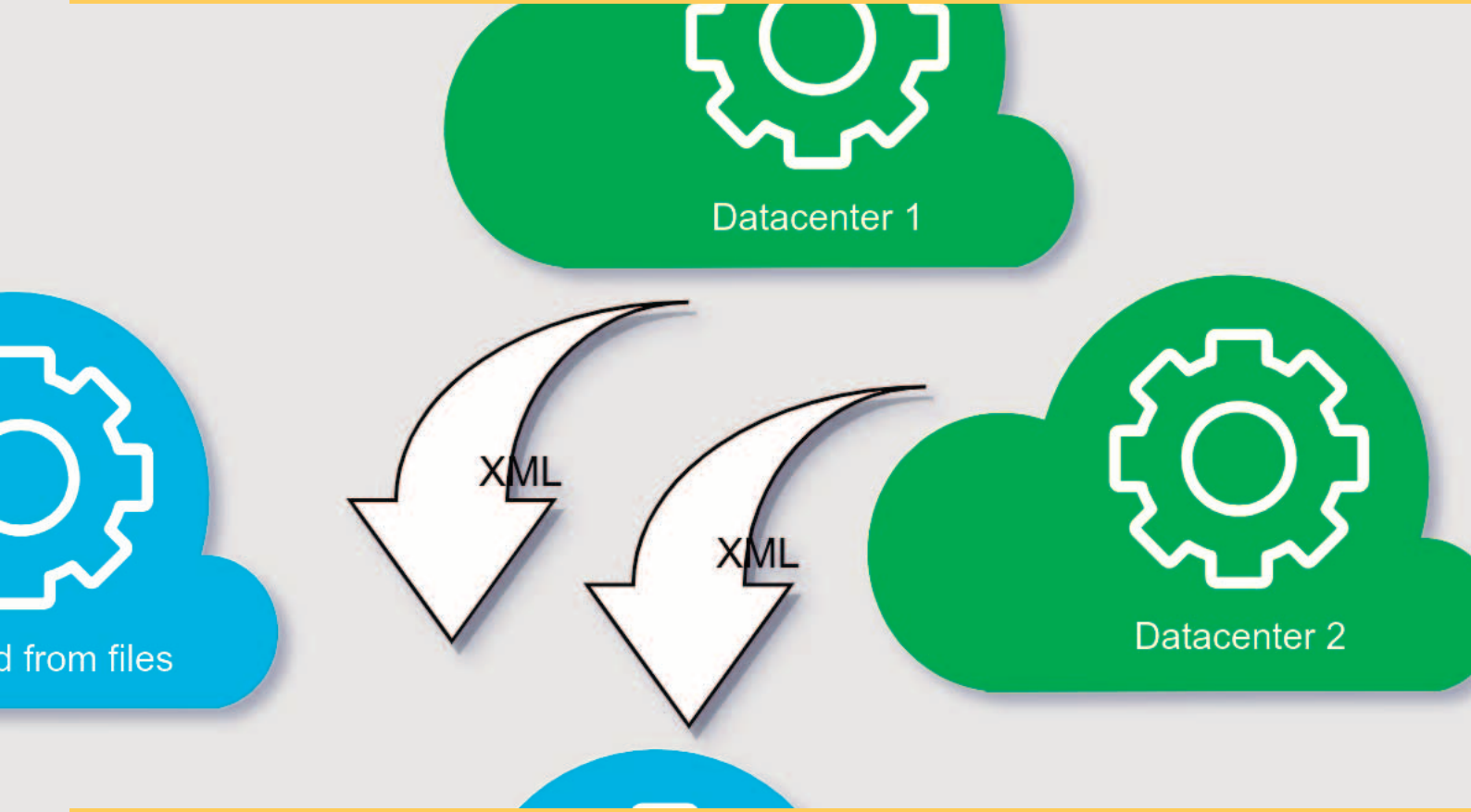
ISSN 2039-7941
anno 2024 numero 478



RAPPORTI TECNICI INGV

Exist-fdsn-station user manual 1.0

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

478

Direttore Responsabile
Valeria DE PAOLA

Editor in Chief
Milena MORETTI (editorinchief.collane-editoriali@ingv.it)

Editorial Board
Laura ALFONSI (laura.alfonsi@ingv.it)
Raffaele AZZARO (raffaele.azzaro@ingv.it)
Christian BIGNAMI (christian.bignami@ingv.it)
Simona CARANNANTE (simona.carannante@ingv.it)
Viviana CASTELLI (viviana.castelli@ingv.it)
Luca COCCHI (luca.cocchi@ingv.it)
Rosa Anna CORSARO (rosanna.corsaro@ingv.it)
Luigi CUCCI (luigi.cucci@ingv.it)
Lorenzo CUGLIARI (lorenzo.cugliari@ingv.it)
Alessia DI CAPRIO (alessia.dicaprio@ingv.it)
Roberto DI MARTINO (roberto.dimartino@ingv.it)
Domenico DI MAURO (domenico.dimauro@ingv.it)
Domenico DORONZO (domenico.doronzo@ingv.it)
Filippo GRECO (filippo.greco@ingv.it)
Alessandro IAROCCI (alessandro.iarocci@ingv.it)
Marcello LIOTTA (marcello.liotta@ingv.it)
Mario MATTIA (mario.mattia@ingv.it)
Daniele MELINI (daniele.melini@ingv.it)
Anna NARDI (anna.nardi@ingv.it)
Lucia NARDONE (lucia.nardone@ingv.it)
Marco OLIVIERI (marco.olivieri@ingv.it)
Nicola PAGLIUCA (nicola.pagliuca@ingv.it)
Pierangelo ROMANO (pierangelo.romano@ingv.it)
Maurizio SOLDANI (maurizio.soldani@ingv.it)
Sara STOPPONI (sara.stopponi@ingv.it)
Umberto TAMMARO (umberto.tammaro@ingv.it)
Andrea TERTULLIANI (andrea.tertulliani@ingv.it)
Stefano URBINI (stefano.urbini@ingv.it)

Ufficio Editoriale
Francesca DI STEFANO - Coordinatore
Rossella CELI - Segreteria di Redazione
Produzione e grafica-redazionale
Barbara ANGIONI
Massimiliano CASCONE
Rossella CELI
Francesca DI STEFANO
Patrizia PANTANI

REGISTRAZIONE AL TRIBUNALE DI ROMA N.174 | 2014, 23 LUGLIO
© 2014 INGV Istituto Nazionale di Geofisica e Vulcanologia | Rappresentante legale: Carlo DOGLIONI
Sede: Via di Vigna Murata, 605 | Roma

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

ISSN 2039-7941
anno **2024** numero **478**

RAPPORTI TECNICI INGV

Exist-fdsn-station user manual 1.0

Stefano Pintore

INGV | Istituto Nazionale di Geofisica e Vulcanologia, Osservatorio Nazionale Terremoti

Accepted 13 December 2023 | *Accettato 13 dicembre 2023*
How to cite | *Come citare* S. Pintore, (2024). Exist-fdsn-station user manual 1.0. Rapp. Tec. INGV, 478: 1-28, https://doi.org/10.13127/rpt/478

Cover | *In copertina* Example of possible use of fdsn-station-sync-xml.py

478

INDEX

Abstract

Exist-fdsn-station is an open source software that implements the standard fdsnws/station web service, integrating the application into a native XML database containing seismic stations metadata in the StationXML file format. Through its HTTP Application Programming Interface, extended with the PUT method for writing, this software can be used as a RESTful microservice. The software is publicly available and licensed under a General Public License. This manual describes all the operational phases, from installation to distribution in a production environment, for using exist-fdsn-station to store a set of StationXML files and exposing them efficiently with a standard fdsnws/station webservice.

