Combination of CP & CSA

D7.6- ScanDB V1 (Software)

Project No.262693– INTERACT

FP7-INFRASTRUCTURES-2010-1

Start date of project: 2011/01/01 Duration: 48 months
Due date of deliverable: 31/12/2013 (M22) Actual Submission date: 31/01/2014

Lead partner for deliverable: ITU
Authors: Javier Gonzalez, Joel Granados, Philippe Bonnet, Panagiota Koltsida, Georgios Kakaletris, Yannis Ioannidis.

Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the Consortium (including the Commission Services)	
CO	Confidential, only for members of the Consortium (including the Commission Services)	

Table of Contents

Publishable Executive Summary

The ScanDB V1 trusted cells software is available by anonymous ftp at effdas.itu.dk (ScanDBV1.zip). The README file describes how to install the software.

The INTERACT Metadata Management System is accessible on http://interact.madgik.di.uoa.gr:8080

Introduction

ScanDB is a tool to help scientists to manage time-series coming from different sources and with different properties. Scientists usually collect data from many different sources that may vary in position, source, weather conditions, etc. Each source is likely to have different properties such as air temperature, wind direction or snow depth. Typically, this data is gathered in spreadsheets and self-made databases which are then used to summarize and analyze it (see D7.1 for details).
Since there is no standard way of describing how to work with scientific data, each scientist defines his/her own workflow. This makes it hard for many scientists to work together and share data. ScanDB V0 aimed to help scientists organize their data, allowing them to import data sets into a repository, annotating changes and keeping track of how data is updated with time. The system has been deployed at NERI.

One issue that has emerged with respect to the management of the ecological data sets is the lack of facility to share the data sets. This problem is particularly acute in the arctic ecological community, where the International Polar Year has failed to make a significant breakthrough in terms of data sharing. While there is a consensus that a collaborative data infrastructure is needed to allow researchers in different domains to collaborate on the same data sets in order to find new insights, there are significant barriers to its realization.

We see two key challenges: (1) How can researchers navigate through existing repositories to find a relevant data set? and (2) How can scientists share their data widely and enforce an open data policy, while retaining some form of control over who accesses this data (access control) and more importantly how it is used (usage control)?

With ScanDB V1, we focus on these two issues: a metadata management system to support search over existing data sets (the ScanDB V1 metadata service), and trusted cells to enforce usage control (ScanDB V1 trusted cells). ScanDBV1 is complementary with ScanDB V0; it is more experimental in nature.

ScanDB V1 MetaData Management System

Vision and Objectives

In the most common case of data production, scientists collect data from sources and stations and deposit those datasets into local databases (relational, file systems etc.) for further use. Sharing those data and metadata among scientists often takes place via proprietary mechanisms and manifestations, a fact that severely limits the discoverability and utilization of those datasets across the scientific community acting in the domain. The work performed on the road to the final ScanDB systems, attempts to further strengthen scientists in the process of sharing and discovering those datasets, in a manner that does intrude into their usual way of operation, enforcing strict data management and description mechanisms that often hinder data production.
As a result, a system that will aggregate dataset metadata allowing searches and exploration across station datasets and metadata, is envisioned. This system is designed as a centralized metadata repository, instantiated on a web accessible server and accompanied by a web based management and exploitation system, and from now on is called the "*INTERACT Metadata Management System*" (IMMS). The system applies loose or no rules on data handling mechanisms and metadata

manifestations, offering its services as a complementary non obstructing facility for those keen in sharing and discovering interesting data.

Technology and Concepts Basis

Technologically, IMMS builds on reusable open-source technologies and the experience and technological outcome of work carried out in other related EU Funded projects and initiatives, such as EarthServer (where a series of base components come from), iMarine and ESPAS. Those concepts and base components are adapted, or generalized, and complemented by domain-specific ones, that specialize on the INTERACT landscape, i.e. the datasets, the processes and the user communities of the project. IMMS makes minimal assumptions on metadata manifestations and no assumptions on mechanisms that manipulate or encapsulate the data (e.g. relational or array databases etc.), while aforementioned initiatives such as EarthServer and iMarine, build on top of the data management system's capabilities (e.g. APIs and query languages). Referencing dataset payload happens through content / protocol agnostic URL, while locating datasets via metadata is supported via term and xquery queries. Datasets can be optionally stored in a supplementary file storage service, that accompanies IMMS, yet its usage is not enforced. IMMS assumes no continuous connectivity with data producing stations, and only user access is required to update and consume its services.

Finally INTERACT Metadata Management System handles the specialized ScanDBv0 dataset metadata, yet it assumes those being one of many manifestations that might occur in the future.
Summing up INTERACT based technologies include:

- XML standard is used to describe datasets' metadata which are saved into an XML database
    - A generic XML schema is used to describe all the imported metadata,
    - The schema of the actual metadata is freely defined for each station/data owner and is integrated into the core schema
- Xquery is provided (through a service and a web UI) for querying the metadata
    - Term search is also supported on metadata
- Search, exploration and metadata management facilities are offered through dedicated web User Interface elements, now on called *portlets*, deployed onto a portlet and content management engine, the Liferay[1] system.

Furthermore, two more facilities are under implementation:

- A tool to export and submit ScanDB v0 local database metadata into the IMMS is under development.
- A file server, integrated with the IMMS system will be provided as a point of web storage for datasets
    - Currently h/w resources are being allocated for the file storage

In the following paragraphs further information is given on the aforementioned technologies.

Involved Standards and Technologies

In the following paragraphs we describe the languages, the standards and the technologies that are involved in the implementation of the IMMS.

[1] http://www.liferay.com/

XQuery – XPath

XPath is a language for addressing parts of an XML document and it also provides basic facilities for manipulation of strings, numbers and booleans. XPath uses a compact, non-XML syntax to facilitate use of XPath within URIs and XML attribute values.
XPath operates on the abstract, logical structure of an XML document, rather than its surface syntax.
XQuery is a language capable of retrieving information from documents in XML structure. Specifically, XQuery instances will extract information from XML documents using the XPath standard. It is designed to be a language in which queries are concise and easily understood. Moreover, XQuery is flexible enough to query a broad spectrum of XML information sources, including both databases and documents
Both these standards can be used for directly querying the available datasets stored in XML format.

eXist XML Database

eXist is an open source database management system entirely built on XML technology, also called a native XML database. Unlike most relational database management systems, eXist uses XQuery, which is a W3C Recommendation, to manipulate its data.
eXist is used for storing and querying the available datasets.

