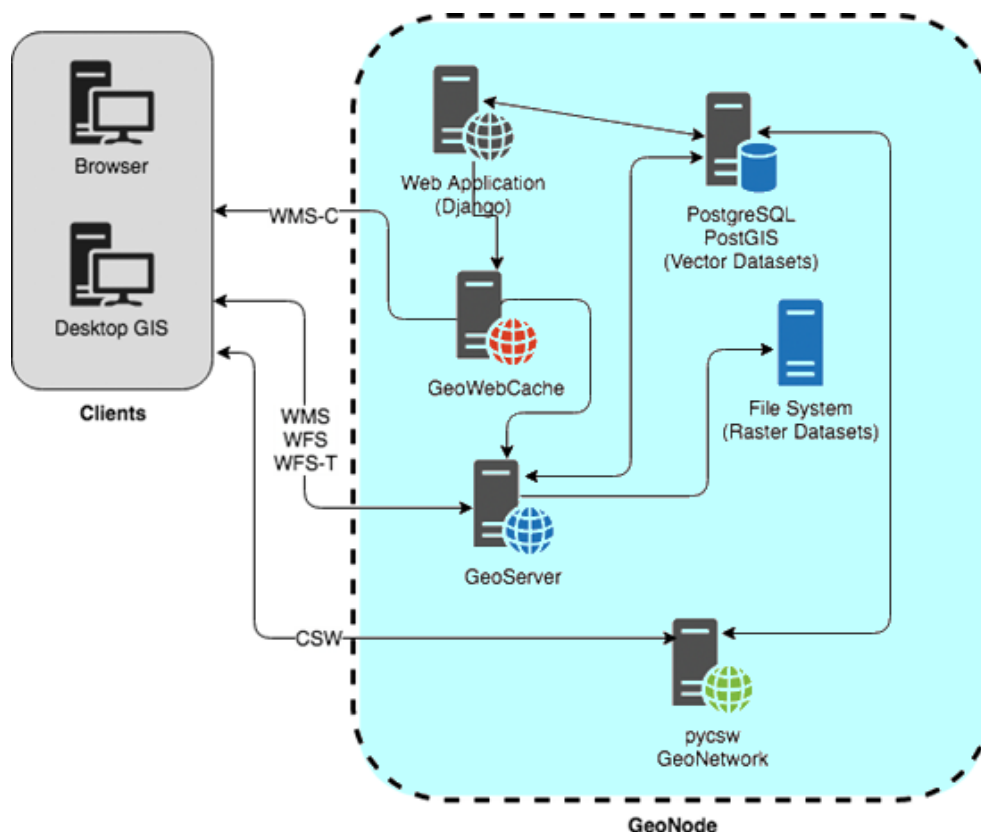
Access from a desktop GIS application

Thanks to the support for many Open Geospatial Consortium (OGC) standards (WMS, WFS, WCS, CSW), user can access to the information distributed through GeoNode directly from within their GIS desktop application (ArcGIS, QGIS, etc.).

The back-end of the spatial component (GeoPortal)

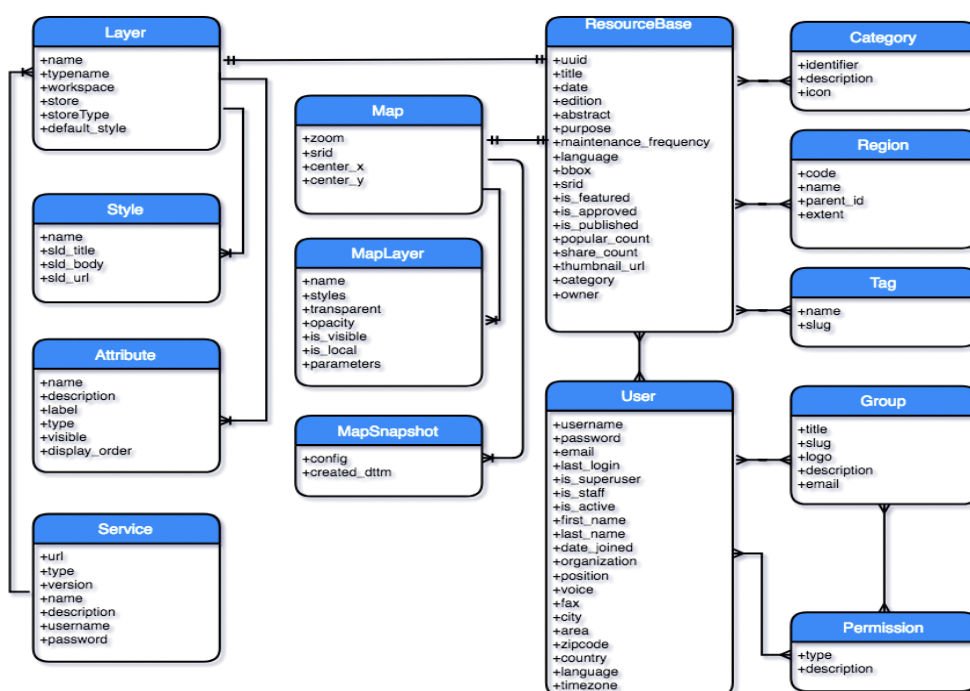
Instructions: Describe here the software "not visible" to the user that makes the system working (for example scheduled functions to harvest information, procedures to pre-process data and make them available, etc.)

The engine of GeoNode is based on several components: web application, database, file system, cache, GeoServer, specific procedures and plugins, ...



Web Application

The GeoNode web application is the central component of the framework and it is the orchestrator of the client interaction with the other components of the system. It is developed in Django, a Python web framework, and it uses a relational database, most typically based on PostgreSQL or SQLite but it can use any RDBMS supported by Django. Figure below shows a simplified version of the entity relationship model of the GeoNode database.





