Network of European Research Infrastructures for Earthquake Risk Assessment and Mitigation

Report

Toolbox definition

Activity:	*JRA2: Tools for real time seismology, acquisition and mining*
Activity number:	*D12.1*
Deliverable:	*Toolbox definition*
Deliverable number:	*D12.1*
Responsible activity leader:	*Alberto Michelini*
Responsible participant:	*INGV*
Author:	*Alberto Michelini*

Seventh Framework Programme
EC project number: 262330

Summary

This report is motivated by the need to interface existing seismological software programs developed solely for research purposes with data available at data centers in real-time or very rapidly. The report describes the definition and the design of a general tool-box system for i.) the interfacing of waveform and parametric data with existing seismological software programs and ii.) the reformatting of the results into QuakeML (or other XML extension for seismology). The tool-box adopts and integrates available services and offers a comprehensive package to all those interested in testing some software for rapid or real-time analysis.

Introduction

Current advances of communication technology together with the advent of new and rapid ways to communicate at global scale (e.g., social networks such as twitter, facebook, google+, ….) have a deep impact not only in the way people interconnect and exchange experiences but also in the way seismological information is disseminated. This represents an important challenge to institutions and agencies in charge of seismic monitoring at a global, regional or national level. There is both a need for shortening the time delay between earthquake occurrence and the release of information, and in improving the accuracy of the analysis and thus the quality of the information being provided. The overall goal is to provide accurate and immediate information to respond to government agencies and the public and, similarly, to provide a clearer picture of the phenomenon (and of its effects) to both seismologists on duty in the seismic centers or others that are responsible to describe what has happened. This whole process requires the seismic monitoring centers to improve their routine analysis by, for example, including innovative procedures previously developed for research purposes. Many of these additional procedures can add substantial information about the ongoing phenomena and therefore improve our comprehension of the earthquake process and its effects.

Prompt availability of seismological waveform data and its derivatives (e.g., phase onset picks, peak ground motion parameters, ….) is of primary importance for real (or near-real) time seismic monitoring systems. These systems are designed to optimize the data flux and the analysis work-flow. In order to implement new software, however, it is acknowledged that a fair amount of expertise in software engineering specific to the adopted system (e.g., SeisComP3, EarthWorm, ….) is required. This is not generally part of the scientists' background skills. At this point there are two options. The first consists of selecting candidate codes without previously testing them in a real-time setting and make the investment to re-engineer them for insertion into the monitoring system(s). The second alternative is to test the software in a real-time setting (external to the seismic systems above) and in conditions that replicate as closely as possible the real-time system and, if found suitable, select them for re-engineering. The object of this report is to describe some general interfaces (toolboxes) intended to serve this second alternative.

With the goal to design this toolbox, work during the first year of NERA "work package 12" (WP12) focused on completion and analysis of a survey of the available software proposed by the WP participants. This survey is presented; it comprises selected software relevant to improving the quality of seismic monitoring and it also indicates the general requirements for the interface design. Next a general scheme (abstraction) of the general analysis work-flow architecture with emphasis on the software technologies available is described. Finally, a prototype procedure embodying the general scheme and tool-box frame is presented.

The material presented in this report has been discussed intensely among the WP12 partners during the course of the first year of NERA. In addition to email discussion, meetings have been held at the kick-off meeting in Vienna (November 17, 2010) and in Utrecht (February 7-8, 2011). This report is therefore to some extent the result of an exchange of points of view on the subject during these meetings. In addition, the meetings have been attended also by representatives of the IT community involved in NA9 - a WP dedicated to the development of web services for the seismological community. This report reflects the compromise between the work flow abstractions supported by the IT representatives and the need for some basic tool as requested by the seismological community.

Survey of existing software

This survey has been carried out to better understand the requirements of the existing software and optimize the design of the general purpose toolbox that will serve each use case - one of the primary aims of this WP.

Table 1 (see also the attached file SurveySummary.xls) provides the detail of the survey. In practice, the main focus of the survey has been to understand what is required to run each software in a completely automatic manner. Among the many quantities explored in the survey (e.g., metadata, memory requirements, libraries adopted, graphics, ….), emphasis has been given to detail the I/O requirements which are crucial for running the software.

The software surveyed can be grouped into different categories depending on the type of analysis that is performed. Most generally, the software analysis categories are earthquake location, earthquake size and mechanism (magnitude, moment tensor, focal mechanism, finite fault) and effects of the earthquake (shakemap). Note that in some cases, the selected software cannot be assigned exclusively to a single category since it may, for example, determine both earthquake location and magnitude. In addition there are differences for what concerns the scale-length of the application (i.e., local, regional or teleseismic data)

Earthquake location

Four different codes have been selected – Virtual Seismologist (*Cua and Heaton, 2007*), WaveLoc (*Maggi and Michelini, 2009*), EARLY-EST (*Lomax and Michelini, 2009, 2011*), NonLinLoc (e.g., *Lomax et al. 2009*). The first three accept real time waveform data streams as input data. The fourth works on a set of picks from a single event. In addition a procedure has been selected for the automatic high-resolution relative location of earthquake sequences (RTRL, *Rietbrock and Waldhauser, 2004*).

Size and mechanism

Virtual Seismologist, SCARDEC (*Vallee et al., 2010*), Real-Time mBc (*Bormann and Saul, 2009*), Time Domain Moment Tensor – TDMT (*Dreger and Helmberger, 1993*), MW-FMNEAR (*Delouis et al., 2009*) and EARLY-EST are the software codes that provide estimates of earthquake size and mechanism. In addition, the algorithm of *Kikuchi and Kanamori (1991)* has been also analysed as a potential candidate for finite fault determination.

Effects of the earthquake

ShakeMap (*Wald et al., 1999*) is the only candidate code that addresses the effects in terms of ground motion resulting from an earthquake at local/regional scale.

Overall, the survey has shown that the selected softwares require either windowed waveform data in some particular format (e.g., SAC, miniSEED, …), or real-time data streams (e.g., from seedlink) or parametric data (e.g., phase onset times, peak ground motion parameters) as input. Station information, or metadata is also required most softwares. It is evident that each software is designed with some unique formats in terms of both input and output, In terms of output, an additional module is required to re-format the output into a standard such as QuakeML (or other XML extension).

Additionally, it is clear that there exists an overlap between some of the user applications and some services already offered by data centers. Some user applications may be "mature" enough to be implemented also as web-services offered by data centers themselves (and a close link with NA9 of NERA). It follows that the toolbox developments could be also viewed as a tool for potential web-services.

General scheme

The development of a general purpose seismological tool-box (or of a minimal set of tool-boxes) requires a thorough understanding of use cases and software requirements. We note also that the software included in the use cases is quite diverse and in many ways is representative of much software available for seismological analysis. It follows that development of a general tool box is of importance not only for the selected use cases of this work package but, in general, for any seismological analysis software.

A simplified cartoon of the interplay between data, software, and results representation is provided in Figure 1. The cartoon shows the role that the tool-boxes play in interfacing i.) the data (real-time streams and offline time windows) to the applications of the use cases and ii.) the results of the applications to a generic and standard representation availing of the QuakeML and its XML extensions.

In Figure 2, attention is put onto the data interface part. A distinction is made (see "Data Type row") between real-time data obtained directly, for example, from a seedlink server (RT, green) and windowed data obtained very rapidly from a data archive after an earthquake target of the analysis (DataArc, red) has occurred. The box indicated with SeisMon indicates data obtained directly and within seismic monitoring packages like Earthworm (http://www.isti2.com/ew/) and SeisComP3 (http://www.seiscomp3.org). The upper part of the cartoon (i.e., "Existing Services") shows some of the main available software libraries and/or services to obtain either the real-time or the windowed, on-request data.

