PROCESSING AND PUBLISHING THE ACTIVE DEFORMATION AREAS OF ALL EUROPE – CONCEPT AND FIRST IMPLEMENTATION STEPS FOR A LOW-COST SOLUTION

José A. Navarro[a,*], Danielly García[a], Michele Crosetto[a], Oriol Monserrat[a]

[a]Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Av. Carl Friedrich Gauss 7, 08860 Castelldefels, Spain — (jose.navarro, danielly.garcia, michele.crosetto, oriol.monserrat)@cttc.cat

KEY WORDS: WebGIS, WMS, WFS, Active Deformation Areas, Land subsidence, Open-source, Low-cost, Geoprocessing.

ABSTRACT:

The European Ground Motion Service provides a plethora of information about land subsidence in the form of Persistent Scatterers (PS). Data for all over Europe will be soon available, facilitating the analysis of ground deformation at a continental level. However, Active Deformation Areas (ADA) may be used instead of PS, since those are more understandable for humans. The CTTC already developed a tool, namely *ADAfinder*, to produce the said ADAs out of input PS. The authors are involved in an ongoing project, at its very early stages, targeted at computing the Active Deformation Areas for all Europe using the PS data provided by the EGMS, making the results available to the community by means of a WebGIS. A few concepts are already very clear, such as the high volume of information to process, the need of a specific software chain to do the actual processing and the economic cost that storing and making the results accessible might imply. This paper describes how the data volume and economic factors led to the selection of an in-house hosted & developed solution rather than resorting to well-established and expensive providers of cloud-based solutions. Furthermore, the set of self-developed tools required to implement the production chain are also explained. Also, the open-source tool used to prepare a server able to store the data and make it available to the public are described here. Very preliminary but promising figures concerning processing and response times have been included too.

1. INTRODUCTION

The European Ground Motion Service or EGMS (EEA, 2021) will very soon start delivering billions of Point Scatterers (PS) covering all Europe (Crosetto et al., 2020). Such abundance of information will become a very important tool with respect to the analysis of ground deformation at a continental level. However, PS are not, perhaps, the best tool to analyze such deformation processes due to, precisely, the huge amount of information at hand, especially taking into account that each of these PS includes a series of deformation values whose length may exceed easily one or two hundreds of values. Moreover, not all the PS included in these data sets will point to areas where actual deformation processes are taking place.

In (Barra et al., 2017) a methodology to identify the so-called Active Deformation Areas (ADA) was described. The ADA represent areas where ground deformation processes have been identified with some degree of certainty. In fact, this methodology not only provides the ADA as a polygon delimiting the area where the deformation process is taking place, but also includes an assessment of the certainty of such process in a discrete four-level variable where 1 means "very reliable", 2 means "reliable" but an analysis of the time series (TS) is suggested, 3 means "not so reliable", i.e., a deeper analysis of the TS is necessary, and 4 means "not reliable". The advantage of using ADA instead of the original PS is twofold: firstly, the amount of information to analyze is drastically reduced, making the problem much more manageable; secondly, the quality assessment of the findings helps the experts to concentrate their efforts in those places that do require their knowledge. Figure 1 depicts this situation. There, the PS (Figure 1a), have been colored according to their mean yearly velocity (see the legend); the ADA (Figure 1b) have been colored according to the aforesaid reliability criterion (green, blue, yellow and red, from most to less reliable results). A tool implementing this methodology, *ADAfinder*, was developed at the CTTC back in 2018 and has been continuously used since then (Tomás et al., 2019, Navarro et al., 2020, López-Vinielles et al., 2021) and it is freely available (contact the authors for more information).

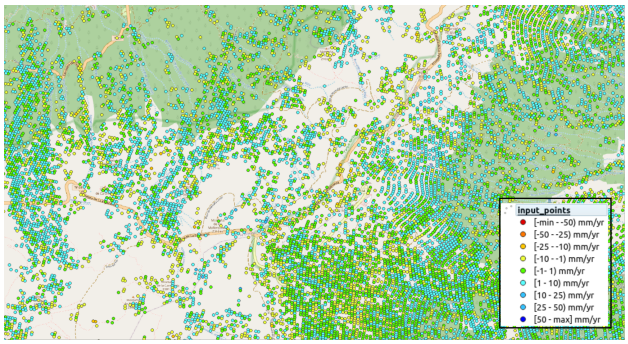
In the context of a project SARAI, funded by the Spanish Ministry of Science and Innovation (*Ministerio de Ciencia e Innovación*, MCIN), the authors have been involved in the task of identifying and publishing on the Internet the ADA covering all Europe. More precisely, such ADA must be made available by means of, at least, a basic WebGIS. Although this project and the task motivating this paper are just starting and far to be finished, some relevant steps have already been taken in order to guarantee that such task is accomplished. This paper explains these preliminary steps and the initial conclusions drawn.