The core information model in GeoNode is the ResourceBase which provides a baseline set of core properties such as a universal unique identifier (uuid), a bounding box, a title, other metadata properties, publishing workflow statuses. Relevant resources (layers, maps, documents) inherit from this model in order to manage common properties in a consistent fashion. GeoNode core entities are layers and maps (in a few instances there can be also documents but for the sake of simplicity we will leave them out in this paper). It is possible to assign one category to each layer or map, a set of geographic regions (countries, continents) and a number of tags.

A **Layer** represents a spatial dataset in the system - which can be of vector or raster type or from a remote internet service. Each layer can be associated to different styles, and - in the case of vector datasets, can be composed of different layer attributes.

A **Map** represents a GeoNode web map and can be composed of different map layers. Each layer can be internal to GeoNode, or a base map from some different organizations or providers like OpenStreetMap, Google Maps, Microsoft Bing Maps, ESRI ArcGIS server or MapBox. It is also possible to register in GeoNode external map services, using OGC standards or ESRI protocols, and have remote layers. For each layer in a given map GeoNode tracks in its database the name of the layer and properties like the visibility, opacity and styles. GeoNode also tracks different versions of a specific map, therefore the user will be able to return to a previously saved version (snapshot) of the same map. The web application can be run using a web server such as nginx or Apache httpd and a Python engine like uWSGI or Unicorn.

Spatial Data Server

GeoNode includes a spatial data server which is used by the client running in the browser, or by an external GIS application, to render map tiles or stream/edit vector geometries using OGC standards such as WMS and WFS/WFS-T. By default GeoNode uses GeoServer as its internal spatial data server, and we will refer to this configuration for the following of the present article. It is possible, though, to use QGIS Server as an alternative to GeoServer.

GeoServer is a powerful Java web application, which can be run using a servlet engine such as Tomcat or Jetty, which is developed on top of the GeoTools Java geospatial library.

Spatial Cache Data Server

GeoNode uses GeoWebCache as a tiling server: its aim is to eliminate redundant request processing and save large amounts of processing time. Typically the GeoNode mapping client request tiles from the WMS-C GeoServer endpoint: if a given tile has already been generated and is not expired then it is returned from GeoWebCache to the mapping client. Otherwise it is generated by a WMS request to GeoServer.



Spatial Database

GeoNode supports both raster and vector geospatial datasets and store them respectively in the file system and in a spatial database, based on PostgreSQL/PostGIS. It is also possible for simple installations not to use the spatial database and store the vector layer on the file system as shapefiles. The spatial database provide a powerful multi user storage for the vector datasets, which let the database to be used also by different applications. For example, it is possible to directly connect to the database with a GIS client to create more advanced cartographic representations, to edit data, to perform GIS analysis on the data and so on.

PostGIS adds the benefit to have a powerful SQL spatial syntax which let external users to run powerful and fast geospatial operations.

Catalogue

A catalogue is required in a SDI to provide an interoperability mechanism to discover data and metadata. The OGC CSW standard (Consortium, 2016) has been designed for this purpose: GeoNode implements it using pycsw, which comes bundled with the GeoNode installation, or alternatively GeoNetwork OpenSource or deegree. pycsw, the default GeoNode catalogue engine, uses the Django relational database tables to provide data discoverability to external clients using CSW requests such as GetCapabilities, DescribeRecord, GetRecordById and GetRecords. A typical example of its use is via the QGIS MetaSearch plugin, which enables the user to search GeoNode datasets from within QGIS. User discovered layers can be added to the map using the WMS/WMTS, WCS or WFS GeoServer endpoints.

Search Engine

Optionally GeoNode can use a search engine such as Solr or Elasticsearch, both based on Apache Lucene to implement extremely fast and scalable search queries and enable searching capabilities such as keyword, temporal and space facets, text stemming and synonyms detection, scored results and many others.

GeoNode uses Haystack, a Django library which abstracts the search engine backend. By using Haystack and its flexible configuration parameters, some of the database models are indexed in the search engine. If the GeoNode models are changing often it is possible to reindex them on a regular basis by creating periodic tasks in the task queue.



Task Queue

GeoNode can optionally use a task queue, based on Celery and RabbitMQ, to run time demanding operations which are better handled asynchronously from the regular HTTP request/response cycle.

Typical operations which can be sent to the task queue for asynchronous processing are layer registration in GeoServer, layer thumbnails generation, email notifications, **remote services harvesting**. It is also possible to use the task queue to schedule periodic tasks, for example to reindex the search engine for an instance of a layer or a map or to do cleanup operations on spatial datasets.

Security and Authorization

The GeoNode authorization mechanism provides a way to authorize users or group of users to access to the resources (layers or maps). For both layers and maps it is possible, using the GeoNode user interface, to enable specific users or group of users to view, download, change the metadata of a specific resource. For vector layers it is possible to enable users or groups to edit the features geometry and to change the style.

GeoNode implements security and authorizations at two levels: at the Django web application level is used the Django-guardian library and at the GeoServer level is used the GeoFence plugin. Django-guardian serialize on the Django relational database the possible authorization types for a user or group of users for a given resource. GeoFence, which is a module embedded in GeoServer, uses a relational database (typically H2 or PostGIS) to store the permissions for a given user or group of users to a specific layer. OGC endpoints like WMS, WMS-C, WFS, and others will be enabled by GeoFence in GeoServer for each specific user or group of users.