It is important to note that the software libraries and the services outlined in Figure 2 are on the one hand important tools that can be used by seismologists to retrieve and use data but, on the other, they are also somewhat highly specialized and they may not be straightforward to use. In addition, they are often individual tools not easily cast within a comprehensive framework so that they do not offer a simple environment to be used for interfacing the data to the user applications. To this regard and if we were to make a simple list of what seismologists effectively need to do their analysis, we will find that the "desiderata" are effectively quite straightforward. Although what presented below cannot be considered a complete list of all the seismologists' wishes, we find that many seismologists will agree on the following requirements.

1. Ability to select and rapidly select and download the "best data" from a single virtual archive
   1.1. The data should be chosen according to
      1.1.1. Geographical Target area
      1.1.2. Time window
      1.1.3. SNCL (Station, Network, Channel, Location)
      1.1.4. Quality of the data (i.e., QC)
   1.2. Data filtered at the data center

1.3. Data output in seismological standard formats (eg, SAC, MSEED,...)

1.4. Data exactly windowed and synchronized

1.5. Data corrected for instrument response

2. Real-time interface that provides data synchronised to RT-Apps and possibly QC'd

   2.1. simple recursive filtering applied

   2.2. generic characteristic function processing or simple processing applied

3. The results of the user application should be easily packaged and shipped for

   3.1. web publication

   3.2. "reinsertion" into seismic monitoring systems

   3.3. "plotting" by dedicated utilities

   3.4. "insertion" into DB

These requirements lead directly to the two layers represented by the toolboxes in Figure 1, that is, an *interface layer* between Data and Applications and an interface layer between Applications and Output.

Interface layer between Data and Applications

This interface entails two basic modes of operation. The first consists of requesting time windowed waveform data off-line and it involves selection according to geographical station selection, time window, SNCL (station, network, channel, location) and quality of the data. In addition, all data required by the user applications should become available and ready-to-use regardless of the data center. An issue of importance for some software, is the availability of exactly the same start-end times of the data window and the same number of samples. Data filtering together with the ability to perform the instrumental removal in a standard manner are other important features. Standard seismological formats such as SAC, MiniSEED, GSE, AH, (...) should be also be provided to provide greater flexibility to input the data.

The second mode of operation consists of connecting to real-time streams such as those obtainable from seedlink servers and provide the resulting data packets to be ingested by the user application software. In this case, the toolbox consists of a library of real-time modules callable within a script/procedure.

This interface layer and the resulting toolbox/libraries should be callable using simple command line requests and the script procedure should be easy-to-implement on user's computer.

Interface layer between Applications and Output

This interface layer is entirely dedicated to the conversion of user application results into QuakeML (or other XML extension). In this manner, the results can be promptly "ingurgitated" for web publication, being "reinserted" into the seismic monitoring systems and/or "plotted" by dedicated utilities, and/or "inserted" into existing DBs.

Survey of available data services

Offline requests

There are several ways available to requests data. Data centers like IRIS and ORFEUS, given their mandates, have been offering various services (e.g., Wilber II, BREQ_Fast, ...) for years and special attention has been placed into offering web tools. In the following focus will be centred to services that can be implemented using simple command lines since they can be easily scripted and automated. In addition, special attention is deserved to web-services since they represent the more modern approach for data request while insuring the largest flexibility (e.g., REST requests are effectively URLs) . Recently there has been also much effort to homogenise the services among the different data centres.

Figure 3, provides a summary of the web services offered by the IRIS, EMSC-ORFEUS and INGV data centers for data and metadata access and download. In Figure 4 there are listed the available clients for the web services above.

The developed clients adopt different languages (Python, Perl, Java, C++), with by large number of clients available in Python through the Obspy module library (http://obspy.org/).

Real-Time

Real-time (RT) services are much less common than the off-line, time window request modes above. This owes to the relatively small number of researchers developing software that can take into full account RT data streams. Nevertheless, it is thought that for the reasons outlined in the introduction, an ever increasing number of researchers will attempt the implementation of software in RT.

Real-time data can be obtained, for example, from the connection to *seedlink* servers or *NaqsServer*. An extended summary of the available software for SeedLink is provided on the IRIS dedicated site (http://www.iris.edu/data/dmc-seedlink.htm). (For the NaqServer, it is available the *libnmxp* library - http://mednet.rm.ingv.it/downloads/soft/nmxptool/README). These are all programs and software libraries designed to work outside the seismic monitoring programs such as SC3 and EW. To this regard, an alternative approach to the use of "external" software for the analysis of RT streams consist of performing a basic installation of SC3 (and of EW) and activating only those modules/libraries necessary for RT data import / export.

In summary, at present there is not available a general purpose toolbox with the functionalities listed in the general scheme section and capable to interface existing software with the data in a working environment independent of those adopted by the seismic monitoring data centers. This last aspect is important since it opens to a much enlarged number of researchers simple ways to test their own software in a seismic monitoring alike environment.

Toolbox Design

Software language selection

In general, software intended to be widely used should not only provide features that other softwares do not, but it must be also easy and straightforward to download, install and use, with many examples showing how it works. Once the software starts to be known and several users and/or practitioners in the field appreciate it (i.e., they use it!), the software effectively start its own lifecycle until replaced by others that better suit the requirements above in the coming years. The duration of the lifecycle depends on many factors not least the degree of assistance offered by the developers to the users and the software architecture framework adopted. For example and when the budget for the development is of restricted amounts, it is important to exploit as much as possible open source software and libraries.

Selection of the language framework development of the toolboxes required some thorough appraisal of the modules and libraries already available. Within seismology, and as shown in Fig. 5, the ObsPy Python library has been found quite appealing given the number of data services already developed. In addition and recently, ObsPy appears to have coagulated a quite active developer and user community while Python, besides being a very powerful scripting language, it is also becoming a widespread developer language at many levels in science and engineering. The availability of a well documented online tutorial of the ObsPy modules, of a ticketing system for assistance and of an easy way to install the libraries on all the most common system platforms has made the selection of ObsPy for the development appealing and for this reason it has been proposed for the development of the interface toolbox using Python.

Basic structure

In the "general scheme" section, it has been laid out what are the "desiderata" that the toolbox should warrant. The cartoon in Figure 6 exemplifies graphically some of these basic requirements.

In Figure 7, we present the basic structure for what concerns the off-line data. This diagram refers to the software *WaveDownloader* that has been designed while attempting to map the ideas of the general toolbox sketched out in the previous part of this report.

We note first that the right part of the tree diagram focuses on the data request part. It includes

> *Server selection*. The assumption made here is that the data access and download, from the user perspective, should be totally transparent to the actual data center effectively providing the data. The procedure (script) has been designed to incorporate those calls required to access the data at the main data centers such as EIDA (European Integrated Data Archive), IRIS and/or others depending on the services availability.
>
> *Time selection*. The data time windows should be selected both using start and end times or the start time and window length.
>
> *Geographical Location*. Selection of the stations using either a bounding box or a circular/annular selection.
>
> *Data type (SNCL)*. Selection according to station name, network, channel and location.
>
> *Quality*. This allows to reject stations channels with gaps or other indicators of poor quality.

On the left part of the diagram of Fig. 7, there are reported some of the basic processing tools included in the prototypal toolbox.

> *Wave processing*. This includes some basic operations on the data such as remove mean and trend, deconvolution of the instrument response, rotation of the horizontal component and decimation. It also includes analysis such as filtering and extraction of characteristic features of the waveforms such as central frequency, calculation of characteristic functions and extraction of peak ground motion parameters.
>
> *Output*. The output - which becomes the input to the user applications, eventually - consists primarily of waveforms in various formats or of parametric data formatted opportunely according to QuakeML or other standard formats (e.g., the Shakemap peak ground motion parameters in XML format).
>
> *Plotting*. Various ways to plot the downloaded data (e.g., according to component, type of channel, radially/transverse rotated components, azimuth and scaled by distance).