Keywords Web service; fdsnws; Seismic station metadata

Introduction

This is the reference manual on how to install and deploy the exist-fdsn-station software. This software offers an XML-based database for seismic stations metadata, compliant with the standard FDSN StationXML format [FDSN, 2012] and accessible with the standard FDSN web service fdsnws/station [FDSN, 2013]. The complete general framework of use of this software, the most relevant internal aspects, and a thorough performance evaluation is extensively presented in [Pintore and Danecek, 2024], here we recall only some basic information.

Historically, seismic station metadata description has been standardized by the Working Group on Digital Data Exchange, established within the International Association for Seismology and Physics of the Earth's Interior (IASPEI), Commission on Practice, then by the International Federation of Digital Seismograph Networks (FDSN). The first release of the Standard for the Exchange of Earthquake Data (SEED) described both waveform data and station metadata assembled into binary files called SEED volumes. SEED 2.3, released in 1992, added features enabling the usage of data-only files (miniSEED) which consist of a simpler stream of data records suitable for real-time transmission. SEED 2.4, was the final version, released in 2012 [Ahern et al., 2012]. The same year, to encompass the restrictions of the binary fixed lengths fields of SEED format, the FDSN decided to create the StationXML format as a replacement and extension of the SEED standard.

The purpose of the FDSN StationXML schema was to define an Extensible Markup Language (XML) representation of the most important and commonly used structures of the SEED 2.4 metadata, extending them mostly with details and references to better describe entities such as networks, stations and the instrumentation. The structure of the StationXML document is a generalised tree. Its root element <FDSNStationXML> contains, nested one inside the other, the elements, which describe the <Network>, the seismic <Station>, the recorded data <Channel>. The <Channel> element is identified by channel, location code and a time span (epoch), it provides parameters such as geographic location and sensor or component orientation. The <Channel> element contains also the complete system transfer function of the digitally converted signal, represented by the <Response> element including a cascade of the composing stages, i.e. the transfer functions of corresponding sensing and acquisition components.

The StationXML format is suitable for transmission via web services and for this purpose the FDSN released the first version of the fdsnws/station web service specification in 2013,

together with the companion fdsnws/dataselect which provides the waveform data in miniSEED. The exist-fdsn-station software not only adheres to the fdsnws/station standard but implements an extension of the standard API specification, permitting also writes into its database using HTTP protocol. This database falls into the class of document-oriented databases, more precisely those based on XML [Moniruzzaman and Hossain, 2013] and, to our knowledge, is the first implementation of the fdsnws/station web service that uses this functionality.
The ability to store and provide seismic station information using the StationXML document format as input and output makes it possible to integrate exist-fdsn-station into a flexible microservice-based architecture.
The necessity of this kind of software has been anticipated in [Danecek et al., 2021], as a founding part of a new workflow for the seismic station metadata management in INGV. In particular it will substitute the database "seisnet", strictly coupled with the SEED metadata definitions, backend of the still currently used but becoming obsolescent, seismic station metadata management interface named Seisface [Pintore et al., 2012].
The exist-fdsn-station software is an application running into an eXist-db [Siegel and Retter, 2014] database, users should consider becoming familiar also with the eXist-db software, in particular with its Package Manager and with the eXide integrated editor. Minimal examples of using both of these tools are described on the following pages. The exist-fdsn-station code [Pintore, 2023], available under the Open Source AGPL-3.0 license, is distributed by the github service, and includes the files necessary to create the necessary docker image and run an eXist-db docker container [Docker Inc., 2023].
In the rest of this manual, commands executed and relative output in the user console have the following paragraph style:

```
$ docker build. -t exist-fdsn-station
```

Every important piece of advice is emphasized using Bold characters. Keywords to be used literally are enclosed in quotes, like the next file path example: "config/settings.json".

1. Installation

There is more than one way to install eXist-db and the exist-fdsn-station web service. You can install the eXist-db database and application in the provided containerized package, following the instructions presented here. Otherwise, if you prefer a standard eXist-db installation, refer to the eXist-db documentation [eXist-db Project, 2014]. In the latter case, after successfully installing eXist-db stand-alone, follow the instructions presented in "Installing using the packages" subchapter, to install Exist-fdsn-station using the package system. Containerized package can run wherever you can install docker and docker compose, these are the minimum dependencies. However, it has not been tested on every theoretically supported architecture, and customization of the compose.yaml file may be necessary in your particular case. The software tools needed to install and use exist-fdsn-station, or required to follow the examples in this manual, are listed in Table 1.

Software	Version	Purpose
docker	>=20.10.24	Build and run the service
docker compose	>=2.17.2	Build and run the service
ant	>=1.10.12	Build packages
curl	>=7.68.0	Run examples
git	>=2.25.1	Clone source from repository

Table 1 Additional software requirements, the versions shown has been tested, even older versions may reasonably work.

1.1 Get the source

Whichever method you choose for installation, you will need to obtain the source code before proceeding. To get the source code, download a copy of the project from the github repository accessing via web the URL: “https://github.com/INGV/exist-fdsn-station”, or clone it directly:

```
$ git clone https://github.com/INGV/exist-fdsn-station
```

1.2 Test run with exist-fdsn-station

Assuming your system account has an executable docker installation, for a quick application test, you can build the image by running the next command from the project directory:

```
$ docker build . -t exist-fdsn-station
```

then start the container using the image just built:

```
$ docker run -d -p 127.0.0.1:80:8080 —name exist-fdsn-station exist-fdsn-station
```

The eXist-db server, with exist-fdsn-station installed into, will be up and running in a short time. Access its Dashboard page at “http://127.0.0.1/exist” using your web browser, it will be displayed as in Figure 1.
Change the 80 port number in the “docker run” command if it is already used by a different service on your system.

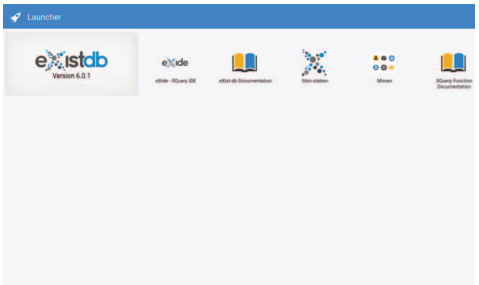


Figure 1 eXist-db dashboard page with icons pointing to the applications.