Special tools

THE APPLICATION OF SOME OF THESE TOOLS REQUIRES A FURTHER EVALUATION OF THE FEASIBILITY AND AN AGREEMENT BETWEEN BENEFICIARY AND CONTRACTORS

Many suggestions, contributions, issues, etc. related to the system have been highlighted from:

- the analysis of the **requirements** of the project
- the panorama of the **stakeholders**
- the **websites and portals already distributing information** related to the Climate Change in the region
- the potential of the **existing infrastructures**
- from the **feedbacks of the many subjects met during the regional and national workshops**

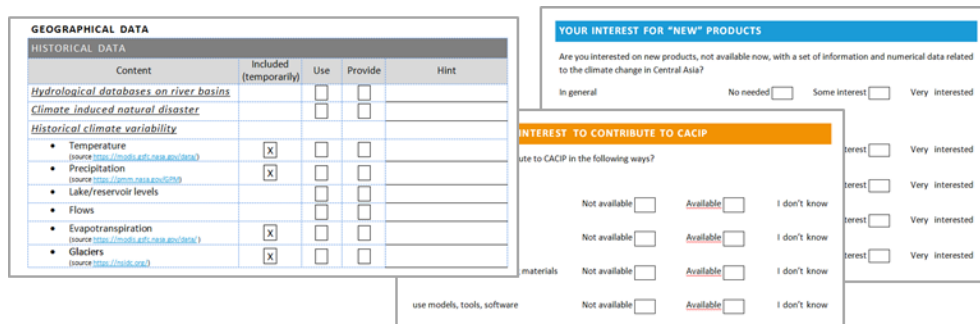
By analyzing these contributions, a set of special tools have been identified, as a solution to the many needs highlighted during the assessment and design phase of the project.

The following tools derive from these analysis, and they are a proposal to satisfy the project requirements and the stakeholders needs.

Tool: Online digital surveys

Online digital surveys (for example like the one distributed during the national and regional workshops during the assessment phase of the project) could be made accessible through the platform.

The results of the surveys are automatically collected and processed by the system to update statistical information about the interests and suggestions of the users of the platform. These information can be aggregated and will be downloadable through the system.



Content	Included (temporarily)	Use	Provide	Hint
Hydrological databases on river basins	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Climate induced natural disaster	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Historical climate variability	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Temperature (source: http://www.climat.gov.cn/)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Precipitation (source: http://www.climat.gov.cn/)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Lake/reservoir levels	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Flows	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Evapotranspiration (source: http://www.climat.gov.cn/)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
• Glaciers (source: http://www.climat.gov.cn/)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

YOUR INTEREST FOR "NEW" PRODUCTS

Are you interested on new products, not available now, with a set of information and numerical data related to the climate change in Central Asia?

In general: No needed ☐ Some interest ☐ Very interested ☐

INTEREST TO CONTRIBUTE TO CACIP

Contribute to CACIP in the following ways?

Not available ☐ Available ☐ I don't know ☐

Not available ☐ Available ☐ I don't know ☐

Not available ☐ Available ☐ I don't know ☐

Not available ☐ Available ☐ I don't know ☐

use models, tools, software

Information architecture

Instructions: Describe the information that will be stored in the system (e.g. Project information, research data, etc.).

The system has two main kinds of information: knowledge and spatial data.

Knowledge base

The information included in the system are:

- publications (reports, webinars, atlases, posters, infographics, conference proceedings, studies)
- best practices, methodologies
- projects reviews
- news, tweets

These information are stored in the internal database, or harvested "on-the-fly" from multiple sources (in this case the internal database stores only the metadata linked to the content).

Spatial data

The system could include the following typologies of spatial data. These ones derive mainly from global datasets. Below a theoretical list of contents derived from the websites inventory, and including the portal providing an interoperability interface (API or web service):

Historical climate variability

Temperature	(https://modis.gsfc.nasa.gov/data/)
Precipitation	(https://pmm.nasa.gov/GPM)
Evapotranspiration	(https://modis.gsfc.nasa.gov/data/)
Glaciers	(https://nsidc.org/)
NDVI, EVI	(https://modis.gsfc.nasa.gov/data/)
Burned areas	(https://modis.gsfc.nasa.gov/data/)
Fire	(https://earthdata.nasa.gov/earth-observation-data/near-real-time/download-nrt-data/viirs-nrt , https://firms.modaps.eosdis.nasa.gov/)
Soil moisture	(https://smap.jpl.nasa.gov/)

Climate characterization

Monthly temperature (avg, min, max)>	(http://worldclim.org/)
Precipitation	(http://worldclim.org/)
Bioclimatic variables	(http://worldclim.org/)

Current data

Surface temperature	(https://modis.gsfc.nasa.gov/data/)
Precipitation	(https://pmm.nasa.gov/GPM)

Land cover

Cover type	(https://www.esa-landcover-cci.org/ , https://modis.gsfc.nasa.gov/data/)
Glaciers/snow cover	(https://nsidc.org/)
Cropland	(https://modis.gsfc.nasa.gov/data/)
Irrigated areas	(http://www.fao.org/land-water/land/land-governance/land-resources-planning-toolbox/category/details/en/c/1029519/)
Tree cover change	(http://earthenginepartners.appspot.com/science-2013-global-forest)

Physical characteristics

Soil carbon density	(https://www.isric.org/explore/soilgrids)
---------------------	---

Global aridity index	(https://cgiarcsi.community/2019/01/24/global-aridity-index-and-potential-evapotranspiration-climate-database-v2/)
Potential Evapotranspiration	(https://cgiarcsi.community/2019/01/24/global-aridity-index-and-potential-evapotranspiration-climate-database-v2/)

Other relevant data

Agricultural productions	(http://www.earthstat.org/)
Spatial production allocation mode 2000, 2005, 2010 (SPAM)	(https://cgiarcsi.community/2019/01/04/global-spatially-disaggregated-crop-production-statistics-data-for-2010/)
Land degradation and desertification	(http://geoagro.icarda.org/cldd/)

These datasets are processed and metadata are stored in the internal database of the system.