2. THE FIRST APPROACH: THE CLOUD

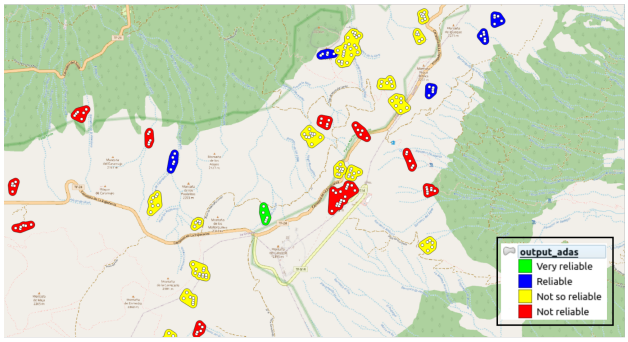
When first thinking about how to solve the problem of making a substantial number of ADA available on the Internet, the first solution that was considered was to use the cloud to do it.

Although it is not possible to ascertain *a priori* what will be the final number of ADA to store and display, it seemed clear that the huge amount of PS would lead to maybe tens of thousands of them. Furthermore, ADA include a noticeable amount of data; they consist of a polygon defining the moving area plus a series of attributes. The polygon itself may be complex, consisting of up to hundreds of points; among its attributes, it is included an averaged version of all the deformation time series of the PS inside the ADA, which, easily, may consist of two of three hundred values.

* Corresponding author



(a)



(b)

Figure 1. The number of PS covering an area (a) may be very big. The ADA (b) are more manageable and convey much more useful information.

Besides the data volume problem, other considerations come to mind, such as the need to back up all this information, the need of nearly 100% server's uptime or the bandwidth required to handle a minimum number of simultaneous users, seemed to made a cloud-based solution almost ideal. Consequently, well-established, well-known cloud providers were contacted in order to obtain quotations stating the price of the service.

The prices given by these cloud providers, we may say, is the reason why this paper exists. To keep things simple, suffice it to say that hosting between 50–100 Gb of data would cost about 24.000€ per year plus others yearly costs amounting close to 7.000€. These figures, unfortunately, far exceeded the limits established by the project's budget. The natural consequence of this situation was to start thinking about building an in-house system hosting not only the data but also the software required to perform the task.

Although this approach is not exactly cheap, it may be said that, *comparatively speaking, such a system is a low-cost one*. A full-fledged, modern server including a RAID disk system to guarantee data integrity may be in the range of 4.000€ to 6.000€, which amounts to the cost of just about two or three months of data storage in the cloud; furthermore, the software stack required to set up a server including (basically) a database, web and geospatial data servers may be easily found in the open-source gratis market, thus adding no extra costs to the solution. This is detailed in section 3. There are, however, two extra tools to develop (the WebGIS and the software publishing the ADA created by ADAfinder) but these should be developed too in the case that the cloud would had been selected, so, from the economical standpoint, they imply no difference.

Of course, this must be seen in the context that the CTTC already operates its own data center and that the personnel to manage, maintain and backup the server is already available, so there is no hidden extra costs related to these activities.

3. THE SERVER AND THE SOFTWARE STACK

It is convenient to start saying that, at the moment of writing this paper, the actual server to host the data and the software stack has not been yet acquired. The only steps taken in this direction have been to ascertain how much a computer able to handle the problem would cost. Since no actual server is available yet, the proposed architecture of the system has been tested in regular development computers. Obviously, such a setup will not be the most appropriate to check the performance of the system, but it will make possible to validate the concept itself. This is a consequence of the fact that the project is still in its very early stages.