Liferay Portal

Liferay Portal is a free and open source enterprise portal written in Java and distributed under the GNU Lesser General Public License and proprietary licenses. It is primarily used to power corporate intranets and extranets.
Liferay Portal allows users to set up features common to websites. It is constructed of functional units called portlets[2]. Liferay Portal is Java-based and runs on any computing platform capable of running the Java Runtime Environment and an application server. Liferay is available bundled with a servlet container such as Apache Tomcat

Google Web Toolkit (GWT)

GWT is a development toolkit for building and optimizing complex browser-based applications. Its goal is to enable productive development of high-performance web application, XML, HttpRequest, and JavaScript. GWT emphasizes reusable, efficient approaches to common web development tasks, namely asynchronous remote procedure calls, history management, bookmarking, User Interface abstraction, internationalization, and cross-browser portability.

INTERACT Metadata Management System Architecture

The main component of our system is the *Registry - Catalog* that provides all the needed functionality for interacting with the XML database in order to store, retrieve and query the available datasets' metadata.

On top of the catalog several portlets have been adapted to IMMS needs, offering rich user interfaces for interacting with the catalog and manipulating the available metadata. These portlets are deployed on a liferay portal engine that can be accessed via any widely used web browser.

[2] Java Portlet Specification: https://www.jcp.org/en/jsr/detail?id=286

In addition, IMMS can optionally[3] offer an OpenSearch[4] interface, thus being interoperable with other search engines.