Proposal for processing of new data (MODIS derived maps)

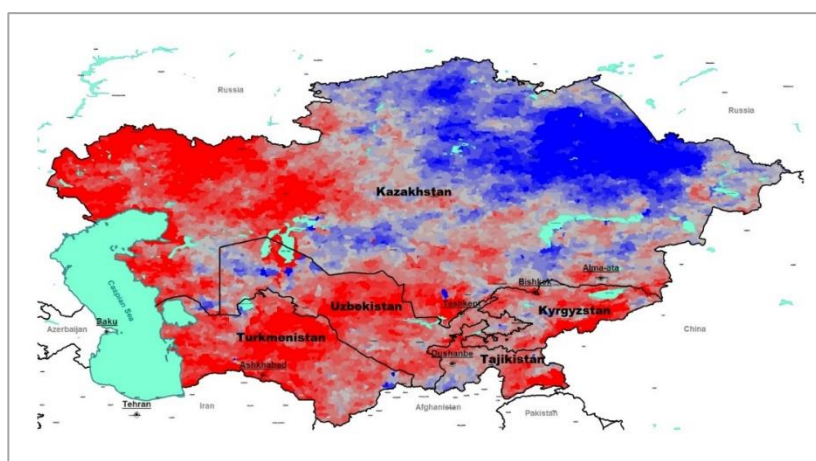
The general approach and requirement of the project is the reuse of existing data. Then all the information listed above are information already available from external/multiple sources.

To improve the potential of the platform and to increase the interest around CACIP, new geographical data could be derived for the whole Central Asia based on the processing of MODIS satellite data.

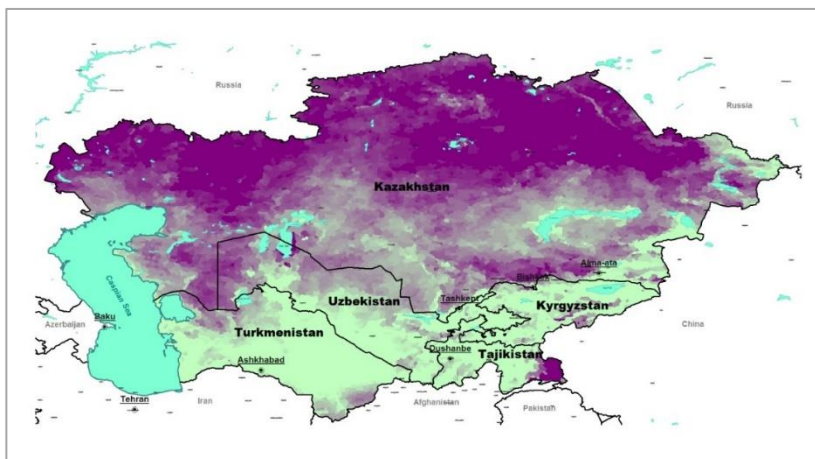
This activity could be focused on the **vegetation index** and the **land surface temperature** derived from the MODIS products MAD13 and MOD11. By analyzing the temporal series of data (available since 2000), reference index and localized anomalies can be calculated and included in the Geo-Portal.

This are some examples of these maps

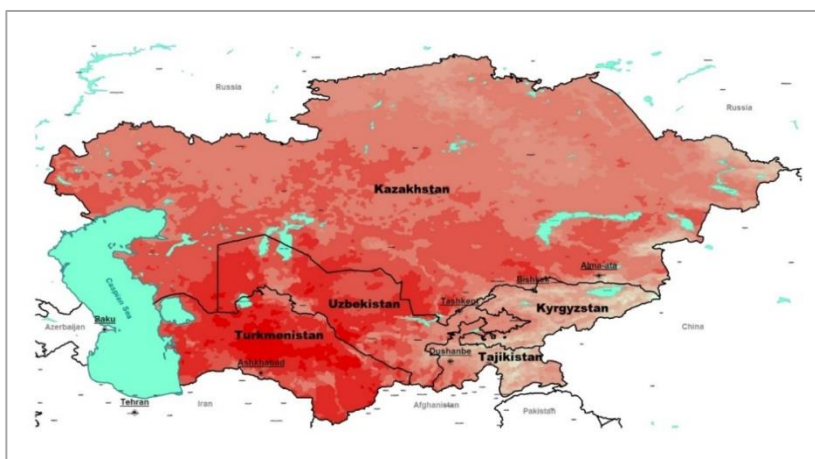
Long term land surface temperature trend