The developed toolbox denominated WaveDownloader is still in a prototypal version and it will be distributed for the first time to the WP partners for testing by the end of 2011.

Real time

This part of the toolbox concerns the feeding of the real-time stream to software designed for real-time processing. The objective is the development of an ObsPy real-time module capable to ingest the seedlink stream and have the resulting packets as input to existing software (e.g., WaveLoc) that accepts real-time streams. To this end, it is now under development an object oriented "architecture" for adding real-time processing within ObsPy. Thus and more technically, a new ObsPy class, `RtTrace`, and package, `obspy.realtime`, will make it possible to the user applications to receive streamed packets and process them according to his/her needs. In practice, this boils down to the implementation in Python of the C library libslink. To this end, Anthony Lomax has been sub-contracted given his experience with implementations in C and Java of similar libraries.

Results to standard QuakeML (or XML) conversion

This part of the toolbox definition is currently the least standardised (and hence so far not developed) because it involves the conversion between individually defined, user application formats and the standard QuakeML format - when possible - or an approriate XML extension. This toolbox is being (and will continue to be) developed on a case by case basis although it is thought that modifications of the user application output into, for example, the comma separated value (csv) will make the conversion to QuakeML straightforward.

Conclusions

This document defines the toolboxes required to interface, implement and test the use case seismic monitoring softwares. The toolbox should support the testing existing software in (near)realtime without the need of installing more complex seismic monitoring systems and embedding the software within. This approach lets the user test first if the application is fit for being engineered within available monitoring systems such as Earthworm or SeisCompP3.

The Python script language has been selected because of its flexibility and ease of use, and because it is available as the ObsPy module library which contains already many of the functionalities required by the toolbox.

Two toolboxes are now under development. The first seeks integration of off-line data obtained from data archives and of realtime data streams obtained through seedlink. To this regard, a time domain python implementation of the C library libslink for the realtime analysis is being developed. The second toolbox prototype will provide an interface between the results of the user applications and more standard QuakeML (or XML extensions).

References

Bormann, P., and J. Saul (2009), A Fast, Non-saturating Magnitude Estimator for Great Earthquakes, Seismological Research Letters, 80(5), 808–816.

Cua, G., and T. Heaton (2007), The Virtual Seismologist (VS) method: A Bayesian approach to earthquake early warning, Earthquake Early Warning Systems.

Dreger, D. S., and D. V. Helmberger (1993), Determination of Source Parameters at Regional Distances With Three-Component Sparse Network Data, J. Geophys. Res, 98, 8107?8125.

Delouis, B., J. Charlety, and M. Vallee (2009), A Method for Rapid Determination of Moment Magnitude Mw for Moderate to Large Earthquakes from the Near-Field

Spectra of Strong-Motion Records (MWSYNTH), BULLETIN OF THE SEISMOLOGICAL SOCIETY OF AMERICA, 99(3), 1827–1840.

Kikuchi, M., and H. Kanamori (1991), Inversion of complex body waves--III, BULLETIN OF THE SEISMOLOGICAL SOCIETY OF AMERICA, 81(6), 2335–2350.

Lomax, A., A. Michelini, and A. Curtis (2009), Earthquake Location, Direct, Global-Search Methods, in Encyclopedia of Complexity and Systems Science, pp. 2449–2473.

Lomax, A. J., and A. Michelini (2009), Tsunami early warning using earthquake rupture duration, Geophys. Res. Lett, 36, –.

Lomax, A., and A. Michelini (2011), Tsunami early warning using earthquake rupture duration and P-wave dominant period: the importance of length and depth of faulting, Geophys. J. Int, 1–9, doi:10.1111/j.1365-246X.2010.04916.x.

Maggi, A., and A. Michelini (2009), Rapid Waveform Earthquake Location in Italy, in EGU General Assembly 2009, vol. 11.

Waldhauser, F. (2004), A narrowly spaced double-seismic zone in the subducting Nazca plate, Geophys. Res. Lett..

Vallée, M., J. Charléty, A. Ferreira, B. Delouis, and J. Vergoz (2010), SCARDEC: a new technique for the rapid determination of seismic moment magnitude, focal mechanism and source time functions for large earthquakes using body☐ …, Geophysical …, 184, 338–358, doi:10.1111/j.1365-246X.2010.04836.x.

Wald, D. J., V. Quitoriano, T. H. Heaton, H. Kanamori, C. W. Scrivner, and C. B. Worden (1999), TriNet ``ShakeMaps‟”: Rapid Generation of Peak Ground Motion and Intensity Maps for Earthquakes in Southern California, Earthquake Spectra, Earthquake Spectra, 537.

Tables

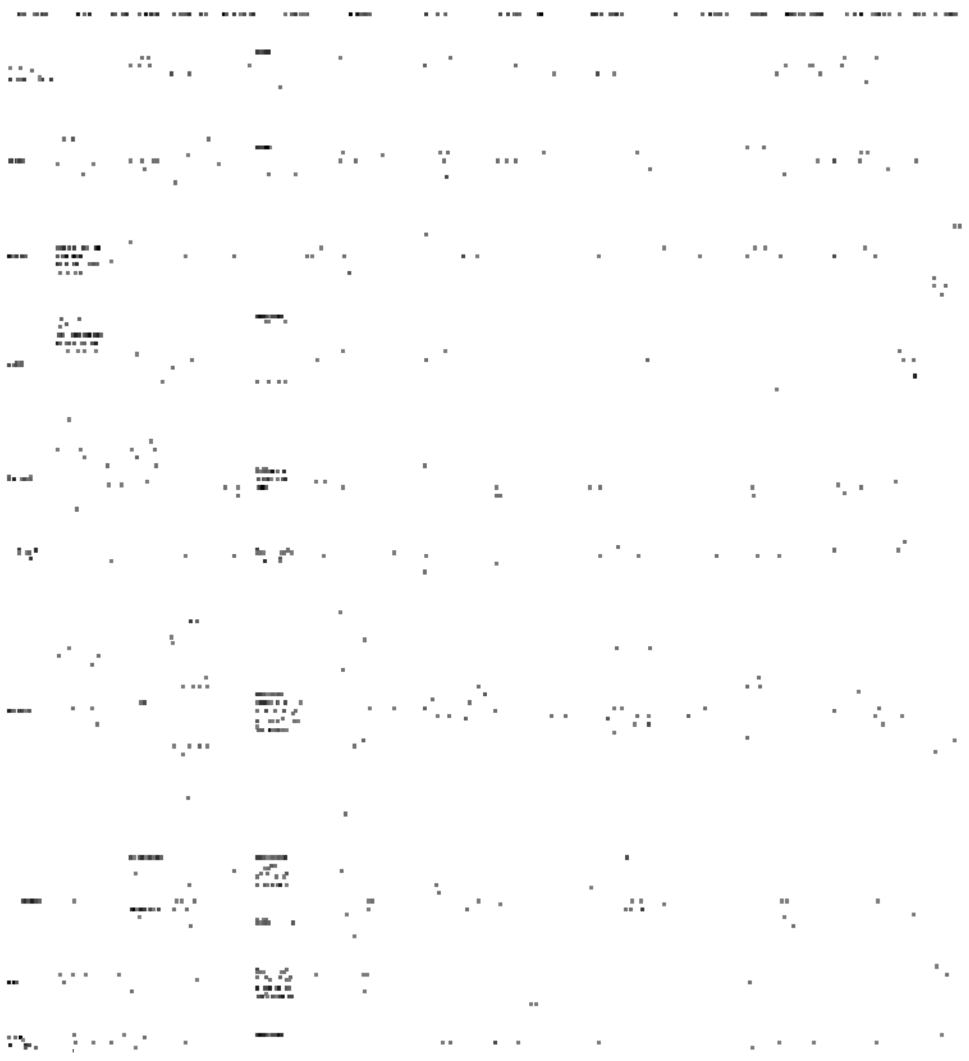

[illegible]