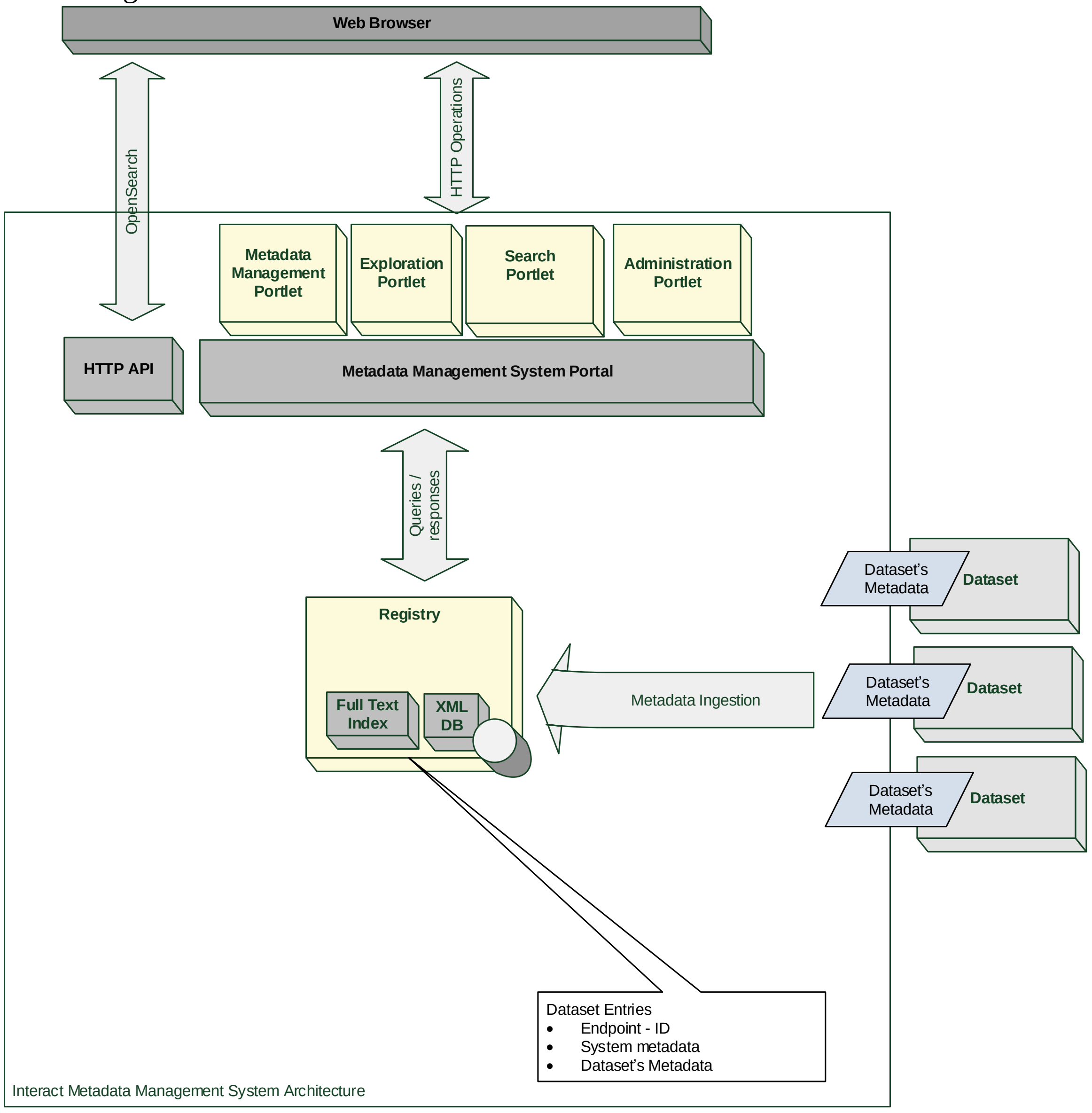


Figure 1 INTERACT Metadata Management System Architecture

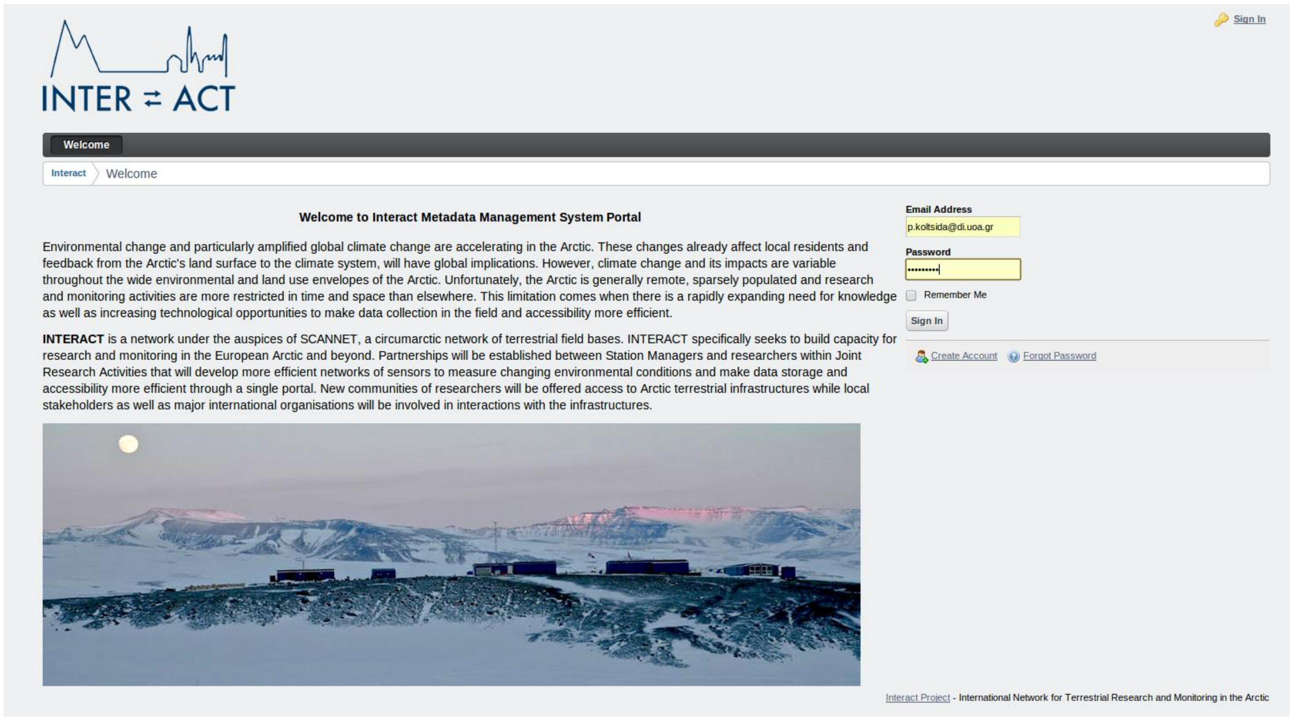
Implemented Components

A deployed, customize liferay-based portal is available at the following url:

http://interact.madgik.di.uoa.gr:8080/web/interact/home

[3] OpenSearch is offered via EarthServer project, yet significant adaptation (or generalization) is required for providing it as an IMMS service.

[4] http://www.opensearch.org/Home



Welcome to Interact Metadata Management System Portal

Environmental change and particularly amplified global climate change are accelerating in the Arctic. These changes already affect local residents and feedback from the Arctic's land surface to the climate system, will have global implications. However, climate change and its impacts are variable throughout the wide environmental and land use envelopes of the Arctic. Unfortunately, the Arctic is generally remote, sparsely populated and research and monitoring activities are more restricted in time and space than elsewhere. This limitation comes when there is a rapidly expanding need for knowledge as well as increasing technological opportunities to make data collection in the field and accessibility more efficient.

INTERACT is a network under the auspices of SCANNET, a circumarctic network of terrestrial field bases. INTERACT specifically seeks to build capacity for research and monitoring in the European Arctic and beyond. Partnerships will be established between Station Managers and researchers within Joint Research Activities that will develop more efficient networks of sensors to measure changing environmental conditions and make data storage and accessibility more efficient through a single portal. New communities of researchers will be offered access to Arctic terrestrial infrastructures while local stakeholders as well as major international organisations will be involved in interactions with the infrastructures.

Email Address
p.koltsida@di.uoa.gr
Password
•••••••••
Remember Me

Interact Project - International Network for Terrestrial Research and Monitoring in the Arctic

Figure 2 Interact Metadata Management System Portal

The implemented portlets offer a rich UI for interacting with the system. Exploiting these portlets a user can:

- Search through the metadata, either by submitting free text queries or using xQuery directly.
- Explore the available datasets
- Visualize them on a map by exploiting their geospatial information, if any.
- Manage the metadata. Edit existing metadata or upload new metadata for each dataset.

In addition users with 'administration' role can access the administrative portlet and add new datasets to the system.

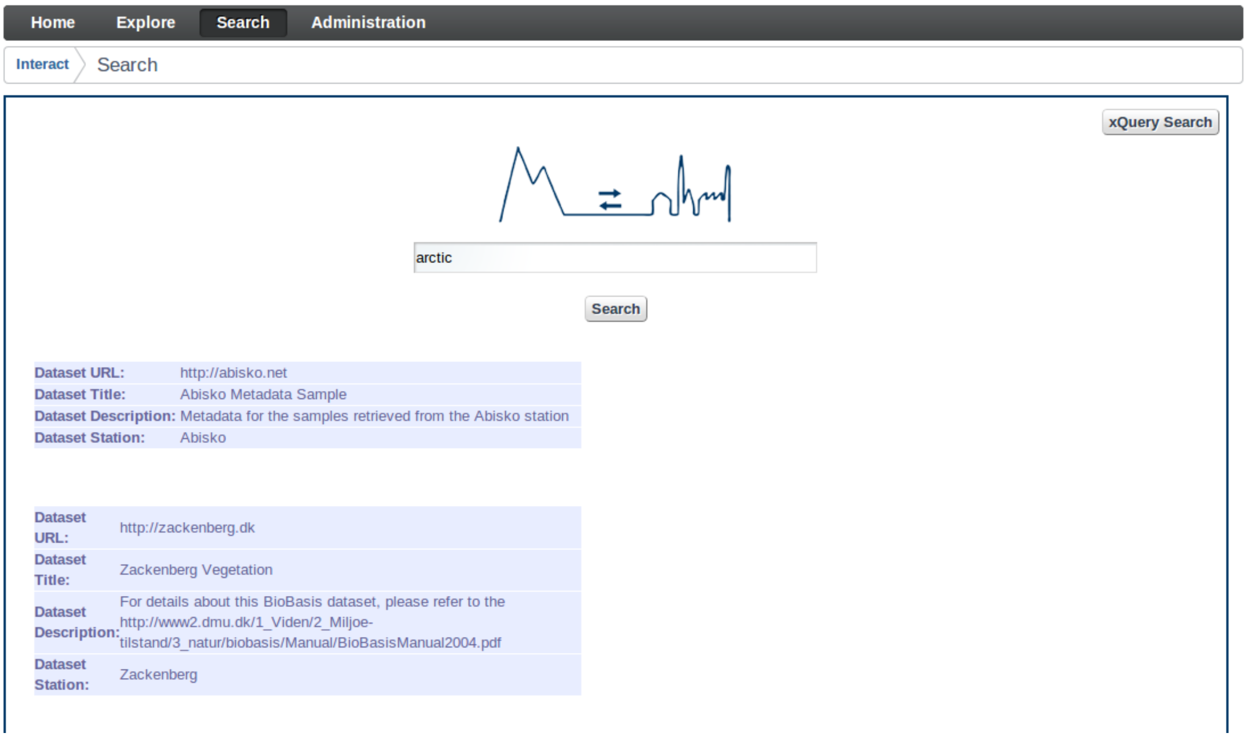
At the time of writing this report the 3 out of the 4 aforementioned portlets have been deployed and are available, and we are working on implementing the metadata management portlet.