Long term average land surface temperature instability



Historical (2000-2019) land surface maximum temperature



Historical (2000-2019) land surface minimum temperature

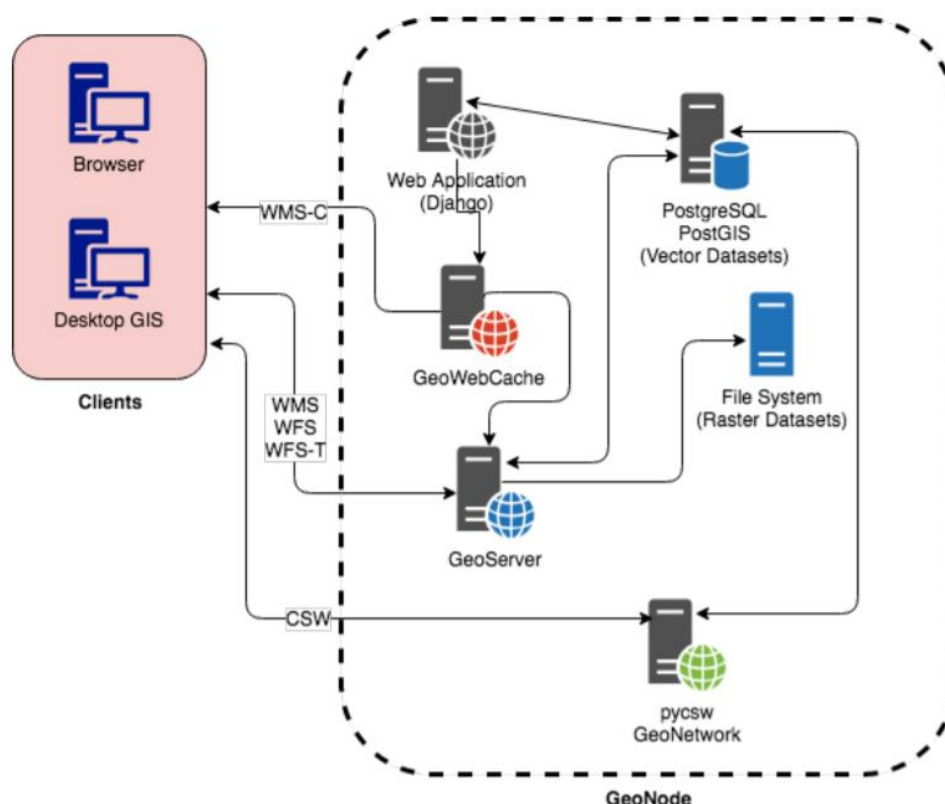


Internal communication architecture

Instructions: Provide a diagram depicting the communications flow between the system and subsystem components

Geo-Portal

GeoNode is based on the main components/libraries: Django, PostgreSQL/PostGIS, GeoServer, GeoNetwork, GeoWebCache. The description of the components and related internal relationships is in “The spatial component (GeoPortal)”



Interoperability

Instructions: Describe how the system relates with other systems, highlight the communication flows from the system and to the system, detail the interoperability standard used and supported.

GeoNode component

GeoNode provides a large number of user-friendly capabilities, broad interoperability using Open Geospatial Consortium (OGC) standards, and a powerful authentication/authorization mechanism. Supported by a vast, diverse and global open source community, GeoNode is an official project of the Open Source Geospatial Foundation (OSGeo).

In detail.

Application Programming Interfaces (APIs)

GeoNode supports multiple points for application integration. Features include:

- OGC-compliant web services
- GeoServer REST API



- GeoNode search and REST APIs

Other interoperability features

GeoNode is a social platform. GeoNode's foundational components make it easy to interact with other GeoNode deployments, external OGC services as well as with many other popular social services and tools.

- connects with other GeoNode deployments
- interacts with OGC-compliant SDIs
- integrates with other social networking platforms (Twitter, Facebook, Google+, LinkedIn, Jive, Telligent)
- integrates with other content management systems (WordPress, Drupal)

Internal and external relationships

The portal is heavily structured in components internal (the Website, the Knowledge Hub and the GeoPortal) and “external” (the portals from where the information is harvested).

Then there are relationships:

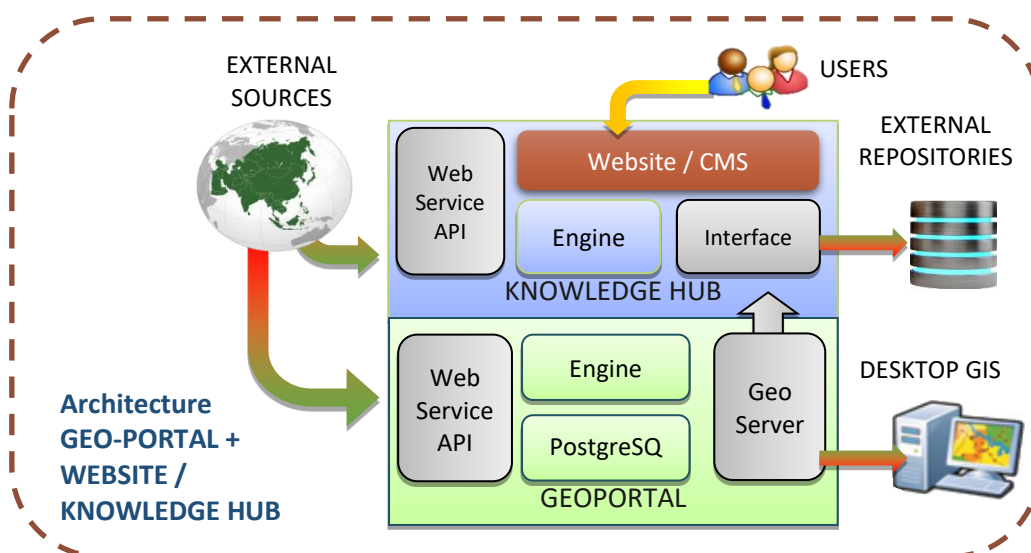
- Between the Metadata Harvester (MDH) and the website inventories (in this case the communications use specific APIs provided by the source portals, and each portal may have its own API)
- Between the front-end (web client) and the “internal” back-end, based on the HTTP protocol. CACIP homepage, knowledge hub, forum, blogs, news, GeoNode and the CACIP DB. (More details will be described during the development phase and summarized at the end of the development)
- Between the module “Today’s Numbers” and other external REST APIs (open source).

System architecture diagram

Instructions: Using the hardware, software, communications, and information designs described above, depict the overall, integrated structure of the system in terms of presentation, application, and data regions.

Diagram(s) must reflect the complete system’s context, i.e., more detailed software components, internal/external interfaces, and their underlying infrastructure, etc.

Include a table of Objects that are in the diagram.





System Design

Instructions: Describe each top level business process including actors and process flows. Insert any related project business requirements or provide a reference to where they are stored.

The analysis of user profiles and use cases is fully described in the specific document "User profiles and use cases".

Databases design

Instructions: Describe the design of all database management system (DBMS) files and non-DBMS files associated with the system. Provide a comprehensive data dictionary showing data element name, type, length, source, validation rules, maintenance (create, read, update, delete (CRUD) capability), data stores, outputs, aliases, and description. The Data Design information can be included as an appendix or recorded in a separate Database Design Document (DDD), as appropriate, which would be referenced here.