Table 1. Summary table of the software profiled for the WP12 tool boxes. *(The SurveySummary.xls file is attached to this report in electronic form).*

Figures

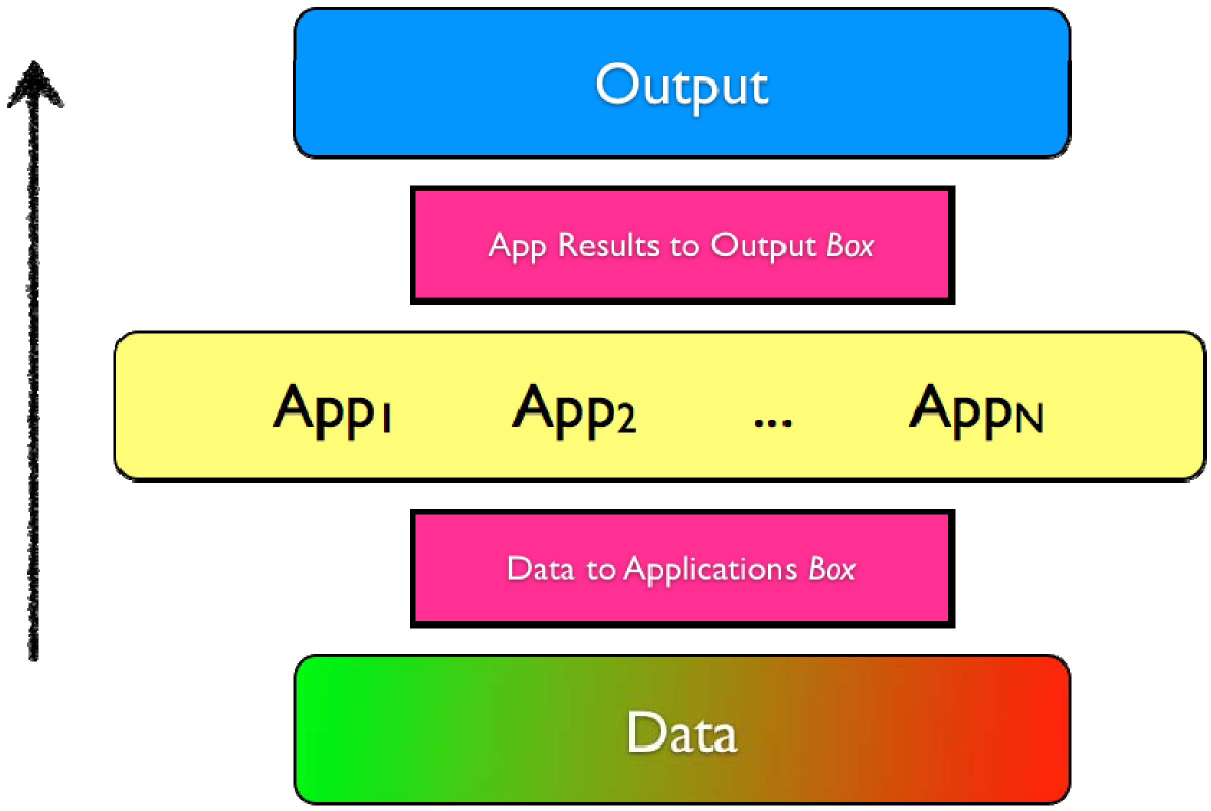


Figure 1. Simplified cartoon showing the role of the tool-boxes. App$_i$ (i=1,2, ...,N) indicate the use case applications. The flow goes from bottom (data) to top (results). A generic tool-box is required to interface the data to the applications. A second tool-box is needed to interface and convert the results of the applications into standard outputs like QuakeML or other XML extensions. The data box at the bottom turns from green to red colours to indicate the two end-members of data required - real-time streams and off-line data windows.

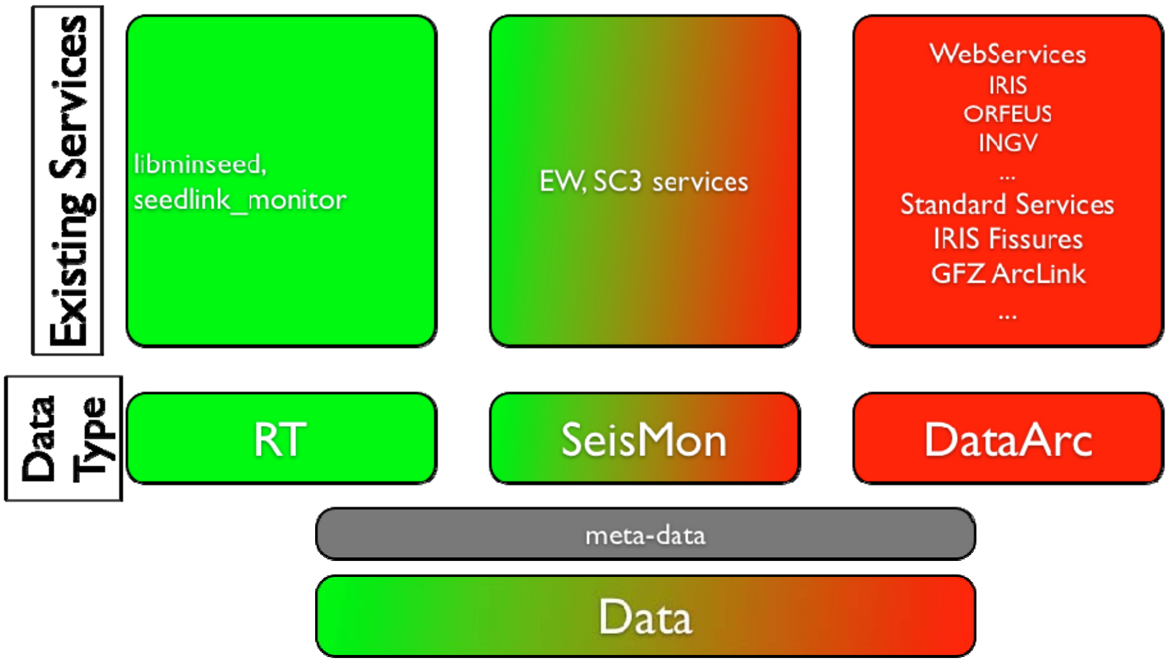


Figure 2. Simplified cartoon summarising the type of data available and the existing software and libraries available for the services. RT Real-Time; SeisMon, seismic monitoring packages like Earthworm (EW) and SeisComP3 (SC3); DataArc, data from archives data. The upper part of the cartoon indicates the existing services and it lists for each of the data types the available software libraries and/or software services and the institution providing them. The purpose is to show the relation and it is not necessarily inclusive of, for example, all the web services available.