When talking about the "software stack", two different interpretation of these words arise. The first refer to the set of software tools that will be used to set up the server—which will be described in section 3.1; the second concerns the applications that will be operated routinely to produce the data that will feed the server, that is, the ADA covering all Europe. These tools will be detailed in section 3.2. Finally, the WebGIS should be included in the first case (server's software stack), for it sits on top of all the other tools required to make the server operative, but since it is an in-house development, it will be discussed separately in section 3.3.

3.1 The ADA Server

The need to keep costs within the limits set by the project budget led to the decision to select, whenever possible, open-source, low-cost or even gratis tools instead of proprietary ones. Furthermore, the set of tools had to cooperate with each other, thus making what could be called an ecosystem able to provide the sought service. Finally, the tools selected had to be present in, at least, the two more common operating systems, that is, Microsoft's Windows and Linux to be able to select the most convenient (or even cheaper) one in the future. With all these conditions in mind, the several components making the server software stack where selected.

The main concern that appeared when the cloud-based solution was rejected because of economical reasons was how to store the information related to ADA. The Active Deformation Areas are, essentially geospatial data, that is, maybe the most characteristic of their attributes is that these refer to a geographic location. Of course, other attributes as the quality assessment are very important for the final user, but, from the storage standpoint, the geospatial component was decisive to select the software tool to store ADA, leading to the well-known, well-established tandem made by *PostgreSQL and PostGIS*. PostgreSQL (The PostgreSQL Global Development Group, 2021) is a regular object and relational database, with more than thirty years of experience, featuring robustness and performance—two characteristics absolutely necessary for a production server. Along with the database server itself, auxiliary tools such as pgAdmin or psql are provided, facilitating the tasks related to database administration and data handling. However, a regular (even able to handle objects) relational database is not enough to manage data with the said geospatial component appropriately. This is the role of the PostGIS (PostGIS Project Steering Committee, 2021), a spatial extension to PostgreSQL. With PostGIS, it is possible store spatial

objects and then access the database using Standard Query Language (SQL) location-based queries, a key feature for the purposes of the system here described. A typical example of such queries would be "retrieve all the features inside some given rectangle".

Next, a tool to publish the data stored in the database had to be selected. At least, the information had be available as a Web Map Service (WMS) to make possible the implementation of the simple WebGIS described in section 3.3. Ideally, the information should also be provided as a Web Feature Service (WFS), especially taking into account that ADA are basically vector data and that using this service it is possible to access their attributes—such as the deformation time series. Offering data as a WFS layer makes them immediately available to tools like QGIS or ArcGIS. Leaving aside proprietary (and costly) solutions as ArcGIS or MapBox, GeoServer (OSGEO, 2022a) was selected for two reasons: firstly, it supports WMS and WFS out of the box and secondly, the authors had prior experience with this software. These are the reasons why, for instance, MapServer, also an open source and free software, was discarded.

From this point onward, the remaining software components making the full server stack were selected almost as a consequence of the previous decisions. GeoServer requires a Java Runtime Environment (JRE) to run, and since a way to make it available on the Internet is installing it as an Apache Tomcat (The Apache Software Foundation, 2021a) web application, this tool, together with a JRE environment, were included in the pack. Finally, the WebGIS needs a classic web server, so the Apache HTTPD one (The Apache Software Foundation, 2021b) was the last incorporation completing the set of tools making the system.

Figure 2 depicts the architecture of the so-called ADA server, including the relationships between the several software components. This is a rather classic setup, dictated by the needs and constraints of the problem to solve, using a combination of open-source, gratis tools. Although this is not an innovative architecture, it is one that does work and that does it at a low cost.

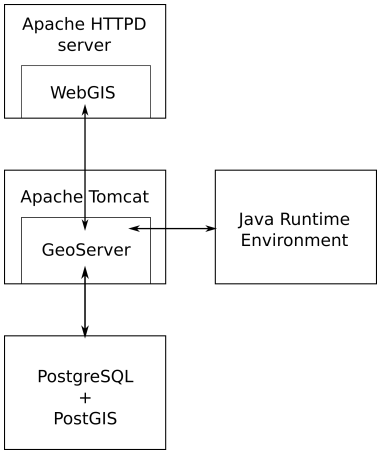


Figure 2. The software architecture of the ADA server.