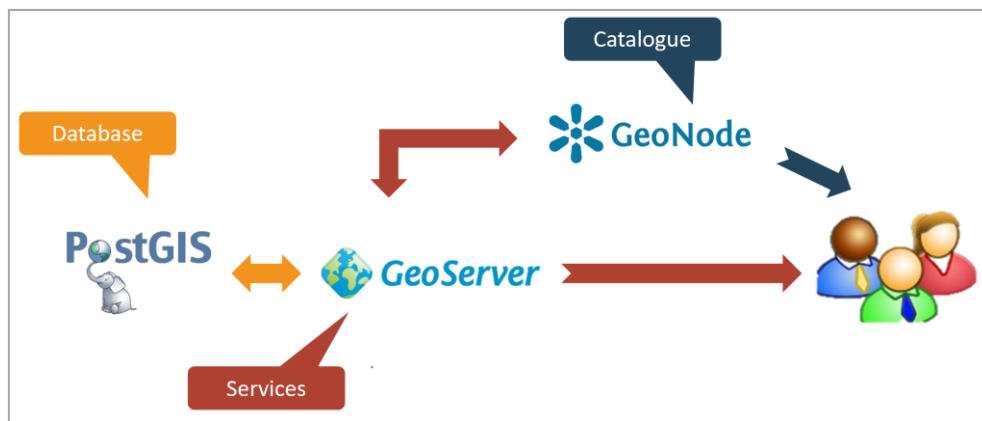
The common database behind the system is PostgreSQL, integrated with the PostGIS extension dedicated to the management of spatial data in vector and raster format.

The system embeds two separated instances: one for the **Website CMS and Knowledge Hub**, and one for the **GeoPortal**.

The Geo-Portal

GeoNode supports both raster and vector geospatial datasets and store them respectively in the file system and in a spatial database, based on PostgreSQL/PostGIS. This spatial database provide a powerful multi user storage for the vector datasets, which let the database to be used also by different applications. For example it is possible to directly connect to the database with a GIS client to create more advanced cartographic representations, to edit data, to perform GIS analysis on the data and so on.

The PostGIS extension adds the benefit to have a powerful SQL spatial syntax which let external users to run powerful and fast geospatial operations.



GeoNode uses PostgreSQL and PostGIS for storing and managing spatial data and information. All the tables and data are stored within a GeoNode database in PostgreSQL, whereas GeoServer uses PostGIS to store and manage spatial vector data for each layer.

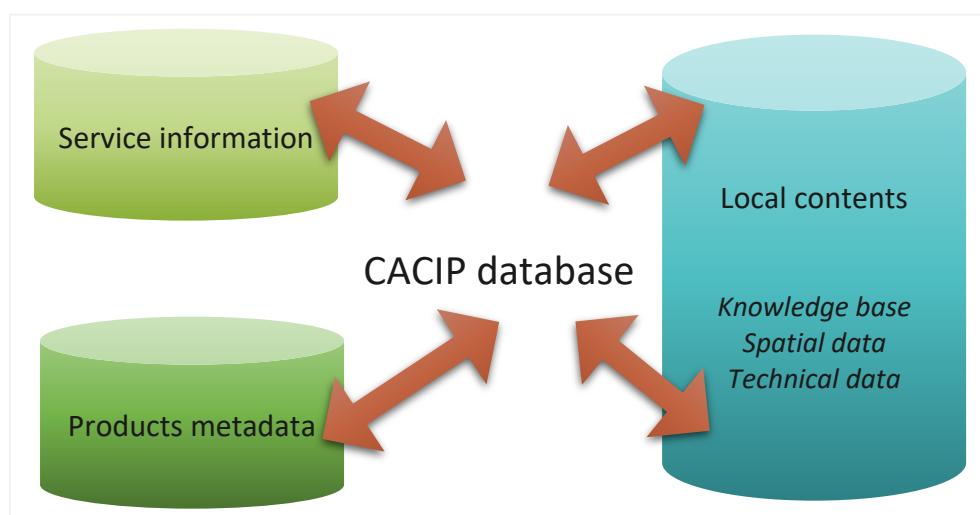
Files and databases structure

Instructions: Using the Logical Data Model (LDM), create a physical data model that describes data storage and manipulation in the systems architectural setting. Describe file structures and their locations. Explain how data may be structured in the selected DBMS, if applicable. Refer to a separate DDD, as appropriate.

CACIP database

CACIP database stores all Information used by the portal:

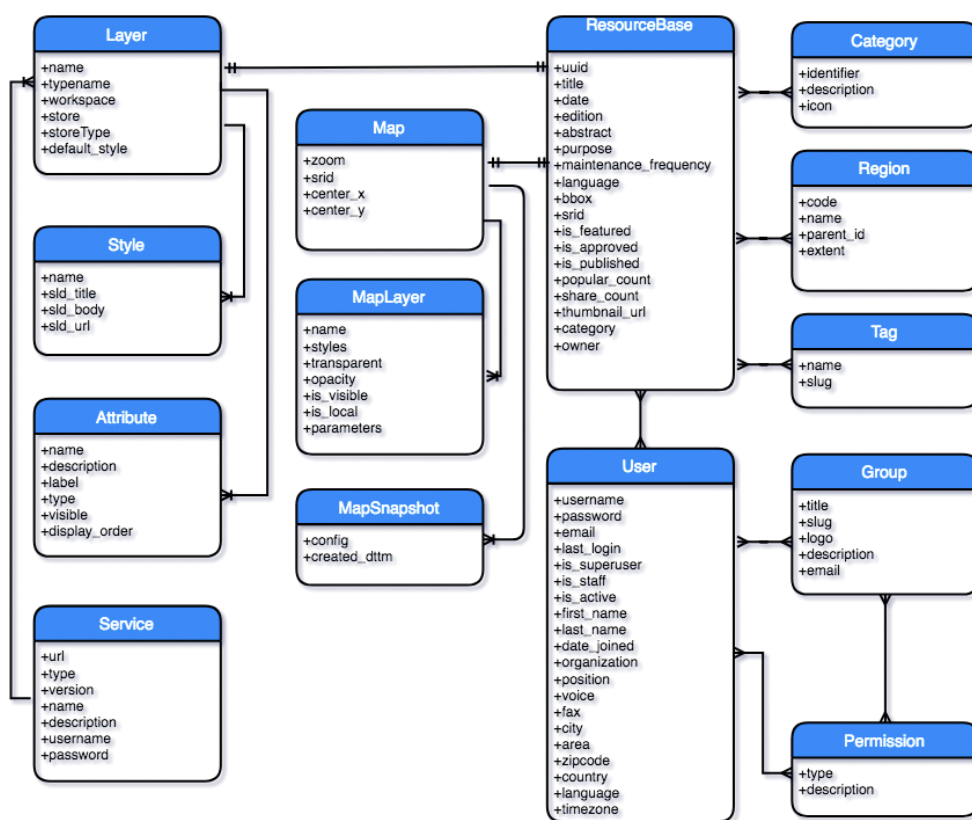
- service information (user profiles, permissions, etc.)
- products metadata (related to knowledge, spatial and technical data, and useful to allow listing, visualizing , searching and retrieving the contents)
- “local” contents (contents directly uploaded by the users and not harvested from other sources: document, geograhical data, etc.)