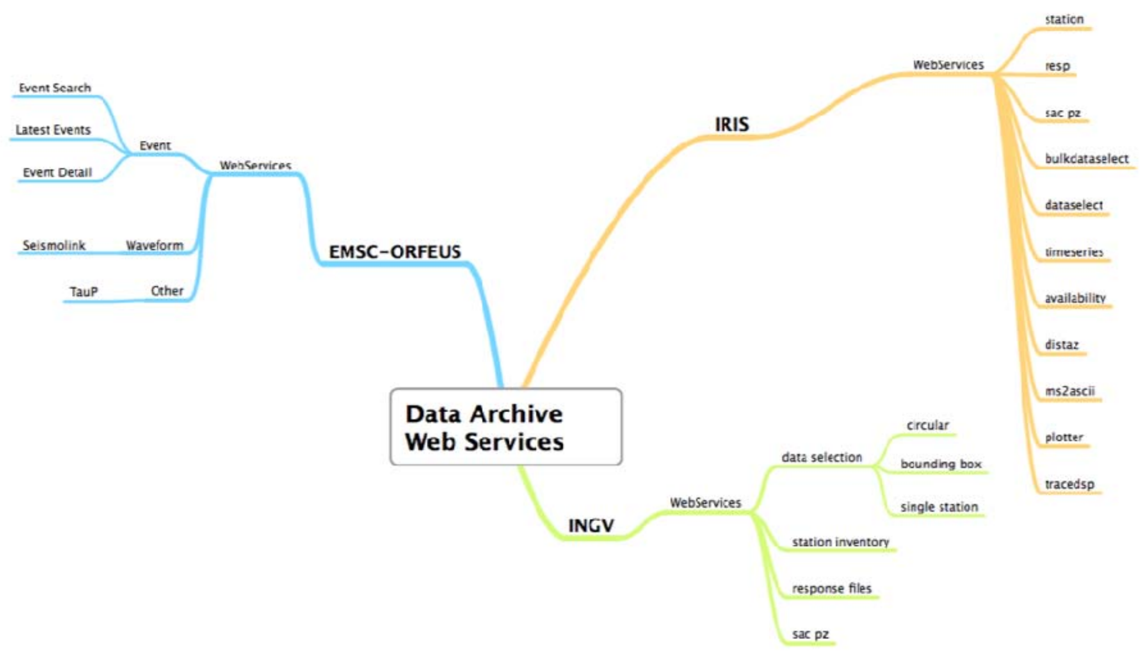
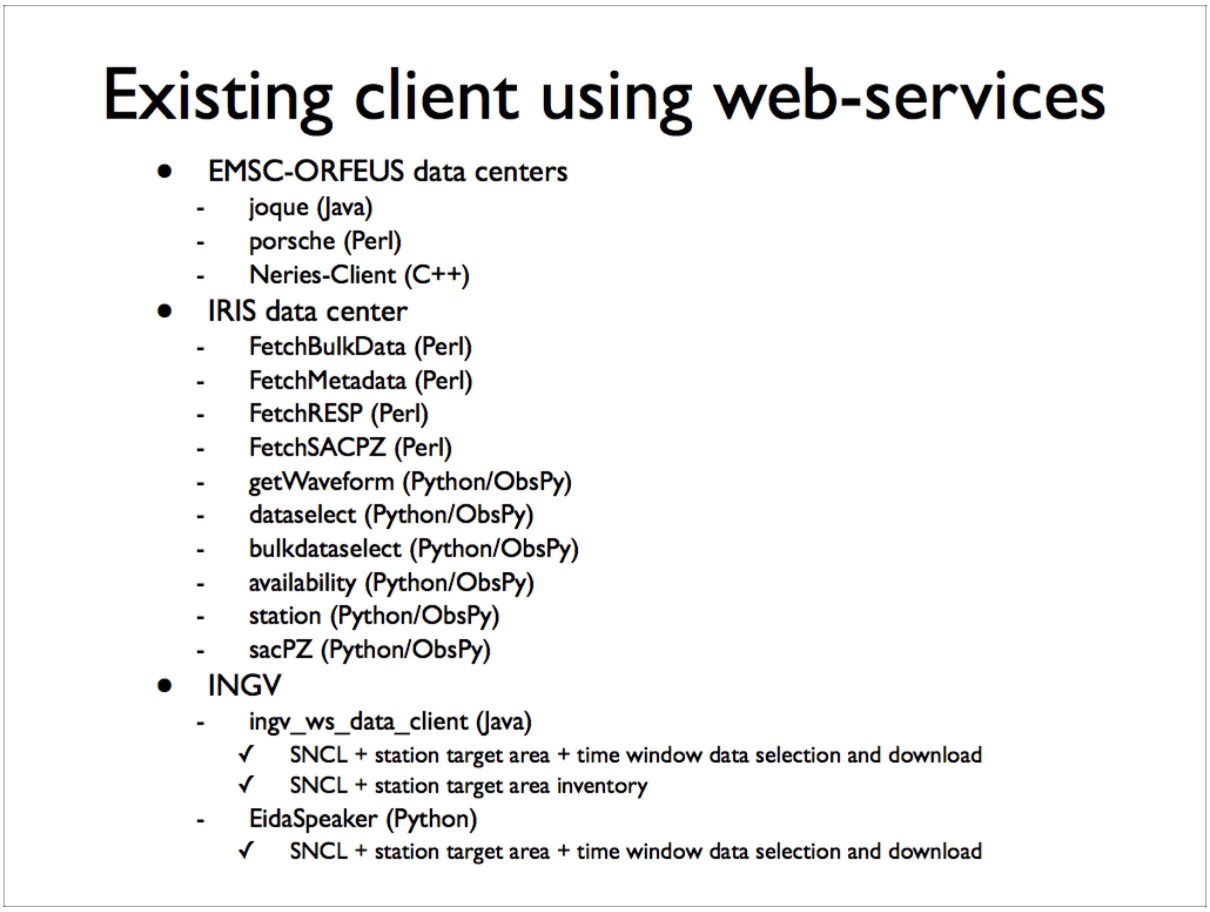


Figure 3. Diagram exemplifying the web services offered for data and metadata access by the IRIS, EMSC-ORFEUS and INGV data centers.



Existing client using web-services

- EMSC-ORFEUS data centers
  - joque (Java)
  - porsche (Perl)
  - Neries-Client (C++)
- IRIS data center
  - FetchBulkData (Perl)
  - FetchMetadata (Perl)
  - FetchRESP (Perl)
  - FetchSACPZ (Perl)
  - getWaveform (Python/ObsPy)
  - dataselect (Python/ObsPy)
  - bulkdataselect (Python/ObsPy)
  - availability (Python/ObsPy)
  - station (Python/ObsPy)
  - sacPZ (Python/ObsPy)
- INGV
  - ingv_ws_data_client (Java)
    - ✓ SNCL + station target area + time window data selection and download
    - ✓ SNCL + station target area inventory
  - EidaSpeaker (Python)
    - ✓ SNCL + station target area + time window data selection and download

Figure 4. Summary of the existing clients an of the language adopted to the web services listed in Figure 3.