Now that the eXist-db interface is accessible, click on the “fdsn station” icon to activate the application. The application home page will look like in Figure 2. User “admin” password is empty.

Persistence of data is not guaranteed using this installation method, and you need to provide your preferred HTTP proxy server to forward queries to the standard path. Now that you have

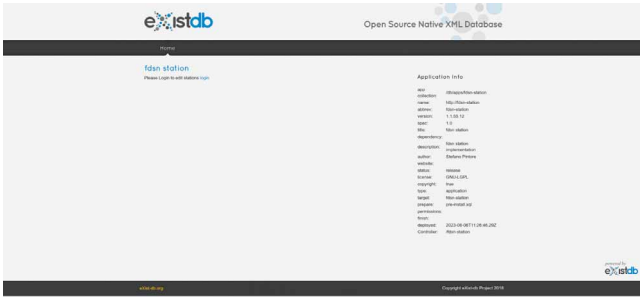


Figure 2 Home page of exist-fdsn-station web interface.

just completed a test run, you can uninstall the whole thing without leaving a trace on your system. You can stop and remove container and image with the next commands:

```
$ docker stop exist-fdsn-station
$ docker rm exist-fdsn-station
$ docker image rm exist-fdsn-station
```

1.3 Installation with docker-compose

The recommended and persistent installation procedure is based on the docker compose command, driven by the compose.yaml file. The installation will store the database on disk in the “data” directory, but you must create it manually before install. If change of path data directory is needed, you need to change compose.yaml file accordingly. From the project directory you can activate the build and start the service using docker compose:

```
$ mkdir data
$ docker compose up -d
[+] Running 2/2
 ✓ Container exist-fdsn-station Started
 ✓ Container fdsn-station-proxy-1 Started
```

This command starts two containers, one with the database and the application, the second with the nginx web server front-end [F5 Nginx, 2023], you can check their running state with:

```
$ docker compose top
exist-fdsn-station
```

```
exist-fdsn-station-proxy-1
UID      PID     PPID    C   STIME  TTY  TIME      CMD
root     107349  107313  0   16:02  ?    00:00:00  nginx: master process nginx -g daemon off;
systemd+ 107513  107349  0   16:02  ?    00:00:00  nginx: worker process
systemd+ 107514  107349  0   16:02  ?    00:00:00  nginx: worker process
systemd+ 107515  107349  0   16:02  ?    00:00:00  nginx: worker process
systemd+ 107516  107349  0   16:02  ?    00:00:00  nginx: worker process
```

After a while, your server will be up and running, access it at “http://127.0.0.1/exist”. While the

administrator can securely access the two endpoints, it is safer for only the nginx server to be exposed to public access, preferably behind another web proxy. With docker compose you can start or stop the two servers independently. In this way it is possible to block the proxy and leave the database reachable only for administration operations:

```
$ docker compose stop proxy
[+] Running 1/1
 ✓  Container exist-fdsn-station-proxy-1          Stopped

$ docker compose top
exist-fdsn-station
UID   PID    PPID   C   STIME  TTY  TIME      CMD
root  12199  12179  1   09:17  ?    00:00:03  java org.exist.start.Main jetty java -jar start.jar jetty
```

1.4 Basic administrative tasks

Administrator credentials are required to perform administrative tasks in eXist-db. The administrator user is called “admin” and has a blank default password. Typical administrative tasks are the user management and the packages installation. To access the Dashboard menu to start the User Manager or the Package Manager you must log in, clicking on the top right link, then fill the login form shown in Figure 3.

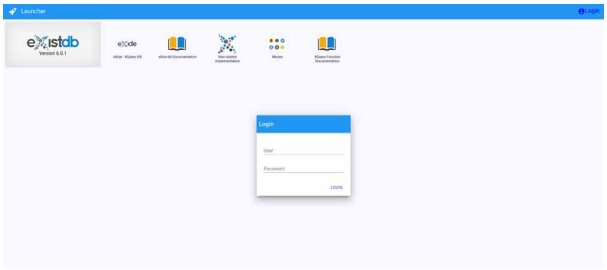


Figure 3 Login to access the Dashboard menu.

Immediately after login, the Dashboard menu appears on the left to start the eXist-db administrative applications, as in Figure 4. Use the User Manager application to change the “admin” password. Installation of the exist-fdsn-station packages automatically creates the “fdsn” user, you need to change its default “fdsn” password too.

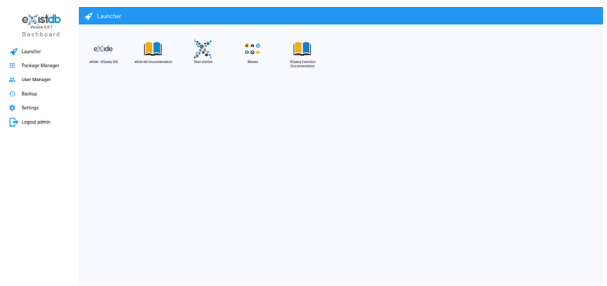


Figure 4 The Dashboard Menu of administrative applications.

1.5 Uninstall

The database uses the “data” directory as a container local volume. Using this precaution, even after removal of both docker container and image, database and application persist safely on disk. Therefore, to completely remove the application installed using “docker compose”, you need to stop and remove the containers, remove their images and finally delete the data directory. The files in the data directory are owned by the superuser, you will probably have to increase your privileges to actually delete them.

```
$ docker compose down
[+] Running 3/3
 ✓  Container exist-fdsn-station-proxy-1      Removed                    0.4s
 ✓  Container exist-fdsn-station              Removed                    0.9s
 ✓  Network exist-fdsn-station_station-net    Removed
docker image rm nginx exist-fdsn-station-exist-fdsn-station
```

1.6 Building the packages

The exist-fds-station consists of two packages: fdsn-station and fdsn-station-data. The first contains the application files, the second contains data and index definitions. Their names follow the following name convention:

- fdsn-station-data-{majorversion}{minorversion}.xar.
- fdsn-station-{majorversion}{minorversion}{microversion}{build}.xar,

The package files, compatible with eXist-db, are in EXPath format [EXPath Community Group, 2021].
Installing the “ant” software is a prerequisite to build the packages. Then you can build the packages, in the form of compressed “.xar” files, invoking “ant” from the project directory.

```
$ ant -f build.xml
Buildfile: /home/stefano/gitwork/exist-fdsn-station/build.xml

xar:
   zip] Building zip: /home/stefano/gitwork/exist-fdsn-station/build/fdsn-station-1.1.55.12.xar

BUILD SUCCESSFUL
Total time: 0 seconds
```

```
$ ant -f fdsn-station-data/build.xml
Buildfile: /home/stefano/gitwork/exist-fdsn-station/fdsn-station-data/build.xml

xar:
   [zip] Building zip: /home/stefano/gitwork/exist-fdsn-station/fdsn-station-data/build/fdsn-station-data-1.1.xar

BUILD SUCCESSFUL
Total time: 0 seconds
```

1.7 Installing using the packages

If you installed eXist-db on your own, you need the two package files to proceed with this kind of installation. Since distribution of binary packages is not currently planned, you need to compile them from source, as shown above.
Since the installation of fdsn-station depends on the presence of the fdsn-station-data package in the database, you must **first install fdsn-station-data**, otherwise the installation will fail.
Launch the Package Manager application from the Launcher, to install the packages one at a time. After startup, Package Manager shows the list of installed and available packages. Use the Upload button to install packages from disk, see Figure 5 and 6.