3.2 From Point Scatterers to Public ADA

The server architecture described in the previous section is able to store, fetch and deliver ADA data via the Internet. However, such data must be first produced and stored there to become available. This section describes the tools required and workflow devised to transform the PS delivered by the EGMS into information ready to be retrieved, either via the WebGIS or as WMS / WFS layers. Figure 3 depicts this workflow.

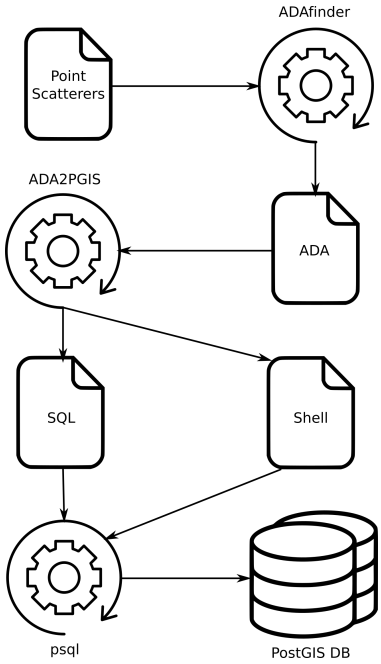


Figure 3. The data production workflow and tools.

Three tools are involved in the production of the published ADA: ADAfinder, ADA2PGIS and psql.

ADAfinder. This is a rather seasoned tool, that went into production back in 2018 and has been involved in the identification of ADA in many past projects, in and out of CTTC (Tomás et al., 2019, Navarro et al., 2020, López-Vinielles et al., 2021).

ADAfinder takes as input a set of PS—in the context of this project, those provided by the EGMS. Its algorithm (Barra et al., 2017) looks for clusters of points that are located close enough and that move beyond some velocity threshold in a coherent way. The output of this process are the so-called ADA, which consist of the polygon defining the area where the movement has been detected, and some attributes such as its centroid and area, the number of PS making the ADA, their minimum, mean and maximum velocities as well as the reliability assessment stating how high is the probability of the said area actually being an ADA.

Additionally, the PS that belong to each ADA—that is, fall inside these—are also output by ADAfinder. This is relevant, since each PS include among its attributes the deformation time series that describe the way it moved over time. In the case of less reliable ADA, this information may be used to analyze whether this zone is actually moving or not. Or, when an ADA is a reliable one, the time series may help to identify the kind of movement. Furthermore, these time series are later averaged by ADA2PGIS (the next tool in the workflow) to provide a global assessment of the movement over time for each ADA. Figure 4 depicts several reliable (thus, in green color) ADA located on a hill just over a highway tunnel. The points (small circles in yellow) are the PS that were used to identify these ADA.

ADA2PGIS. The name of this tool, the second one in the workflow, reveals its purpose: taking the necessary steps to insert the

Figure 4. Some reliable ADA and the PS inside them.

ADA into the PostGIS database. Strictly speaking this is not true, since ADA2PGIS does not perform the actual insertion by itself; instead, it generates a series of SQL scripts containing the necessary instructions to insert the ADA at a later stage (see Figure 3). This approach, deferring the actual responsibility of the insertion, made possible to devise a simpler tool, not having to care about how to interface with the PostgreSQL + PostGIS server—this task is assumed by the PostgreSQL built-in command line tool psql.

Therefore, ADA2PGIS might be seen as a pure format converter, since it takes ADA in one format (ESRI shapefile, the output of ADAfinder) and transforms them into another one (SQL commands). Well, it is *almost* this way. This tool performs an extra task which is worth to talk about. ADA have no related time series information by themselves; such data are present only in the PS that define the said ADA and PS are not stored in the database—this is a decision taken by the development team; furthermore, the PS are not part of the information that must be shown to the users. Consequently, the information concerning the history of the movement would be completely lost if no additional measures were taken. Note that (1) the minimum, mean and maximum values of the velocity of the ADA are part of their attributes and (2) there is a reliability assessment stating how good are the ADA, so *some* information about how they move does exist. However, and as it was stated above, the time series for all the points inside ADA are averaged to provide each of them with a simplified snapshot describing how they moved over time.