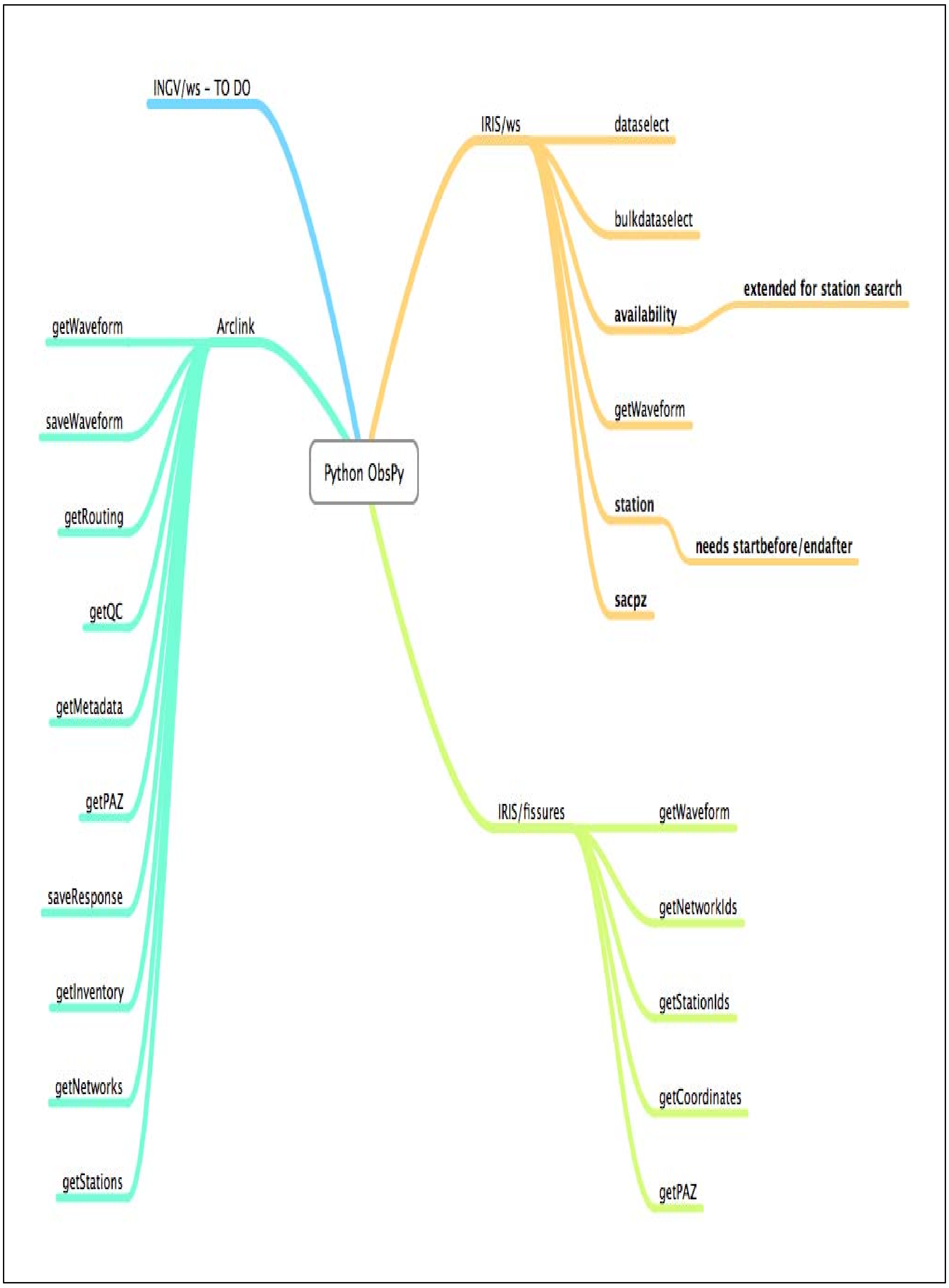


Figure 5. Cartoon summarising the services offered by the ObsPy module library. Note that in addition the IRIS web services, it offers also modules to interface with ArcLink. The INGV web services - developed during the project - are effectively ready for being inserted into ObsPy as module granting access to the whole federated EIDA (European Integrated Data Archive).

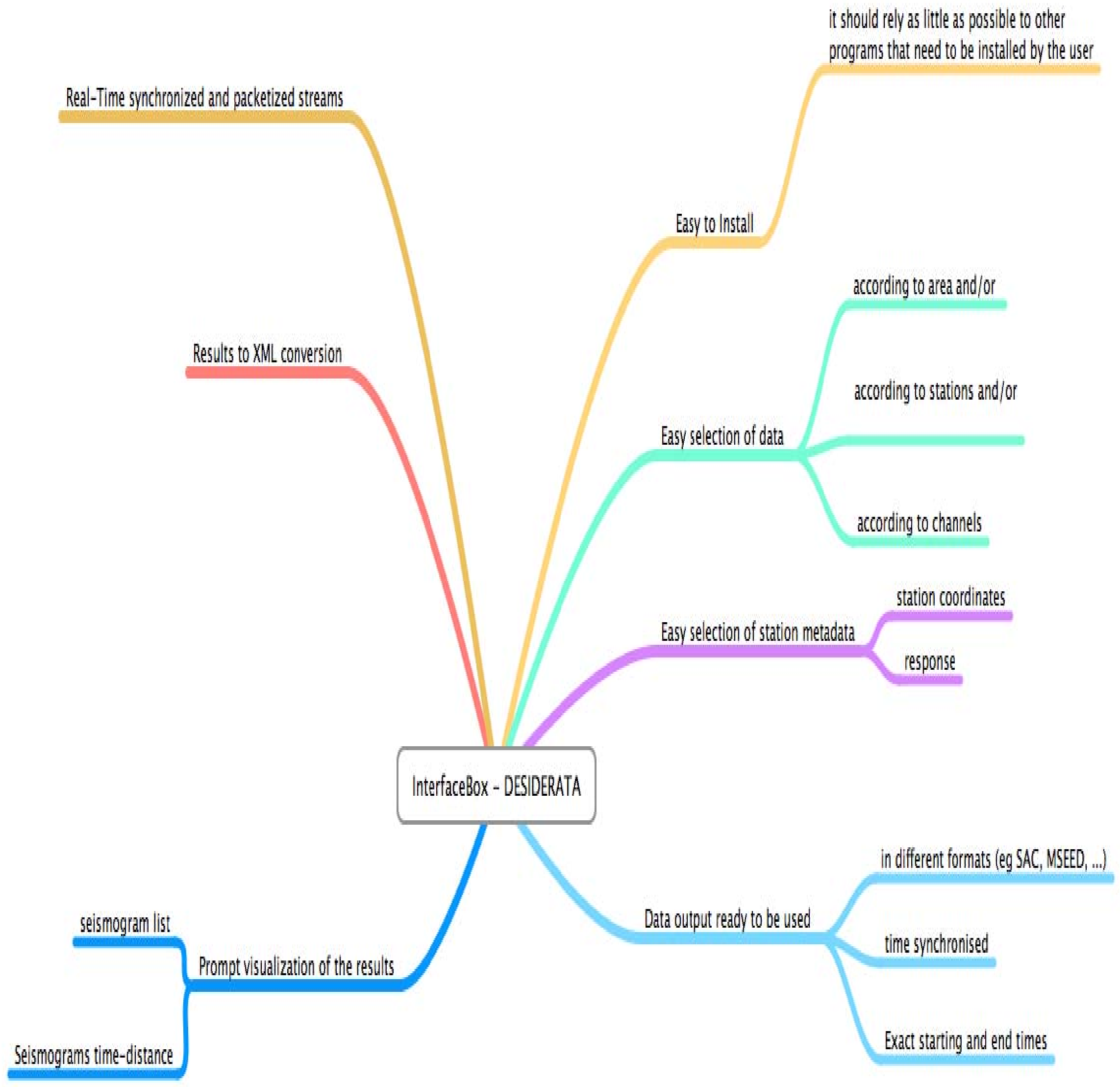

Real-Time synchronized and packetized streams
it should rely as little as possible to other programs that need to be installed by the user
Easy to Install
according to area and/or
Results to XML conversion
according to stations and/or
Easy selection of data
according to channels
station coordinates
Easy selection of station metadata
response
InterfaceBox - DESIDERATA
in different formats (eg SAC, MSEED, ...)
seismogram list
Data output ready to be used
Prompt visualization of the results
time synchronised
Seismograms time-distance
Exact starting and end times

Figure 6. Simplified cartoon illustrating some basic wishes of the seismological community for what concerns data access and basic analysis.