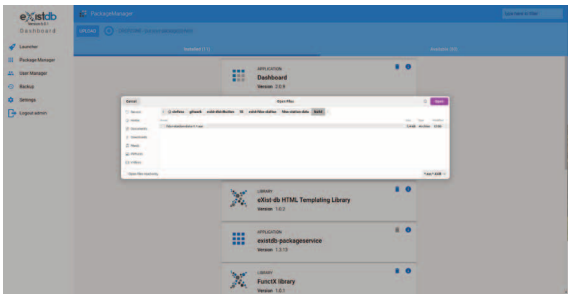


Figure 5 Choose the fdsn-station-data package file from your file system.

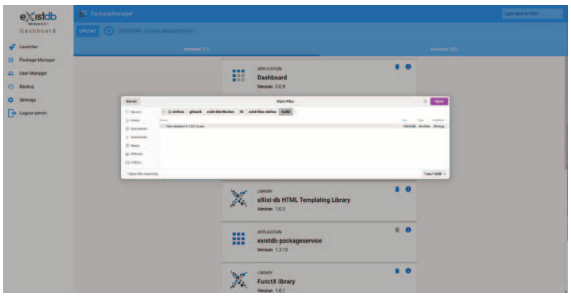


Figure 6 Choose the fdsn-station package file from your file system.

The fdsn-station-data package contains only the initial collection and its index definitions. This collection will be filled with your stations data, so when the system will be in production you should **avoid reinstalling the fdsn-station-data package, otherwise your data will be deleted**.

1.8 Updates

Updates of the package follow the same procedure for every kind of installation. Updates of the fdsn-station package are data safe and do not require updating the fdsn-station-data package. Future versions of the fdsn-station package that eventually require changes to the data collection will also require an updated fdsn-station-data package and will fail installation otherwise.

2. Station Database maintenance

The application interface exposes some basic database maintenance operation through the “Manage” page. Maintenance operation are subjected to authorization, therefore administrators need to click on the Login link in the application home page (see Figure 2), then the home page will change as in Figure 7.

Figure 7 The exist-fdsn-station home after login.

After clicking the Home link to obtain the menu, choose Manage to visit the maintenance page of exist-fdsn-station, see Figure 8. Beware, there are no confirmation dialog windows, therefore **all buttons you press execute immediately** the action required.

Figure 8 Access to the Manage page.

2.1 Purge the database

Sometimes it is necessary to delete all stations from the database, one of these times is immediately after installation. A dummy station is in fact contained in the fdsn-station-data package, it is necessary to remove it before starting to use the database with your data. To purge the station database, deleting all stations, click on the “Purge” button. Beware, **you cannot undelete after clicking**.

2.2 Cache clean

There is a kind of cache system implemented in the application, if you moved or deleted some station document using the eXide application or accessing the database collection with other clients, you probably broke this system. Minor problems can be solved using the “Empty” button to reset the cache. Typical symptoms of a broken cache are inconsistent results of queries from the web service. This operation is completely safe for data in the station collection.

2.3 Fix database

Major problems occurring to the cache system can be fixed resetting the whole cache system, using the “Fix” button. Typical symptoms of a broken cache are inconsistent results of queries from the web service, persisting after a Cache Clean attempt. If Cache Clean did not get the expected results, you can still try to Fix the database. This operation is safe for data, but could change data modification time of documents used by queries with the parameter updatedafter.

2.4 Touch database

This function permits to update all modification time of all documents in the database at the same time. This operation is safe for data, but will change data modification time of documents used by queries with the parameter updatedafter. Use it whenever you want to force all your data to a given update time.

2.5 Log verbosity

To change the logs verbosity of the application you must check one or more desired levels, then click apply. Changes are applied immediately, starting from the next query to the service. Remote access of authenticated users and changes to the stations collection are always logged whatever the settings.

- Enable log: add to the log of the application only error 400. It is a recommendable default.
- Enable debug: enable logging of internal calls of the application, use it only when trying to debug the software.
- Enable query: enables GET and POST query logs by the service. Might be useful for statistics, but could fill up your disk if inadvertently left enabled for too long, without a proper log files rotation policy. When query logs is disabled, you can still see query logs of the nginx proxy. Change its configuration file if you need the POST request content, not enabled in the provided “proxy/nginx.conf”.

2.6 Where are my logs?

The exist-fdsn-station logs are in the same place where eXist-db logs go. If you are using the docker compose installation procedure you can see all the logs using:

```
$ docker compose logs
```

or, separating for the different services:

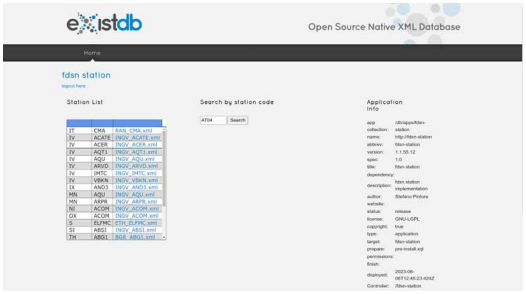
```
$ docker compose logs exist-fdsn-station
$ docker compose logs proxy
```

Logs can be rotated by docker daemon, instructions on how to configure it are beyond the scope of this manual.

2.7 Where are my files?

While the web service can provide responses to queries in the form of StationXML, it will never provide the original stored files. Only using eXide or through the application interface, you can view and download the original StationXML browsing the fdsn-station-data collection. To get the chosen file just click on the link in the station list present in the home page, as in Figure 9. The file content is slightly different from the response file you can get querying the service for the given station code at the response level, at least because it contains a different creation date. The content could differ also from the original file you uploaded using the web service PUT method, due to some normalization operation the application optionally performs during the input phase.

Figure 9 The file list and the search field.



When the list is very long it may be useful to use the “Search” function to find the input files using the station code as the search key. More search keys could be available in future releases of the software.

3. Data Management

Having removed the dummy station by trying the “Purge” of the database, it’s time to load some station.xml files to continue playing around or to put in production your new fdsnws/station web service. The better way to load data in a production environment is using the PUT HTTP method, in this way data load can be fully automated.

3.1 PUT your stations in the database

In order to upload station files, you need a web client capable of submitting a PUT request with a payload to the webservice. PUT is a method subject to authorization, you need to know the “fdsn” user credentials. You can store in the database station files coming from different sources and, as station names are not uniquely attributed to a provider, it’s **mandatory** to use a **naming convention** for files. The file names need to be in the format: PROVIDER_STATION.xml where PROVIDER is a string used to identify the provider of the metadata station file and STATION is a string equal to the seismic station code.
Notice that you can use any PROVIDER string, there is not problem in mixing stations with different PROVIDER when necessary. It’s possible to insert in the database two files with the same station name and different PROVIDER. The service will return the metadata present in one or both station files based on the query parameters that will be requested.