Search Catalog portlet

This portlet provides a user interface with the following functionality:

- Free text search on the registered datasets' metadata
- XQuery search on the datasets' metadata

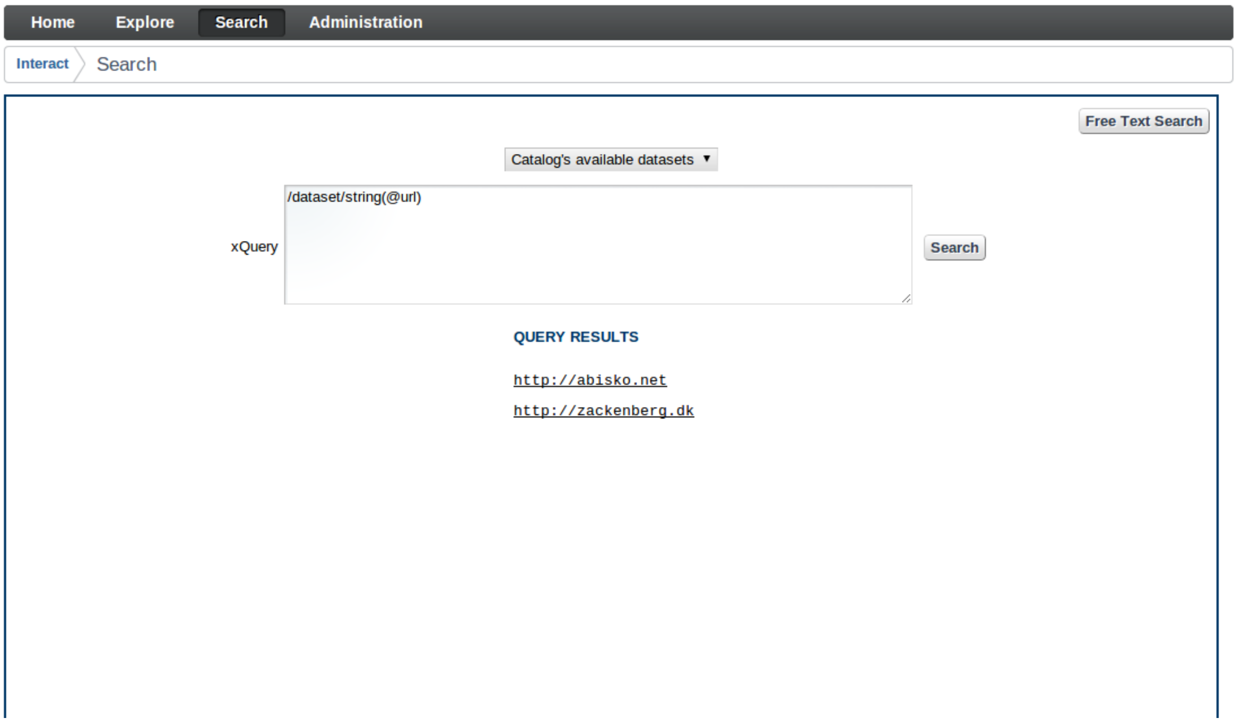
It interacts with the catalog (XML database) and retrieves the requested information. A full text index has been triggered for all the available metadata by exploiting eXists functionality.



Dataset URL:	http://abisko.net
Dataset Title:	Abisko Metadata Sample
Dataset Description:	Metadata for the samples retrieved from the Abisko station
Dataset Station:	Abisko

Dataset URL:	http://zackenberg.dk
Dataset Title:	Zackenberg Vegetation
Dataset Description:	For details about this BioBasis dataset, please refer to the http://www2.dmu.dk/1_Viden/2_Miljoe-tilstand/3_natur/biobasis/Manual/BioBasisManual2004.pdf
Dataset Station:	Zackenberg