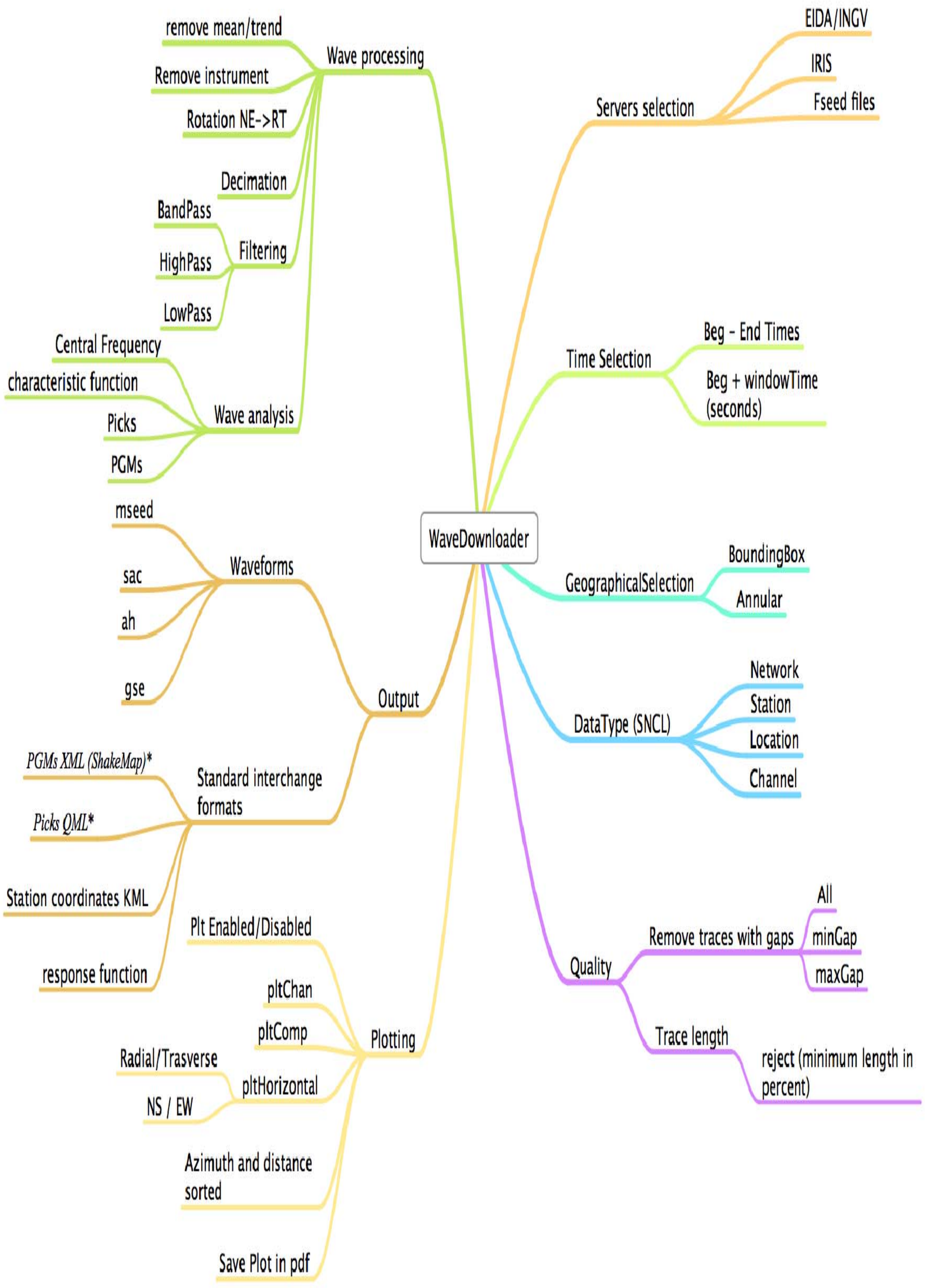


Figure 7. Cartoon diagram showing the main functionalities of the prototypal toolbox - WaveDownloader - for data to software interface for the offline data retrieval.