Here is an example of data input using curl, passing through the nginx proxy server:

```
$ curl -X PUT “http://127.0.01:80/fdsnws/station/1/query?” -H “accept: application/xml” -H “Content-Type: application/octet-stream” -H “filename: INGV_ABSI.xml” —data-binary @Station/INGV_ABSI.xml -o output.xml -i -v -fdsn:password
```

and another example using curl directly addressing the eXist-db host, notice the different URL:

```
$ curl -X PUT “http://172.17.0.2:8080/exist/apps/fdsn-station/fdsnws/station/1/query?” -H “accept: application/xml” -H “Content-Type: application/octet-stream” -H “Expect:” -H “filename: INGV_ABSI.xml” —data-binary @Station/INGV_ABSI.xml -o output.xml -i -v -fdsn:password
```

3.2 DELETE your stations

Usually stations are inserted and updated overwriting them, but if you really want to delete one station, you can use the DELETE HTTP method, passing the filename header. Example:

```
$ curl -v -H “filename: INGV_ACER.xml” -X DELETE “http://127.0.0.1:80/fdsnws/station/1/query?” -u fdsn:password
*   Trying 127.0.0.1:80...
* Connected to 127.0.0.1 (127.0.0.1) port 80 (#0)
* Server auth using Basic with user ‘fdsn’
> DELETE /fdsnws/station/1/query? HTTP/1.1
> Host: 127.0.0.1
> Authorization: Basic ZmRzbjpmZHNu
> User-Agent: curl/7.81.0
> Accept: */*
> filename: INGV_ACER.xml
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx
< Date: Wed, 07 Jun 2023 09:18:22 GMT
< Content-Type: text/xml;charset=utf-8
< Transfer-Encoding: chunked
< Connection: keep-alive
< Vary: Accept-Encoding
< Cache-Control: max-age=60, must-revalidate
< X-XQuery-Cached: true
<
* Connection #0 to host 127.0.0.1 left intact
```

If you need to remove a whole network, pertaining to a given PROVIDER, you have to use another syntax, passing on the URL the PROVIDER string and the network code:

```
$ curl -v  -X DELETE “http://127.0.0.1:80/fdsnws/station/1/query?provider=INGV&net=IV” -u fdsn:fdsn
*   Trying 127.0.0.1:80...
* Connected to 127.0.0.1 (127.0.0.1) port 80 (#0)
* Server auth using Basic with user ‘fdsn’
> DELETE /fdsnws/station/1/query?provider=INGV&net=IV HTTP/1.1
> Host: 127.0.0.1
> Authorization: Basic ZmRzbjpmZHNu
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx
< Date: Wed, 07 Jun 2023 09:40:03 GMT
< Content-Type: text/xml;charset=utf-8
```

```
< Transfer-Encoding: chunked
< Connection: keep-alive
< Vary: Accept-Encoding
< Cache-Control: max-age=60, must-revalidate
< X-XQuery-Cached: true
<
* Connection #0 to host 127.0.0.1 left intact
```

A different syntax is available for removing all the networks of the given provider in a single request, using a wildcard to match every network code:

```
$ curl -v -X DELETE “http://127.0.0.1:80/fdsnws/station/1/query?provider=INGV&net=*” -u fdsn:fdsn
* Trying 127.0.0.1:80...
* Connected to 127.0.0.1 (127.0.0.1) port 80 (#0)
* Server auth using Basic with user ‘fdsn’
> DELETE /fdsnws/station/1/query?provider=INGV&net=* HTTP/1.1
> Host: 127.0.0.1
> Authorization: Basic ZmRzbjpmZHNu
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Server: nginx
< Date: Wed, 07 Jun 2023 09:42:39 GMT
< Content-Type: text/xml;charset=utf-8
< Transfer-Encoding: chunked
< Connection: keep-alive
< Vary: Accept-Encoding
< Cache-Control: max-age=60, must-revalidate
< X-XQuery-Cached: true
<
* Connection #0 to host 127.0.0.1 left intact
```

For a complete reference of the extension of the API implemented in exist-fdsn-station see Table 2.

Method	Parameter	Header	Description
PUT	None requested	filename	Add a station to the database. The filename of the stationXMLpassed is in the format PROVIDER_STATIONCODE.xml.
DELETE	None requested	filename	Delete a station, the name of the stationXML to be removed is passed in the format PROVIDER_STATIONCODE.xml.
DELETE	provider	None requested	Provider of the stations to be removed, as per the prefix of the station file previously inserted.
	net	None requested	Select one network code to delete all stations of the network of the given provider, '*' to delete all stations of the given provider.

Table 2 fdsnws/station API extension provided.

3.3 Managing data using eXide

Managing data directly using the eXide application is possible, but it is an advanced technique that could potentially lead to data loss. If you are brave enough, and you have made a backup of your database using eXist-db tools or, at least, copied your "data" directory after stopping containers, you can try to speed up some input operations by loading all the datafiles you need in a single interactive operation.
StationXML files are loaded by the PUT operation into the "/db/apps/fdsn-station-data/Station" collection. You can manually upload files to this library using eXide, as long as they have been named following the naming convention specified above. In the same way you can update or remove stations as well.
Using PUT ensures some check and normalization of the files, while the manual load using eXide does not change at all your files. **After manual editing of the station collection, it's mandatory to "Fix" the database** using the application management interface as described above.
The eXide editor launcher is available in the eXist-db Dashboard (see Figure 1), click on its icon to launch it. From the file menu, choosing the "Manage" entry you can browse the "/db/apps/fdsn-station-data/Station" collection to upload your files, see Figure 10. Navigate the structure of the database collections, using the left panel, to directly edit data.
Please note that using this tool you can also edit all application files. Although this method of loading station files may seem advantageous in some use cases, it is advisable to use it only in a phase of learning how to use the software, until the adoption of an automatic procedure to fill the database that takes advantage of the API.

Figure 10 Access the eXide editor.

3.4 Get responses from the web service

Just after station loading, querying your service will start to give some meaningful results. The entry points differ depending on the server you interrogate, see Table 3 for reference.

container	entry point
exist-fdsn-station	http://INTERNAL_IP:8080/exist/apps/fdsn-station/fdsnws/station/1/query?
exist-fdsn-station-proxy-1	http://127.0.0.1:80/fdsnws/station/1/quer?

Table 3 Entry points of the fdsnws/station service's.

Notice that INTERNAL_IP is the IP assigned on the docker network to the exist-fdsn-station. You can find it with:

```
$ docker network inspect exist-fdsn-station_station-net
```

Access the service path "/fdsnws/station/1" to obtain a human readable description of the service specification. This description is generated automatically starting from the formal definition of the API, contained in the application.wadl file, that is also directly accessible at "/fdsnws/station/1/application.wadl". Table 4 contains a summary of the implemented API conformance to the fdsnws/station standard.