Figure 3 Search Catalog Portlet - Free text search



```
/dataset/string(@url)
```

http://abisko.net
http://zackenberg.dk

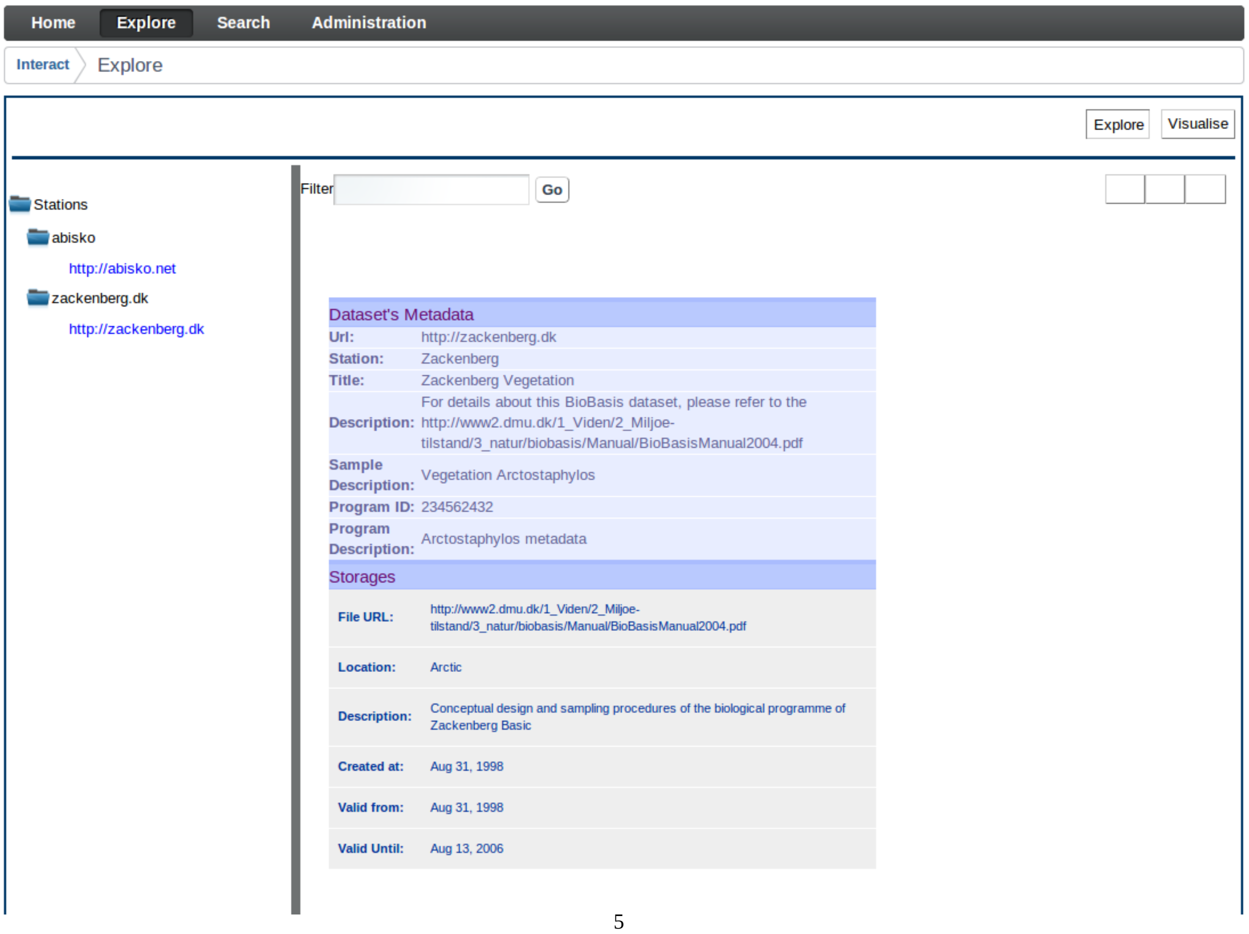
Figure 4 Search Catalog Portlet - XQuery search

Explore Catalog portlet

This portlet provides a user interface with the following functionality:

- Lists all the available metadata, grouped by station

- Displays the metadata either in HTML format or in XML plain format
- Visualizes the geo information on a map, if any
- Geospatial search over the bounding boxes of the registered datasets, supporting the “contains”, “intersects” and “is_contained” relations



Dataset's Metadata

Url:	http://zackenberg.dk
Station:	Zackenberg
Title:	Zackenberg Vegetation
Description:	For details about this BioBasis dataset, please refer to the http://www2.dmu.dk/1_Viden/2_Miljoe-tilstand/3_natur/biobasis/Manual/BioBasisManual2004.pdf
Sample Description:	Vegetation Arctostaphylos
Program ID:	234562432
Program Description:	Arctostaphylos metadata

Storages

File URL:	http://www2.dmu.dk/1_Viden/2_Miljoe-tilstand/3_natur/biobasis/Manual/BioBasisManual2004.pdf
Location:	Arctic
Description:	Conceptual design and sampling procedures of the biological programme of Zackenberg Basic
Created at:	Aug 31, 1998
Valid from:	Aug 31, 1998
Valid Until:	Aug 13, 2006

[5]

Figure 5 Explore Catalog portlet- Metadata Display

[5] The metadata being displayed in the screenshots are sample and in no case they present real metadata

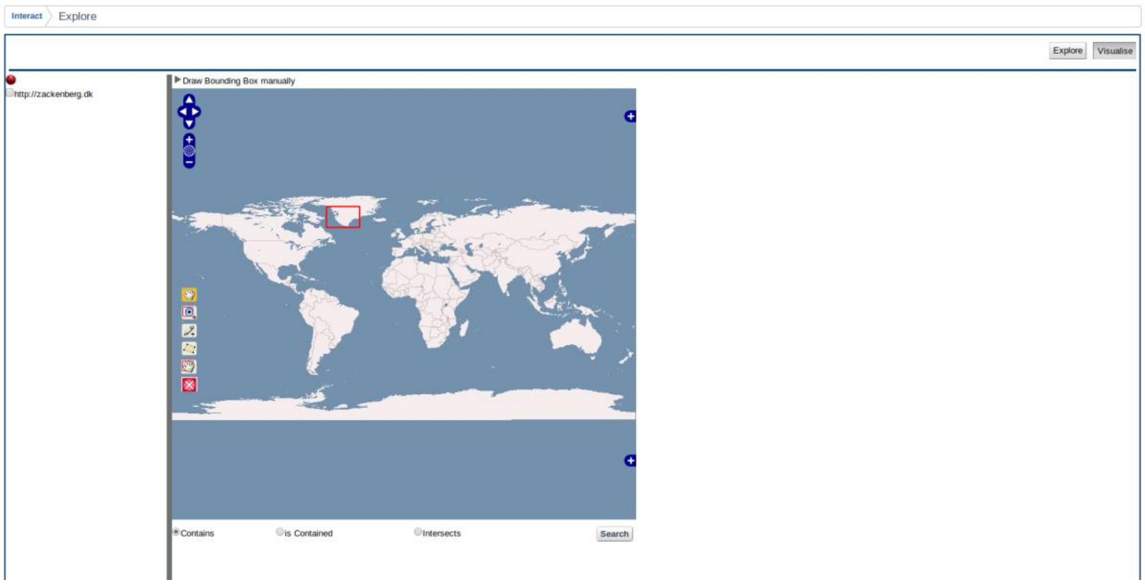
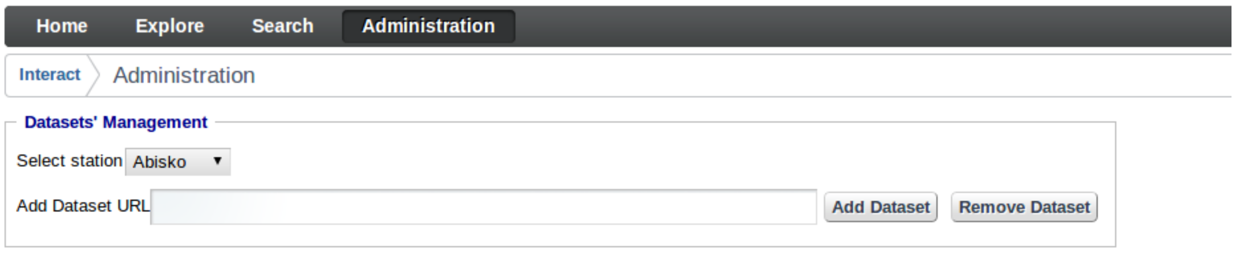