CACIP database is implemented with PostgreSQL, and It Is fully Integrated with GeoNode.

GeoNode and the Spatial component

Figure below shows the entity relationship model of the GeoNode database.



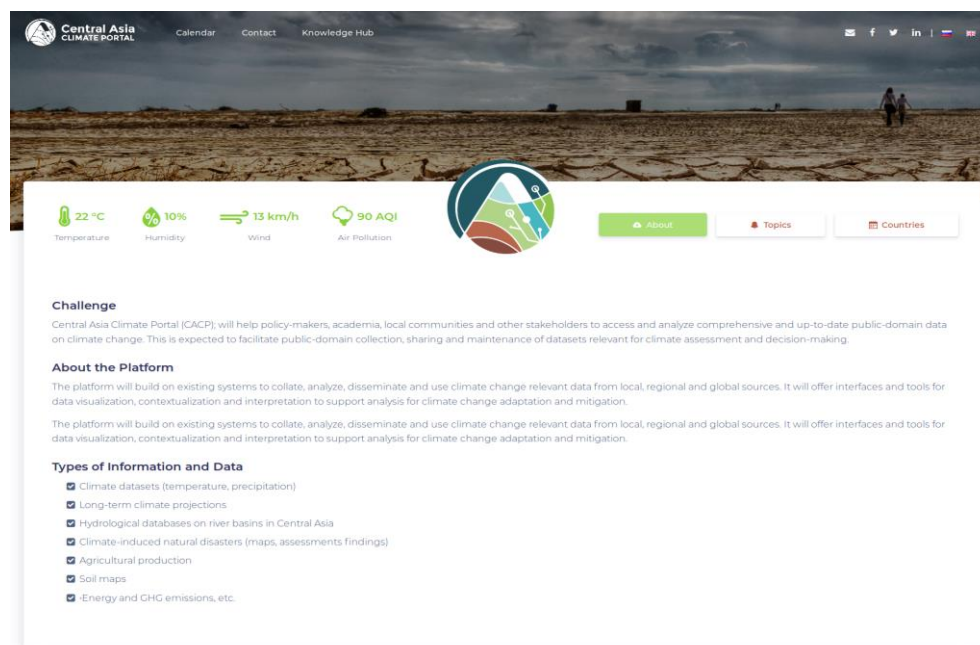


Graphical interface design

Instructions: Provide graphical visualizations of the graphical solutions adopted in the user interface

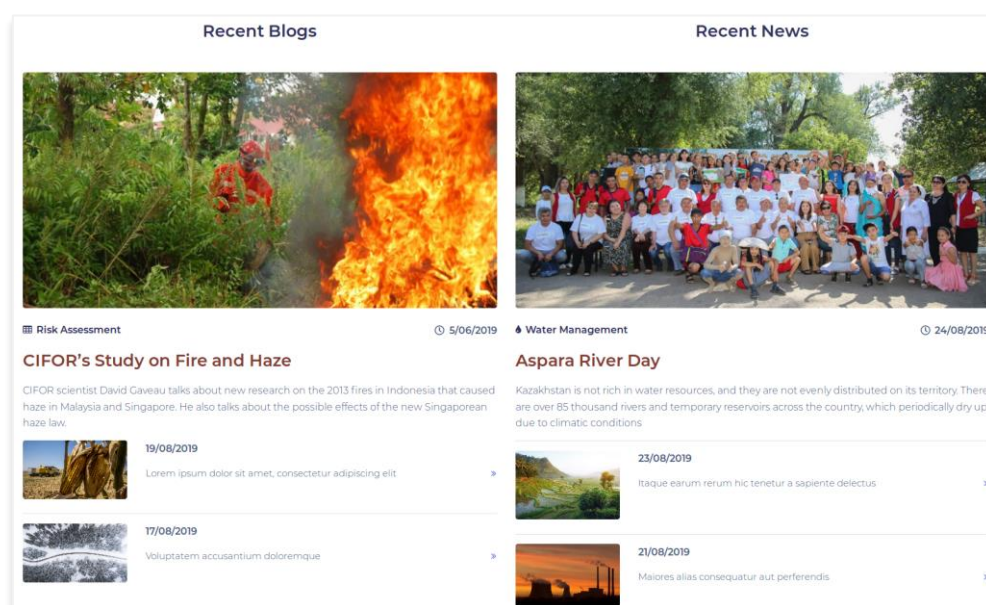
The graphical interface of the system has been designed according with the branding solution: logo, colors, fonts are the ones designed for CACIP. Below some sample pages of the website.