Parameter	Support	Implemented	Description
starttime	Required	Y	Limit to metadata epochs starting on or after the specified start time.
endtime	Required	Y	Limit to metadata epochs ending on or before the specified end time.
startbefore	Optional	Y	Limit to metadata epochs starting before specified time.
startafter	Optional	Y	Limit to metadata epochs starting after specified time.
endbefore	Optional	Y	Limit to metadata epochs ending before specified time.
endafter	Optional	Y	Limit to metadata epochs ending after specified time.
network	Required	Y	Select one or more network codes. Can be SEED network codes or data center defined codes. Multiple codes are comma-separated.
station	Required	Y	Select one or more SEED station codes. Multiple codes are comma-separated.
location	Required	Y	Select one or more SEED location identifiers. Multiple identifiers are comma-separated. As a special case "—" (two dashes) will be translated to a string of two space characters to match blank location IDs.
channel	Required	Y	Select one or more SEED channel codes. Multiple codes are comma-separated.
minlatitude	Required	Y	Limit to stations with a latitude larger than or equal to the specified minimum.
maxlatitude	Required	Y	Limit to stations with a latitude smaller than or equal to the specified maximum.
minlongitude	Required	Y	Limit to stations with a longitude larger than or equal to the specified minimum.
latitude	Optional	Y	Specify the latitude to be used for a radius search.

longitude	Optional	Y	Specify the longitude to be used for a radius search.
minradius	Optional	Y	Limit results to stations within the specified minimum number of degrees from the geographic point defined by the latitude and longitude parameters.
maxradius	Optional	Y	Limit results to stations within the specified maximum number of degrees from the geographic point defined by the latitude and longitude parameters.
level	Required	Y	Specify the level of detail for the results.
includerestricted	Optional	Y	Specify if results should include information for restricted stations.
includeavailability	Optional	N	Specify if results should include information about time series data availability.
matchtimeseries	Optional	N	Limit to metadata where selection criteria matches time series data availability.
updatedafter	Optional	Y	Limits to metadata updated after specified time updates are data centre specific. While this option is not Required it is highly recommended due to usefulness.
format	Optional	Y,(+)	Specify format of result, either 'xml' (default) or 'text'. If this parameter is not specified the service must return StationXML. Non-standard 'json' and 'geojson' are also supported.
nodata	Optional	Y	Select status code returned for "no data", either '204' (default) or '404'.
minradiuskm	Optional	(+)	Limit results to stations within the specified minimum number of kilometers from the geographic point defined by the latitude and longitude parameters. Non-standard extension.
maxradiuskm	Optional	(+)	Limit results to stations within the specified maximum number of kilometers from the geographic point defined by the latitude and longitude parameters. Non-standard extension.

Table 4 fdsnws/station specification implemented by exist-fdsn-station. (+) in support column indicates a proprietary extension of the standard.

4. Deployment in production

4.1 Hardware requirements

The exist-fdsn-station software is not particularly greedy for resources. A single deployment with 1000 seismic stations runs with good performance in a virtual server with 4 CPU Core, 8GB RAM, 40 GB of disk space. Similar performances, with almost 20000 seismic stations, are reached in a virtual server with 16 CPU Core, 32GB RAM, 50 GB of disk space.

4.2 Customization before deployment

The project file "config/settings.json" contains the application settings, it is copied to the image in the path "/db/apps/fdsn-station/config/settings.json" during creation, it can be changed before this operation. The settings file can also be edited on a running system by accessing it with eXide. To customize your distribution **you should at least change the sender field** as needed. Only log level settings can be changed via the application interface. Complete documentation of the settings is summarized in Table 5.

Name	Type	Default	Meaning	Manageable by interface
enable_log	boolean	true	Enable basic logs	yes
enable_debug	boolean	false	Enable query debug	yes
enable_query_log	boolean	false	Enable query logs	yes
post_limit_rows	integer	1000	Limit of rows accepted in POST	no
translate_units	boolean	true	Filter units during PUT operation	no
remove_tz	boolean	true	Filter timezone during PUT operation	no
sender	string	INGV-ONT	Sender of file served	no
source	string	ExistDB	Source of file served	no

Table 5 Application settings.

Customization is necessary before releasing your service in public, it requires to edit manually two static files:

1. “application-wadl.xml”
2. “application-Swagger20.json”

before building the docker images, or using eXide if acting after this phase. They are contained in the “Static” project directory and they go in the “/db/apps/fdsn-station/Static” collection in eXist-db after build and installation. Find and replace the strings containing “ingv” with more appropriate content, or completely change the content of the files if needed. Their content is shown directly by the application, at:

1. http://127.0.0.1/fdsnws/station/1/application.wadl
2. http://127.0.0.1/fdsnws/station/1/swagger.json

Notice that the content shown by the service entry point “http://127.0.0.1/fdsnws/station/1/” is produced by the application interpreting the application.wadl file.

4.3 Automating database updates

The API permits to write into the database programmatically, you can use whatever instrument to exploit it for manage data into exist-fdsn-station. For user convenience, in the “bin” project directory you can find fdsn-station-sync-xml.py, a python script capable of synchronizing your database with every fdsnws/station web service source or with a directory full of station files. Use the “—help” option or inspect the code to understand its use. It could need some customization for adapting to your particular deployment.
Just notice that the option -q permit to specify a query to be submitted to the source service to select the desired station but level=channel&format=xml are mandatory query parameters to use. Option -q is ignored when source uri specifies the file:// protocol.
The file “bin/config.py” must be edited to insert the credentials needed to authenticate the script with the exist-fdsn-station destination server.

```
$ ./fdsn-station-sync-xml.py —help
Usage: fdsn-station-sync-xml.py [OPTION...]
Synchronize destination exist-fdsn-station with a fdsnws/station service, or with a directory
containing station xml files named like PROVIDER_STATIONCODE.xml
  -s, —source                 Source uri
  -d —destination             Destination url
  -p —path                    Temporary files path
  -q —query                   Query to select stations to sync [level=channel]
  -P —provider                      provider code, uppercase
  -f —force-station           Comma separated station list, if present sync only this station
  -v                          Use virtualnetworks service [INGV only]
  -l —leave_unmatched               Leave stations on destination even when not found on
source
  -e —existdb-source          Source with exist-fdsn-station url prefix
  -x —existdb-destination     Destination with exist-fdsn-station url prefix
  -n —no-save                 Do not save station files in temporary files path
  -h —help                    print this help


Example 1: fdsn-station-sync-xml.py -v -s https://webservices.ingv.it -d http://172.17.0.2:8080
-p /tmp -q “level=channel&network=*&format=xml&includerestricted=true”
Example 2: fdsn-station-sync-xml.py -s file:///opt/Station -d http://127.0.0.1:80  -p /tmp  -x
-P INGV -l”
Example 3: fdsn-station-sync-xml.py -s https://webservices.ingv.it -d http://172.17.0.2:8080
-p /tmp —force-station=”ACER,AMUR”
```

An example of using multiple instances of exist-fdsn-station that allows you to consolidate seismic station metadata from different sources into a single database using fdsn-station-sync-xml.py is shown in Figure 11. The StationXML files read from archives or obtained from fdsnws/station web services are moved to the exist-fdsn-station databases, taking advantage of the improved API.