Figure 6 Explore Catalog portlet - Map Visualization

Administration portlet

The administration portlet can be used to:

- Insert or remove a dataset from the catalog



Select station Abisko

Add Dataset URL

Figure 7 Administration portlet

Metadata Management portlet

A metadata management portlet is being implemented to offer the following functionality:

- View and edit a dataset’s metadata
- Insert xml metadata directly, using an uploader. These metadata will be hosted together with the dataset’s systemic metadata
- Insert and edit user annotations

NOTE: All the above described User Interface elements will be enhanced based on the needs and the feedback provided by end users in the subsequent periods of the projects

ScanDB V1 Trusted Cells

Vision

In a simplified example of how ecological data sets are collected and shared in the INTERACT community, Figure 1 shows activities taking place in a scenario involving two field sites (Zackenberg and Abisko), four labs (DMU, U.Lund, U.Oulu, and U.Alaska) and the online Knowledge Network for Biodiversity (KNB) repository for ecological data sets recommended by Nature. Data acquisition takes place at the field sites. Data acquisition is either manual or automatic; most of the collected data is digital on data loggers or PDAs, some of the data collected manually is in analog form (and is later digitized in a lab). On Figure 1, A and E are technicians collecting data in the context of monitoring programs at Zackenberg and Abisko, while B is a researcher from U.Oulu who is collecting observations during an expedition to Zackenberg.

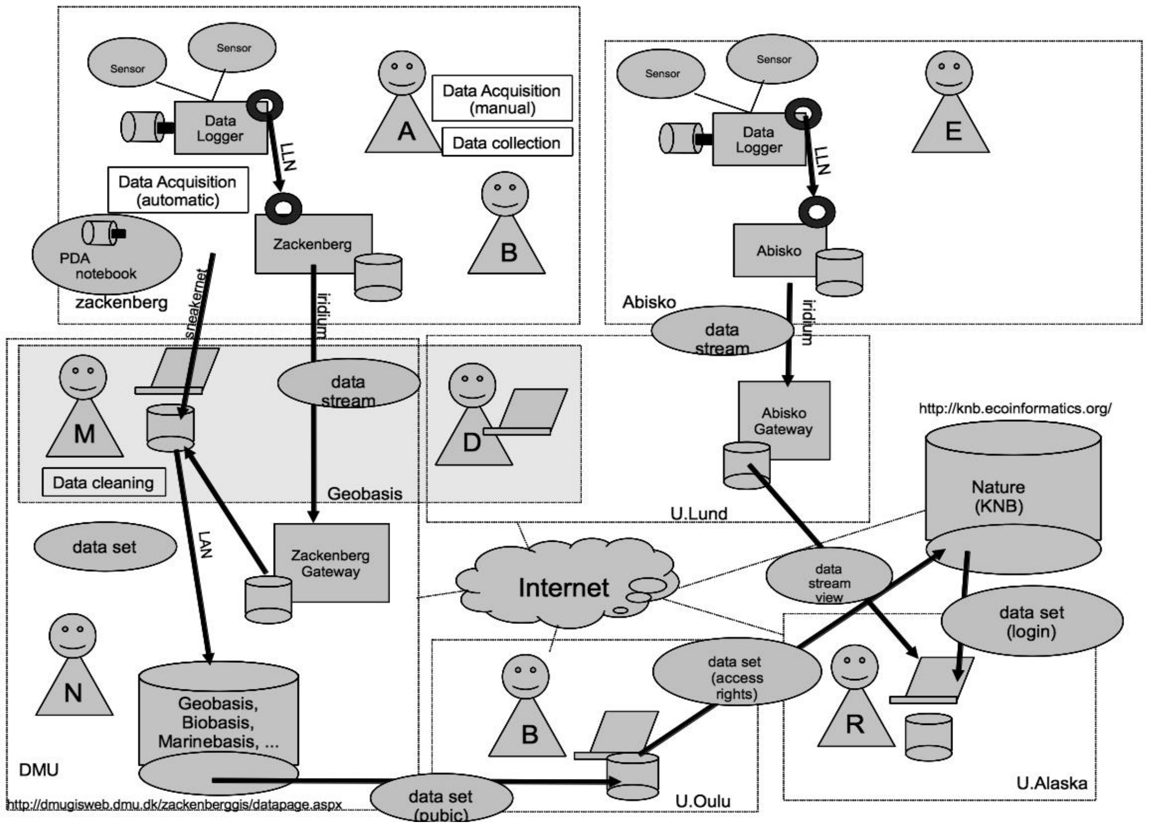


Figure 1: Sanitized case study illustrating data sharing in the INTERACT community

Today, data is transferred from the field sites to a lab either via physical transfer or the storage media (sneakernet). A goal of the INTERACT project is to make the streams of data collected by data loggers available on the Internet. Note that data loggers cannot be directly connected to the Internet because of obvious connectivity limitations. Data is gathered via a low power local area network (LLN) onto a hub that is connected to a lab gateway via satellite connection (Iridium). The data collected from the data loggers are stored onto a database on the gateway, then made available online as a stream (views might be defined for the purpose of access control).