The amount of information to dump to the output files with SQL commands is very big. For instance, the polygon defining the contour of the ADA may have several hundreds of points; moreover, the deformation time series may have a similar number of values. This means that the SQL script files may become very large, so much, in fact, that the next step in the workflow, the actual insertion via psql, might fail because of the limitations of this tool. To solve this problem, ADA2PGIS splits the output in as many SQL script files as necessary—according to some parameter given by the user—thus limiting their size. This approach, however would make the next step a bit more complicated, since the operator should take care of running psql as many times as script files were created. To simplify this process and reduce the possibility of errors, ADA2PGIS outputs a second output, a shell (or batch) file, containing the necessary commands to perform the actual insertion into the database. Thus, the operator only needs to type a single command (the name of the output shell file) to finish the process. Figure 5 shows an example of one of these shell files, where two SQL script files are inserted into the database.

```
@ECHO OFF
ECHO Messages are logged to 20220123_psql.log
SET PGPASSWORD=mypassword
ECHO Processing file 20220123_1.sql
psql --host=ADAserver
     --port=5432
     --dbname=adadb
     --username=dbmanager
     --file=20220123_1.sql >> 20220123_psql.log
ECHO Processing file 20220123_2.sql
psql --host=ADAserver
     --port=5432
     --dbname=adadb
     --username=dbmanager
     --file=20220123_2.sql >> 20220123_psql.log
ECHO Please, check the output log file.
```

Figure 5. An example of a shell file running psql, reducing to a minimum the operational complexity.

psql. This is a PostgreSQL built-in command line tool that lets the operator interact with the database. For the ADA production workflow, the operator runs the shell file output by ADA2PGIS; the shell file calls psql to perform the insertion of the ADA using the SQL script files. psql produces an output log that the operator must check in order to guarantee that the insertion process finished successfully. This step is the one finishing the data production workflow-the data.

As soon as the operator completes the workflow, the information thus stored in the database becomes immediately available via GeoServer; that is no extra steps need to be taken for the users being able to obtain it.

3.3 The WebGIS

The WebGIS is, in fact, part of the ADA server's software stack. It sits on top of the several software components integrating such server (Figure 2) and is one of the ways that external users may use to retrieve ADA data—the other one is using directly the WMS and WFS layers.

This one is a simple web application offering a minimum set of features to visualize the ADA. Figure 6, which has been rotated counterclockwise to better accommodate it, shows the WebGIS depicting some synthetic data that was created to assess the performance of the whole system—thus, the perfectly round and evenly spaced ADA. Besides the usual zooming and panning features, it is possible to filter data selecting and combining the values of up to three ADA attributes (area A in Figure 6). Clicking on one of these ADA will bring up a popup window showing the values of its attributes (B). ADA are colored according to the mean deformation velocity attribute (C). Several public background layers (D) are available (OpenStreetMap, Google Satellite or Terrain) to be used as the base cartography. It is also possible to download the current map view as a PNG file (E). At the time of writing this paper, no visualization of the averaged deformation time series is available. This may change in later stages of the project.

The WebGIS was designed to visualize the data using the ETRS89-extended / LAEA (EPSG:3035) coordinate reference system, since this is the one used by the EGMS to deliver the PS and it covers Europe completely.