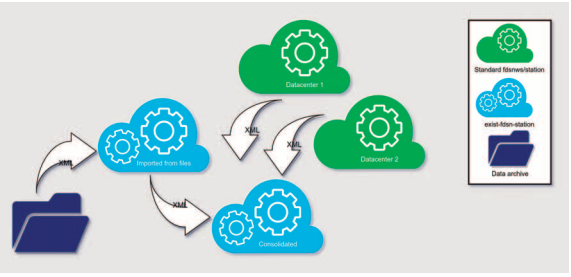


Figure 11 Example of possible use of fdsn-station-sync-xml.py. StationXML taken from disk or source Web services are copied in the destination database directly using the enhanced HTTP API.

5. Security

The docker compose solution could be adopted as is for deployment only in a trusted environment. Exposing the service on the Internet requires at least the presence of a carefully configured additional forward proxy server. Refer also to the security topic in the eXist-db documentation.

5.1 Hiding the management interface

The management interface is protected by a simple user and password mechanism. **This interface must not be directly exposed to the public**. Using a forward proxy server, forwarding only requests matching the fdsnws/station legal queries and denying the route "/exist", is the minimal security measure to take.

5.2 Exposing the fdsnws/station service

The standard service route obviously needs exposure to public requests. You need to **filter out by the forward proxy server all the PUT and DELETE incoming requests**, permitting only the GET and POST method.

6. Performance and tuning

The performance of the service can be monitored in real time using the pre-installed Monex application, accessible from the eXist-db home page, see Figure 1. The service works best when XML output is requested, while text or other formats output requires translation, with a slight penalty. This must be taken into consideration when creating applications that access the service and have critical needs in terms of response times. The eXist-db server can be configured and optimized to better suit a specific traffic load in term of concurrent users. The server responds to requests using parallel executor threads, called brokers. The brokers number is configurable as the maximum number of connections allowed in the so called connection pool, this setting can be modified editing the "etc/conf.xml" project file. Be aware that changes in the "etc/conf.xml" must be done before building the exist-fdsn-station image. Note that the broker number must be chosen together with the connection_workers parameter of the nginx configuration in the "etc/nginx.conf" file. A good place to start is with the two numbers equal to each other.

7. Conclusions

This manual guides the user starting from the step of downloading the code from its public repository and first exploring the different installation options of "exist-fdsn-station". Topics related to database maintenance, system customization and performance optimization are progressively covered. Provides guidance for increasing system security in the production environment. Station data management is illustrated, both using the web interface and taking advantage of the exclusive fdsnws/station API extension made available to allow writing to the database. We also introduced a Python tool to load StationXML files into the database. The tool can be used to synchronize exist-fdsn-station with available files on disk or directly from a source fdsnws/station web service. It allows cloning of a given fdsnws/station service or consolidation of the contents of multiple services into a single one, helping demonstrate the flexibility of exist-fdsn-station as a microservice.

References

Ahern T., Casey R., Barnes D., Benson R., Knight T., Trabant C., (2012). *SEED Reference Manual*; *Version 2.4*; Incorporated Research Institutions for Seismology: Washington, DC, USA. available at http://www.fdsn.org/pdf/SEEDManual_V2.4.pdf (last accessed 5 October 2023).

Danecek P., Pintore S., Mazza S., Mandiello A., Fares M., Carluccio I., Della Bina E., Franceschi D., Moretti M., Lauciani V., Quintiliani M., Michelini A., (2021). T*he Italian Node of the European Integrated Data Archive*. Seismological Research Letters 92, 1726–1737. https://doi.org/10.1785/0220200409

Docker Inc., (2023). https://docs.docker.com/ (last accessed 5 October 2023).

eXist-db Project, (2014). https://exist-db.org/exist/apps/doc/ (last accessed 5 October 2023).

EXPath Community Group, (2021). *Packaging System EXPath Candidate Module 9 May 2012*, available at http://expath.org/spec/pkg (last accessed 5 October 2023).

FDSN - International Federation of Digital Seismograph Networks, (2012). *StationXML schema 1.0*, available at https://www.fdsn.org/xml/station/fdsn-station-1.0.xsd (last accessed 5 October 2023).

FDSN - International Federation of Digital Seismograph Networks, (2013). *FDSN Web Service Specifications Version 1.0*, available at http://www.fdsn.org/webservices/FDSN-WS-Specifications-1.0.pdf (last accessed 5 October 2023).

F5 Nginx, (2023). https://docs.nginx.com/ (last accessed 5 October 2023).

Moniruzzaman A.B.M., Hossain S., (2013). *NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison*. International Journal of Database Theory and Application Vol. 6.

Pintore S. and Danecek P., (2024). *An XML database-based implementation of the FDSN station Web service: exist-fdsn-station*. Submitted.

Pintore S., Marcocci C., Bono A., Lauciani V., Quintiliani M., (2012). *SEISFACE: interfaccia di gestione delle informazioni della Rete Sismica Nazionale Centralizzata*, Rapp. Tec. INGV, 218: 1-30.

Pintore S., (2023). *INGV/exist-fdsn-station: Stable release v1.1.56*. Zenodo. https://doi.org/10.5281/zenodo.10034026

Siegel E., Retter A., (2014). *eXist*, O'Reilly Media, Inc. ISBN: 9781449337100.

QUADERNI di GEOFISICA

ISSN 1590-2595 https://istituto.ingv.it/it/le-collane-editoriali-ingv/quaderni-di-geofisica.html/

I QUADERNI DI GEOFISICA (QUAD. GEOFIS.) accolgono lavori, sia in italiano che in inglese, che diano particolare risalto alla pubblicazione di dati, misure, osservazioni e loro elaborazioni anche preliminari che necessitano di rapida diffusione nella comunità scientifica nazionale ed internazionale. Per questo scopo la pubblicazione on-line è particolarmente utile e fornisce accesso immediato a tutti i possibili utenti. Un Editorial Board multidisciplinare ed un accurato processo di peer-review garantiscono i requisiti di qualità per la pubblicazione dei contributi. I QUADERNI DI GEOFISICA sono presenti in "Emerging Sources Citation Index" di Clarivate Analytics, e in "Open Access Journals" di Scopus.

QUADERNI DI GEOFISICA (QUAD. GEOFIS.) welcome contributions, in Italian and/or in English, with special emphasis on preliminary elaborations of data, measures, and observations that need rapid and widespread diffusion in the scientific community. The on-line publication is particularly useful for this purpose, and a multidisciplinary Editorial Board with an accurate peer-review process provides the quality standard for the publication of the manuscripts. QUADERNI DI GEOFISICA are present in "Emerging Sources Citation Index" of Clarivate Analytics, and in "Open Access Journals" of Scopus.