To illustrate how data is processed and shared in the context of a monitoring program, M is manager of the monitoring program at DMU (NERI). He receives the data collected by A in the field. Her responsibility is then to go through a data cleaning phase, and possibly a data derivation phase. These phases might be conducted together with colleagues from other institutions (U.Lund) in our example) participating in the program, under the responsibility of the program manager. Today, data sets are exchanged by mail or made available on ftp sites, i.e., a weak form of access control is enforced and there is no usage control. Double checks are necessary to make sure that anomaly detection or a data derivation is conducted properly. This is very time consuming.

When the program manager is satisfied with the quality of a data set, she can publish it. To do so, she sends the data set to a local data manager who loads the data into a database equipped with a web query front-end. Once data is made publically available on the DMU web site, the program manager has no longer control over who accesses it or how it is used. For example, if a new annotation is attached to a data set, there is no way to communicate it to the researchers who previously downloaded the data set. There is also no way to check that the researchers that use the data actually follow the usage policy defined by the data manager.

Let us now turn our attention to how data is processed and shared in the context of a research project. B collected data in the field. Back in the lab, at U.Oulu, he combines the data he has patiently gathered over the last 10 years with data sets he downloaded from the DMU web site. He builds a new predictive model and publishes his findings in Nature. As part of the Nature publication policy, he makes his data available online on the KBN web site. **Ideally, reviewers and other researchers should be able to check that his model actually fits the data he gathered, they should be able to check that there is no obvious anomaly in the collected data, but they should not be able to derive a new model without explicitly crediting B for his work.** Note that the access control mechanism provided by the KBN web site allows B to control who is accessing his data, but not how this data is then used (and possibly transmitted to a third party). For example, R from U.Alaska has contacted B to get access to his data set. Once access is granted, R cannot track how the data is used and how it is combined with the data stream obtained from Abisko. **More interestingly, R might not credit DMU for collecting the data sets that were instrumental for building B's model.** This is an illustration of the issue identified by the OECD report on open data policies: "*Whenever possible, access to data sets should be linked with access to the original research materials, and copied data sets should be linked with originals, as this facilitates validation of the data and identification of errors within data sets.*"

Note that researchers might in fact be more interested in a form of auditing of how their data is used rather than enforcing strict usage control policies, governmental organizations might share this goal, while private organizations might want to enforce more precise usage control policies. Our approach consists in providing a uniform solution to those different requirements.

But, what is usage control? Usage control models usually refer to UCONABC [3]. In the UCONABC model, subjects provide or consume data objects. A subject accesses objects via a set of usage functions referred to as rights. A reference monitor is responsible for taking usage decisions based on the subject and object attributes as well as Authorization, oBligations and Conditions (ABC). Authorizations are predicates that define whether a subject is authorized to hold a right; obligations are predicates that define the actions that a subject must take before or while it holds a rights; and conditions define predicates that must hold true about the environment in which the subject requires a given right. In our experience, current open data policies defined for ecological data sets can easily be expressed in terms of (i) rights, and (ii) authorizations, obligations and condition rules.

What does it take to enforce a UCONABC model? The data platform should implement a reference monitor and guarantee that there will be no access to data objects unless appropriate usage decisions (i.e., decisions that respect the contract negociated by all parties) are taken. The basic idea is that the reference monitor is a software component that relies on hardware security features separating a secure world (where objects, attributes, as well as rights, authorizations and conditions are securely stored, while usage decisions are securely executed) and a non secure world (where the rest of the processing takes place). There is today, to the best of our knowledge, no implementation of any UCONABC model.

How to enforce access control and usage control in this scenario? A solution is to consider that all actors rely on Google's FusionTable (or a similar centralized architecture) to store and share their data. The features of FusionTable might not exactly match the needs of all actors, but they come pretty close. A problem could be the lack of connectivity on the field sites, but this might not be a major barrier to adoption. A more serious problem would be the monopoly position gained by Google if all researchers and institutions started to publish their data on FusionTables.

The solution we investigate with ScanDB V1 is a different point in the design space. We define a distributed infrastructure for sharing ecological data sets with access and usage control guarantees. In this distributed architecture, researchers and data managers are equipped with trusted cells [1]. We describe the architecture of trusted cells in the next Section.

Trusted Cells

The decentralized architecture we propose for the sharing of ecological data sets with access and usage control guarantees is based on *Trusted Cell*s interconnected via an *Untrusted Infrastructure*.

Trusted Cells: A trusted cell implements a client-side reference monitor [10] on top of secure hardware. At a minimum, the hardware must guarantee a clear separation between secure and non-secure software. We abstract a Trusted Cell as (1) a Trusted Execution Environment, (2) a tamper-resistant memory where cryptographic secrets are stored, (3) an optional and potentially untrusted mass storage and (4) communication facilities. Physically, a trusted cell can either be a stand-alone hardware device (e.g., a smart token) or be embedded in an existing device (e.g., a smartphone based on ARM’s TrustZone architecture).

The very high security provided by trusted cells comes from a combination of factors: (1) the obligation to physically be in contact with the device to attack it, (2) the tamper-resistance of (part of) its processing and storage units making hardware and side-channel attacks highly difficult, (3) the certification of the hardware and software platform, or the openness of the code, making software attacks (e.g., Trojan) also highly difficult, (4) the capacity to be auto-administered, contrary to high-end multi-user servers, avoiding insider (i.e., DBA) attacks, and (5) the impossibility even for the trusted cell owner to directly access the data stored locally or spy the local computing (she must authenticate and only gets data according to her privileges).