Home page



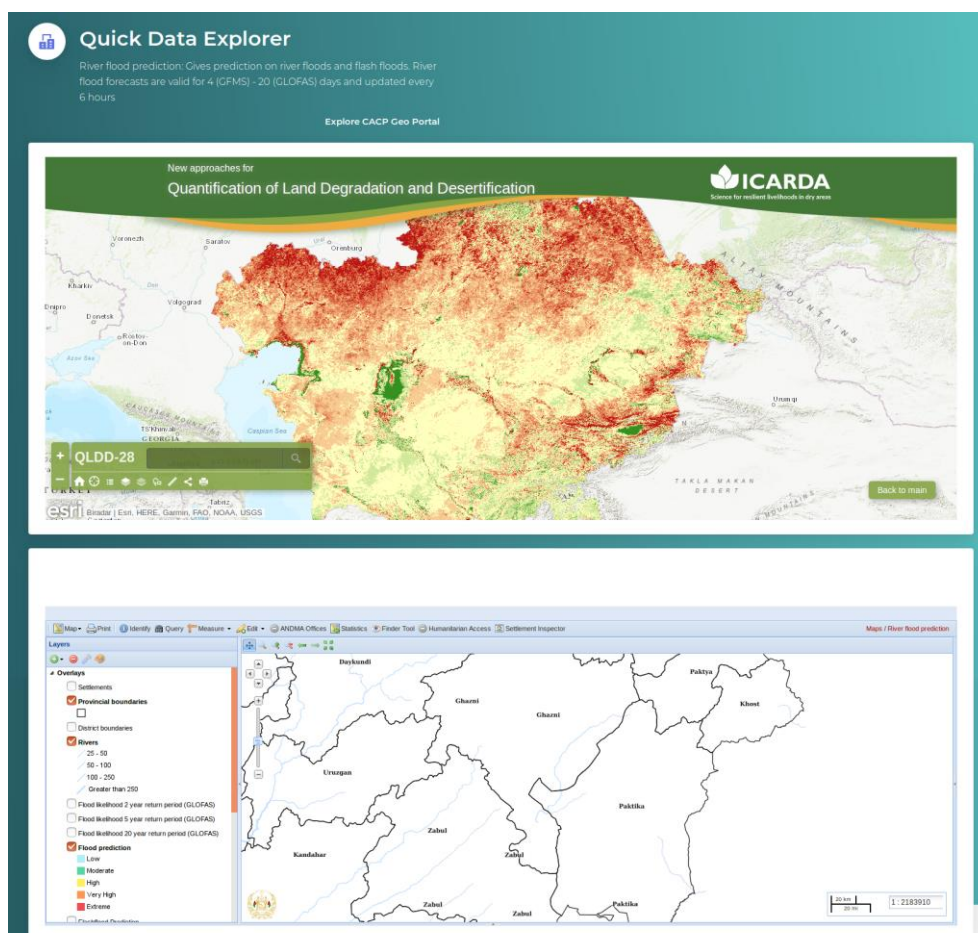
Prototype of the home page

Latest news



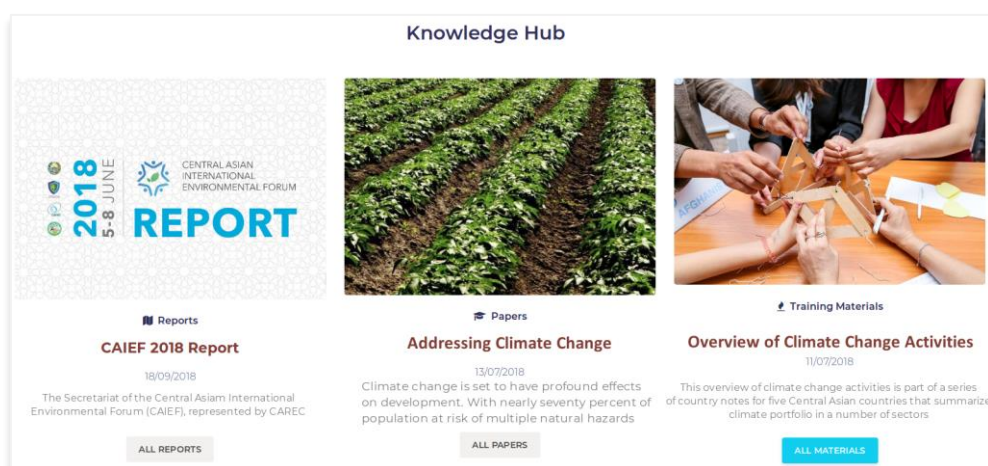
Prototype of "Recent Blogs" and "Recent News"

Map explorer



Prototype of the Map Explorer

Knowledge Hub



Knowledge Hub

2018 5-8 JUNE REPORT
CENTRAL ASIAN INTERNATIONAL ENVIRONMENTAL FORUM

Reports
CAIEF 2018 Report
18/09/2018
The Secretariat of the Central Asian International Environmental Forum (CAIEF), represented by CAREC

Papers
Addressing Climate Change
13/07/2018
Climate change is set to have profound effects on development. With nearly seventy per cent of population at risk of multiple natural hazards

Training Materials
Overview of Climate Change Activities
11/07/2018
This overview of climate change activities is part of a series of country notes for five Central Asian countries that summarize climate portfolio in a number of sectors

ALL REPORTS **ALL PAPERS** **ALL MATERIALS**

Prototype of the Knowledge Hub



Statistics on the information collected through CACIP



Site map

Instructions: List the main sections of the portal. Where possible, describe the full structure of the portal.

The site map is a text file, usually in XML, useful to describe the structure of the portal and to guarantee the accessibility to all pages. Then the real site map can only be produced when the development of the portal has been completed.

The following chart is the graphical representation a prototype version of the site map of the portal.