RAPPORTI TECNICI INGV

ISSN 2039-7941 https://istituto.ingv.it/it/le-collane-editoriali-ingv/rapporti-tecnici-ingv.html/

I RAPPORTI TECNICI INGV (RAPP. TEC. INGV) pubblicano contributi, sia in italiano che in inglese, di tipo tecnologico come manuali, software, applicazioni ed innovazioni di strumentazioni, tecniche di raccolta dati di rilevante interesse tecnico-scientifico. I RAPPORTI TECNICI INGV sono pubblicati esclusivamente on-line per garantire agli autori rapidità di diffusione e agli utenti accesso immediato ai dati pubblicati. Un Editorial Board multidisciplinare ed un accurato processo di peer-review garantiscono i requisiti di qualità per la pubblicazione dei contributi.

RAPPORTI TECNICI INGV (RAPP. TEC. INGV) publish technological contributions (in Italian and/or in English) such as manuals, software, applications and implementations of instruments, and techniques of data collection. RAPPORTI TECNICI INGV are published online to guarantee celerity of diffusion and a prompt access to published data. A multidisciplinary Editorial Board and an accurate peer-review process provide the quality standard for the publication of the contributions.

MISCELLANEA INGV

ISSN 2039-6651 https://istituto.ingv.it/it/le-collane-editoriali-ingv/miscellanea-ingv.html

MISCELLANEA INGV (MISC. INGV) favorisce la pubblicazione di contributi scientifici riguardanti le attività svolte dall'INGV. In particolare, MISCELLANEA INGV raccoglie reports di progetti scientifici, proceedings di convegni, manuali, monografie di rilevante interesse, raccolte di articoli, ecc. La pubblicazione è esclusivamente on-line, completamente gratuita e garantisce tempi rapidi e grande diffusione sul web. L'Editorial Board INGV, grazie al suo carattere multidisciplinare, assicura i requisiti di qualità per la pubblicazione dei contributi sottomessi.

MISCELLANEA INGV (MISC. INGV) favours the publication of scientific contributions regarding the main activities carried out at INGV. In particular, MISCELLANEA INGV gathers reports of scientific projects, proceedings of meetings, manuals, relevant monographs, collections of articles etc. The journal is published online to guarantee celerity of diffusion on the internet. A multidisciplinary Editorial Board and an accurate peer-review process provide the quality standard for the publication of the contributions.

Coordinamento editoriale
Francesca DI STEFANO
Istituto Nazionale di Geofisica e Vulcanologia

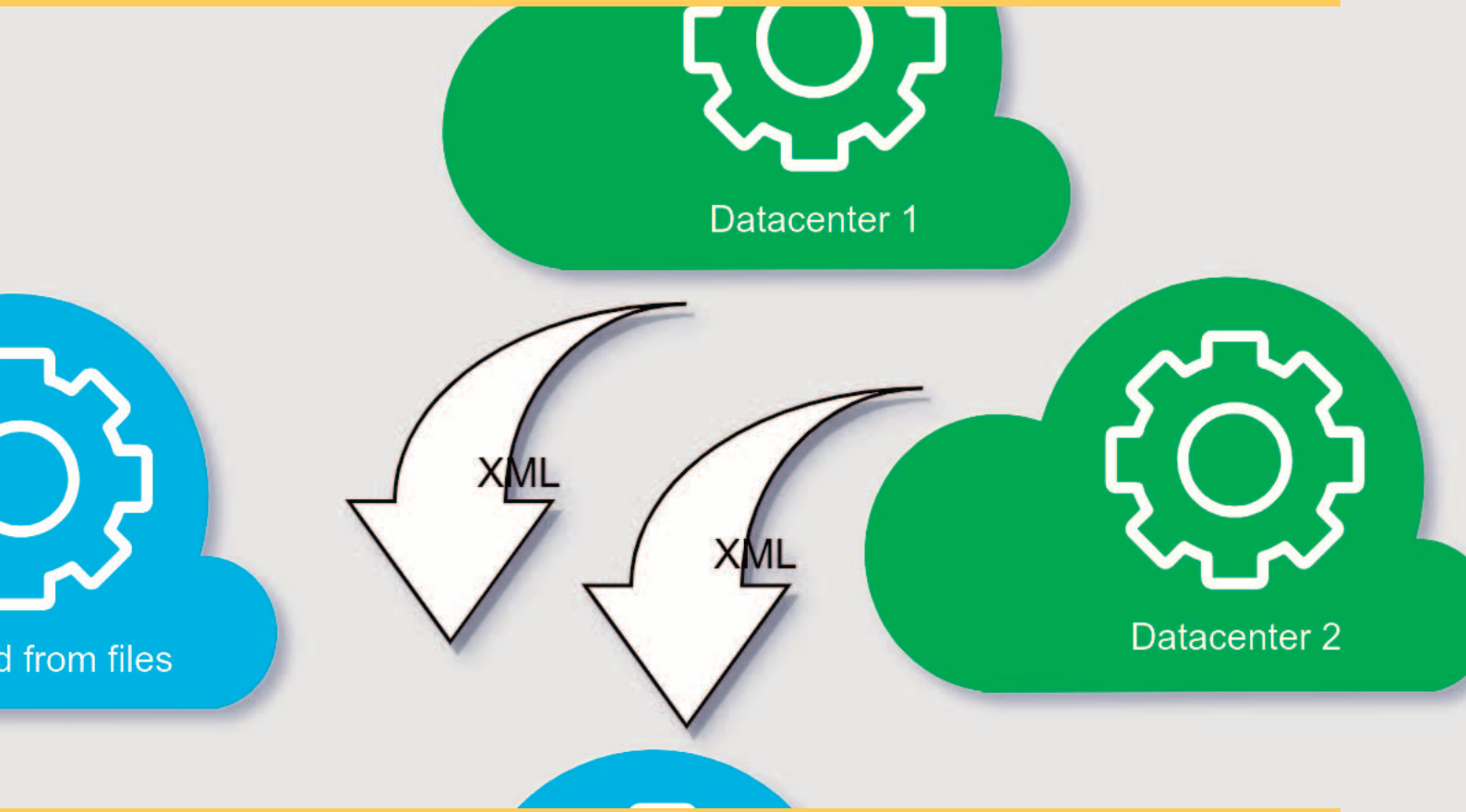
Progetto grafico
Barbara ANGIONI
Istituto Nazionale di Geofisica e Vulcanologia

Impaginazione
Barbara ANGIONI
Patrizia PANTANI
Massimiliano CASCONE
Istituto Nazionale di Geofisica e Vulcanologia

©2024
Istituto Nazionale di Geofisica e Vulcanologia
Via di Vigna Murata, 605
00143 Roma
tel. +39 06518601

www.ingv.it

Creative Commons Attribution 4.0 International (CC BY 4.0)


Datacenter 1
XML
XML
Datacenter 2
d from files