In terms of functionality, a full-fledged trusted cell should be able to (1) acquire data and synchronize it with the user’s digital space, (2) extract metadata, index it and provide query facilities on it, (3) cryptographically protect data against confidentiality and integrity attacks, (4) enforce access and usage control rules, (5) make all access and usage actions accountable, (6) participate to computations distributed among trusted cells. Basic (e.g., sensor-based) trusted cells may implement a subset of this.

Untrusted infrastructure: The infrastructure provides the storage, computing and communication services, which expand the resources of a single trusted cell and form the glue between trusted cells.

By definition, the infrastructure does not benefit from the hardware security of the trusted cell and is therefore considered untrusted. We consider that a Cloud-based service provider implements the untrusted infrastructure[6].

In terms of functionality, the untrusted infrastructure is assumed to: (1) ensure a highly available and resilient store for all data outsourced by trusted cells, (2) provide communication facilities among cells and (3) participate to distributed computations (e.g., store intermediate results), provided this participation can be guaranteed harmless by security checks implemented at the trusted cells side.

Software Design

ScanDB V1 focuses on the storage and retrieval of time series in a **trusted execution environment**. This is the foundation of a trusted cell. The design of access and usage control policies adapted for the INTERACT community, and the design/implementation of mechanisms to enforce these properties is a topic for on-going work.

According to the GlobalPlatform consortium1, a trusted execution environment (TEE) is a secure area of a computing device that ensures that sensitive data is only stored, pro- cessed and protected by authorized software. The secure area is separated by hardware from the device's main operating system and applications. Put differently, computing devices fall in two categories: (i) general purpose devices that do not provide any guarantee for authorized software, and (ii) secure devices, where authorized software can execute in a secure area which is not accessible by the rest of the system. Secure devices separate a secure area from a non-secure area, which we denote rich area in the rest of this document, following the terminology from the GlobalPlatform consortium, while general purpose computing devices merely provide a rich area.

A potential large number of applications are expected to be installed and run concurrently in the rich area, therefore sharing software and hardware resources (e.g., libraries, drivers, peripherals, memory, etc.). As in any modern rich execution environment, access control is in effect to protect shared resources, with a user space distinct from a kernel space. Software running in kernel space has access to all resources (e.g., data and programs in memory or on secondary storage). The rich area is in constant exposure to the outer world, thus attacks are to be expected. The assumption should be that with enough time and expertise an attacker can obtain root access to kernel space. As a consequence, all programs and data stored in or handled in the rich area are at risk of being exposed, including sensitive data and encryption keys.

As we stated above, a secure device provides a hardware separation between the secure area and the rich area. Note that this is a slight generalization of GlobalPlatform's definition, which mentions that rich and secure areas should run on a same processor. We extend this de_nition so that

a secure co-processor, or a smart card are considered trusted execution environments. In fact, we consider that any device equipped with a processor for the rich area, and another one for the secure area is a secure device as long as there is a guarantee that only authorized software runs in the secure area. We even consider that a virtual machine (VM) is a trusted execution environment as long as the hypervisor relies on hardware primitives to isolate each VM. Defining an ontology of these secure devices and precisely qualifying how secure they are is beyond the scope of this demonstration. It is a topic for future work.

[6] A P2P infrastructure among trusted cells could be envisioned but would raise many technical issues of limited interest for this article.

Whenever rich and secure areas share physical memory, it is mandatory that software running in the rich area by no means is able to access memory allocated to the secure area. If peripherals are shared Secure must have prioritized access to them in such a way that if the rich area is compromised, the peripheral can still be accessed from the secure area even when a DoS attack is launched against it. More generally, the only assumption made in the secure area is that the authorized code that is run there can be trusted to protect the integrity and con_dentiality of the data. This is a big assumption, but model-based development and formal methods can provide interesting guarantees. Again, exploring how model-based development can be leveraged in the context of secure data management is a fascinating topic for future work.

ScanDB V1 relies on ARM TrustZone. In [2] we presented a framework that combines commercially available hardware and open source software and enables the development of applications for TrustZone's secure area. In this framework, rich and secure areas are running on a single processor. TrustZone provides a hardware mechanism to separate the secure and rich areas. This mechanism relies on the so-called NS bit, an extension of the AMBA3 AXI Advanced Peripheral

Bus (APS), a peripheral bus that is attached to the system bus using an AXI-to-APB-bridge. The NS bit distinguishes those instructions stemming from the secure area and those stemming from the rich area. Access to the NS bit is protected by a gatekeeper mechanism in the Operating System. The Operating system thus distinguishes between user space, kernel space and secure space. Only authorized software is running in secure space, without interference from user or kernel space. TrustZone defines a communication abstraction for the interaction between programs running in the rich area that act as clients, and programs running in the secure area that act as servers. This client-server communication is session-based (each session is bound to programs in the rich area); no state is kept in the secure area across sessions (any state must be explicitly stored in memory or on secondary storage).

TruztZone is implemented in ARM popular processors: Cortex-A9 and Cortex-A15 (e.g., powering platforms such as Samsung Exynos and Nvidia Tegra series) that can readily be used in laptops, smartphones, PDAs and tablets that researchers use or delpoyed in the field to extend data loggers. The description of the platform (hardware, software combination) that we are using for the ScanDB V1 prototype can be found in [2].

In ScanDB V1, running on Trustzone, the rich area provides the data platform that rely on trusted storage primitives. Authorized storage components running in the secure area can encrypt data and store keys in a tamper-resistant chip thus guaranteeing data confidentiality; They can verify that the encrypted data that has been stored corresponds to the data that was written, thus guar- anteeing data integrity; They can replicate data stored locally on several remote instances (e.g., on the cloud), thus providing availability and durability. Note that availability and durability come at the cost of performance. Exploring this trade-off is a topic for future work. ScanDB V1 achieves data integrity and confidentiality on top of TrustZone.

References

[1] N. Anciaux, P. Bonnet, L. Bouganim, B. Nguyen, I. Sandu Popa, and P. Pucheral. Trusted cells: A sea change for personal data services. CIDR, 2013.
[2] J. Gonzalez and P. Bonnet. Towards an open framework leveraging a trusted execution environment. In Cyberspace Safety and Security. Springer, 2013.
[3] Jaehong Park and Ravi Sandhu. The uconabc usage control model. ACM Trans. Inf. Syst. Secur., 7(1):128{174, February 2004.