1. Application Name	2. Objective	3. Language	4. Pipeline	5. Sequential / Parallel	6. Application mode	7. Data input		8. Metadata		9. Data Output			10. Libraries	11. DB	12. HW requirements			13. OS	14. Interfacing	15. Graphics	16. External software		17. Licen
						7.i Waveform	*7.ii Parametric*	*8.a Type*	*8.b Format*	*9.i time/freq series*	*9.ii Parametric*	*9.ii Logging*			*12.a Processor type*	*12.b Memory*	*12.c Typical CPU times*				*16.a Program*	*16.b Libraries*	
VS-EEW (Virtual Seismologist **Earthquake Early Warning**)	Event Location, magnitude, and peak ground motion	C/C++	Client/server application with UNIX message queue between client and server	Multi-threaded	Continuous broadband and strong motion data	**STREAM:** per station real-time waveform data (E,N,Z) components as raw sensor counts. VS currently accepts ew format;	NO	Station DB:station name, component, latitude, longitude, elevation, sampling rate, gain			Real-time XML reports with origin time, magnitude, location estimates at each second	picks times; P/S discriminant; Quake Filter o/p	Typical UNIX libraries (pcre, apr, aprutil, activemq, qlib2, otl, odbc, expat)	YES	SPARC, x86	512 MB	40% CPU	Solaris, Linux	connections to Earthworm waveform buffer and local metadata dB. Also in ETH connected to proprietary system.	NO	Earthworm binder module	CISN waveform processing library (open source)	amoeba simplex algorith Numeric Recipes
WaveLoc	Earthquake location using continuous data streams. This is achieved through back-projection and the adoption of simple reference waveforms.	Python	The software relies on a main python script and a number of general and "ad hoc" python classes	The code is sequential but it has been ported to a cluster in the job array mode and it consists an "embarrassingly parallel application"	continuous data streams on off-line sac and miniseed data. It should be ported to the analysis of real-time streams	**STREAM:** continuous SAC / miniseed waveforms arranged in 35' overlapping time windows	NO	geographical station location, Earth model for arrival time calculation	ascii NLLoc format for station data, currently layered Earth model, soon NLLoc 3D model format and travel-time grids	time domain maximum value of the backprojection procedure	Event Detection (OT, Lat, lon) + other logging parameters, currently uses in house format, can be converted to NLL or QuakeML		PROJ.4 as embedded into the matplotlib python module (basemap)	NO. In the future it may be useful to save the results found	x86	~2GB	In sequential mode. 5-10 minutes for 35 minutes of continuous data depending on number of stations. In parallel mode using job array on a 512 CPU, ~40' for 25 days of data.	Linux and MacOSX	NO	Adopts Python matplotlib mudule and unix convert	NO, though it may soon require interfacing with parts of NLLoc	Numpy, Scipy, matplotlib and ObsPy	NO
NonLinLoc	set of programs for velocity model construction, travel-time calculation and **probabilistic, non-linear, global-search earthquake location in 3D structures,** and for visualisation of 3D volume data and location results.	C, sh, GMT script	NLLoc, self standing command line application (see s/w survey for detail)	Sequential	Event driven		post-event detection values (i.e., acquired from file, DB or web portal)	geographical location, station corrections, travel-time and pick uncertainties	mostly ASCII		phase onset picks, earthquake location, magnitude, ….; Format: NLL, ASCIII, some QuakeML field support	NLL format fields		NO but may be useful in future devs	x86	standard PC	Subsecond to minutes on PC	Solaris, Linux, MacOSX	YES: EW and SeisComP3	YES.GMT, Interactive with and Java Seismicity Viewer	TauPToolkit for teleseismic, global mode location	NO	NO
SCARDEC	Simultaneous determination of **focal mechanism, depth, moment magnitude, and apparent source time functions**, for any earthquake with Mw> ~6.3. This information is retrieved from the deconvolution of several teleseismic body waves (P,PcP,PP, SH,ScS). See Vallée et al. (GJI, 184, 338-358, 2011) for more information.	Fortran. No build tools required	Fortran codes for synthetic simulation of waves, deconvolution, and optimization. All is glued together by bash.	Both are implemented. A cluster is very useful because it speeds up a lot the analysis (30-40min sequential, a few minutes on a 12-core machine)	event driven	**Data windows**. NOTE: Now, waveforms are retrieved by semi-automatic interaction with Wilber II (IRIS) : seed file of teleseismic waveforms is retrieved, then converted into SAC files, which have each the epicentral location and origin time in their header. This SAC files are therefore the unique data input.	Epicentral location + origin time	Instrument response, geographical location and channel orientations	RESP and from SAC header values	Source time functions in ASCII format	Strike, dip, rake, depth, seismic moment in ASCII format		Proj libraries	NO. In the future it may be useful to save the results found	x86	standard PC	30-40 minutes in sequential mode. Few minutes on a 12 core machine	Linux	NO	MatLab	YES, TauP, rdseed, SAC, MATLAB, lhsac (from the "sismoutil" package)	NO	MATLAB
Real-time mBc	Compute mBc in real-time, progressively as more data come in. Potentially long rupture durations require equally long time windows of P-wave energy. Optimally fast response times require measurement of rupture duration in parallel to the magnitude itself	C++, SeisComP3 libraries will provide input/output of waveform/parametric data	Essentially two modules talking to the rest of the world through SC3 messaging. 1) the magnitude computation itself; 2) the determination of rupture duration required by the magnitude computation	Sequential within a module	Actually both: listen for events and react immediately, which involves collecting and processing data in real time from the input data stream	**STREAM and Data windows (format mseed)**	Yes, SC3 messages, access through DB	Location & Instrument response	SC3 DB	Possibly for debugging (mseed and/ot sac)	Magnitude, amplitude, rupture locatiion; SC3 messages	SC3	Yes, requires SC3 and therefore comes with the same dependencies	Yes	x86, single CPU	Moderate, at worst 1 MB per stream	Seconds for many streams	Linux, Solaris	YES, SC3	GUI to display the high-frequency duration function is planned	SC3	NO	all, Seiscom Public License, http://w seiscom rg/wiki/ se
Time Domaain Moment Tensor - TDMT	Compute Moment Tensor	C and shell scripts	Low pass filtered windows are inverted for moment tensor given a suite of GFs	Sequential	Event driven	**Data windows extracted using ArcLink**	Epicentral location and OT + local magnitude	Instrument response, geographical location and channel orientations	from INGV DB	NO (possibly for debugging)	ASCII, falut mechanism parameters, moment tensor, scalar Moment, Magnitude	output strings of the program	Yes, requires SC3 and therefore comes with the same dependencies	Yes	x86, single CPU	small to moderate (<< 1GB)	depending on number of stations from fraction of second to a few seconds	Linux, MacOSX, FreeBSD	YES, INGV seismic center DB	old routines developed by Don Helberger	SAC	Some Numerical receipes routines	NR are licensed
MW-FMNEAR	Automatic determination of the moment magnitude Mw and of the focal mechanism using near source records. Focal mechanism obtained with an extended linear source for M > 5.5, allowing the possibility to obtain a first estimate of rupture length and directivity. The codes are activated once an event has been detected and located. Input data are strong motion and/or broadband records. (Deliverable D12.2)	FORTRAN	Combination of FORTRAN and script (Linux) command shells (sh). Call to SAC commands in scripts.	The code is **partly sequential** but in the present version parts of the code uses up to 7 CPUs simultaneously. So it is optimally coded for running on a 8 cores single machine. *It could be easily optimized for a 24 cores machine and some of the remaining sequential operations may be re-programmed for multi-core execution. So, in the Frame of the NERA project part of the work will be dedicated to improve the multi-core optimization* . Big cluster not really necessary, a single machine with 12 to 24 cores is expected to be the ideal situation.	Event driven	**Data windows, presently SAC, but could be other if we decide to promote a uniform data format among us**	hypocenter coordinates (lat, long, depth, OT) and initial magnitude	station coordinates (lat, long, elevation); sampling rate; component orientations; units of amplitudes on the seismograms , or conversion factor (e.g. sensitivity for raw data in counts); - ideally one SAC file per component, for of each station. But it can be in another format, if it can be easily converted to SAC (e.g. ascii);	Presently read from the header of SAC waveform files, but could be easily extracted from ascii or other types of files if we decide to promote a uniform data format among us. This is a prerequisite; Meta-data (SAC headers) are supposed to be automaticall y generated by the seismic network information system in case an event is	NO (possibly for debugging)	(Graphical plot of waveform fit in postscript or jpeg format)	Results in ascii format: Mw, strike, dip, rake, number of component used, RMS misfit function, index of confidence of the solution...	NO	No, provided the Meta-data listed in 8. are already correctly extracted and written in the headers of the waveform SAC files.	No specific requirement	Standard memory (a few GB is OK)	Depends on the number of stations, distance to the stations, complexity of the 1D-layered model, magnitude. Examples on a 8 CPUs linux machine for Mw and focal mechanism with a linear finite source model completed in: • 6 minutes for l'Aquila (Mw 6.3) earthquake using 11 stations at distance < 200 km; • 11 minutes for the Tocopilla (Chile, Mw 7.7) using 5 stations at distance < 200 km	Linux	Automated determinations in test in Taiwan and Nice	Very raw (not elaborated) graphics at present day (waveform fit and focal mechanism in postscript or jpeg)	SAC	SAC	NO
EARLY-EST	EArthquake Rapid Location System with Estimation of Tsunamigenesis	C, sh, Python, GMT script	**miniseed_process**: time-domain processing on Mini-SEED event files, **seedlink_monitor**: time-domain processing on continuous data returned from a SeedLink servers	The main processing code is sequential but the graphics/reporting at 1 min intervals is done in a separate process. It may be possible to parallelize the association/location algorithm into 8 simultaneous calculations.	event driven continuous broadband	**Data windows in miniseed format extracted using ArcLink or from Wilber II** **STREAM:**from seedlink server	NO	station coordinates and sensitivities obtained from files or from webservice requests (e.g. iris/ws, ...); ak135 spherically layered earth model for travel-times and take-off	ascii format for station data, travel-times and take-off angles integrated into C header files		XML reports, same for both programs, EXCEPT OUTPUT EVERY MINUTE FOR seedlink_monitor	to stdout and stderr, variable verbosity levels	libslink, libmseed, libxml2, GMT		any processor supported by gcc	depending on the number of stations input data	for a single event of around M5-6 (20-100 associated picks) takes about 5sec and the reporting module about 5-10sec collecting data packets from the SeedLink servers for about 200 stations and processing the data in the trace-processing module takes very little CPU time, 2-3% CPU on average.	linux, MacOSX	Installed ay INGV with data from IRIS, INGV, GeoScope and GFZ seedlink servers	GMT	iaspei_tau (only if need to use other than ak135 model), HASH , Python, GMT		GN
RTRL	Rapid relative location for earthqukes using a well located event data base	C&Fortran and shell scipts	time domain processing of miniseed files, libmseed, seedlink	Main processing is sequential, but parallel streams for each station is envisaged	event driven now, future on continuous waveforms for event detection	**Data windows either from file or through arclink server; cont. waveforms seedlink.**	Evnt location from single event location + picks...	station coordinates and calibration info	whatever, ASCII and SAC		parametric	to srdout and log files	none	no, but building right now infrastructure for waveform database	-	PC, but needs large memory and fast disks for IO	scalable	Linux	No	GMT	none	none	GNU
Finite Fault Inversion (Kikuchi & Kanamori)	Set 1: Inversion allowing mechanism changes. Set 2: Fine tuning of the source time function and slip distribution. Set 3: Inversion with a fixed fault mechanism.	Fortran (main code), perl, sac macros	Combination of FORTRAN and perl scripts. Call to SAC macros in scripts.	Sequential	Event Driven	**Data windows:** sac binary file format (from fseed format)	Lat Lon, depth and event time. Earthmodel (for Green's functions), frequencies of bandpass filter	Instrment Response and station coordinates (from fseed archive)	ASCII	Green's functions	Source parameters		Numerical recipes	NO	x86, SPARC (?)	Standard PC	Subsecond to minutes on PC	Linux, MacOSX, Solaris(?)	Command line on terminal. Can be easely automatized.	Postscript files of outputfiles	SAC, perl	numerical recipes	numeric recipes (eigsrt, jacobi, lubksb, ludcmp, quad3d)