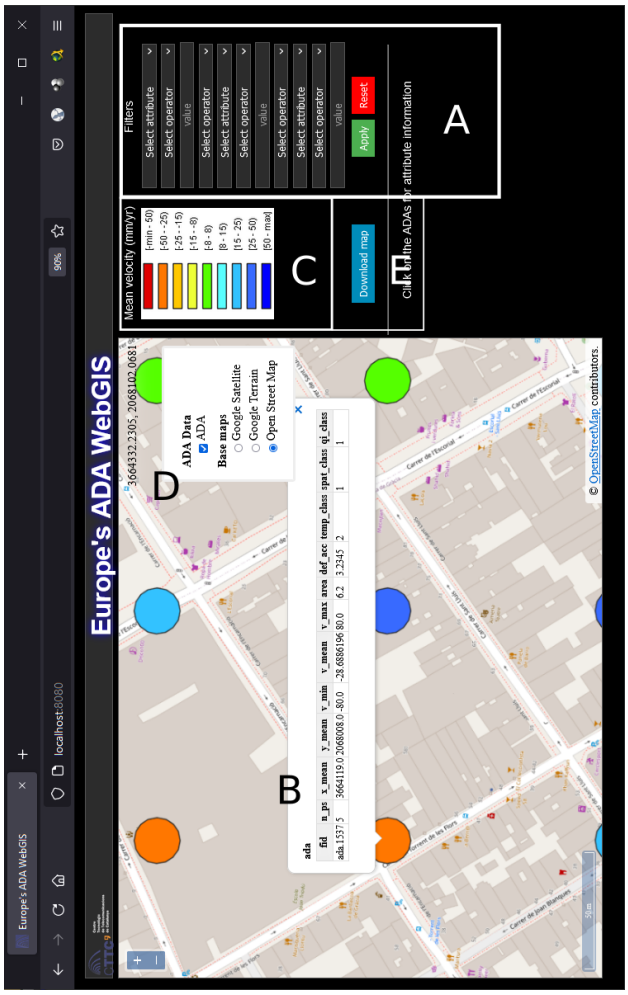


Figure 6. The WebGIS (rotated counterclockwise).

Again, well-known technologies have been used to develop this WebGIS: HTML and JavaScript as programming languages and Cascade Style Sheets (CSS) as the mechanism to define the visualization styles. The OpenLayers library (OSGEO, 2022b) played a very important role to implement the interaction with the GeoServer to retrieve and then display map data. This tool has been designed following the standards and concepts of the Spatial Data Infrastructure (SDI) (Janssen and Dumortier, 2007).

3.4 Design and Implementation Considerations

When designing the system, two important questions arose. These had relevant consequences in how some components should be designed and implemented.

The first one is the CRS used to store and display the ADA and the base cartography used as background. This is a project covering the whole extension of Europe. Typically, CRS valid for much narrower areas (such as the family of UTM coordinate reference systems) are used to characterize spatial data. In this case, such an approach was not possible, since the ADA for all the continent had to refer to a common framework.

Fortunately, the EGMS itself will provide the PS using a unique CRS for all Europe, more specifically the ETRS89-extended / LAEA (EPSG:3035) already mentioned in section 3.3. This eliminates the need to reproject all the incoming data to a common CRS (thus saving lots of effort and possible errors) before executing the data production flow described in Figure 3 and section 3.2; while reprojecting a single data set may imply not too much work, performing this very same task with maybe hundreds of them is not a trivial task in terms of time and effort.

The second issue that had to be considered was the variable structure of the deformation time series accompanying the PS. These are a set of tuples, each of which consists of a date (that of the time in which the measurement was made) and the deformation value itself. Since this project will cover the full extension of Europe, the availability of observations (that is, the set of tuples making the series) will vary between areas. For instance, in some hypothetical "Area 1", the tuples might cover dates Date1, Date3 and Date4, while in "Area 2", the set of dates could be Date1, Date2, Date3 and Date5. Both series are structurally different.

This is a problem when using relational databases, such as PostgreSQL + PostGIS, for data must adhere to a *fixed structure*—all entries in database tables must have exactly the same set of attributes. This means that the structure of the table holding the ADA must be able to cope with all the possible variations of the time series since the very beginning of the project; otherwise, it should be updated *a posteriori* (extending the set of fields for new dates) to cope with deformation time series including observations for times not considered before. Although this is feasible from the technical standpoint, it is a costly operation that, at least in the opinion of the authors, should be avoided at all costs—especially taking into account the big amounts of information that will be stored in the database.

Fortunately, it is possible to check the whole range of dates for which deformation observations will be available all over Europe. Furthermore, relational databases accept null values in their fields, providing these have been defined appropriately. Consequently, the decision taken was that the table including the ADA would include all the possible dates for which at least one deformation observation existed. When inserting ADA with no values for some of the dates included in the table, null values would be used in these cases. For instance, going back to the two example time series above (Date1, Date3, Date4) and (Date1, Date2, Date3, Date5), the resulting all-inclusive time series accepted by the database would be (Date1, Date2, Date3, Date4, Date5) and the following values would be inserted respectively: (Value1, NULL, Value3, Value4. NULL) and (Value1, Value2, Value3, NULL, Value5).

4. PERFORMANCE: PRELIMINARY FIGURES

An extra concern regarding the implementation of the system just described is performance. This concern, in fact, is twofold: in the first place, it refers to the ADA server; the second one is the data production workflow.

Prior to describing the preliminary results obtained with the several benchmarks run to test the performance of these two components, it is important to talk about the equipment used to do it. At the moment of writing this paper, the server had not been yet acquired; this is so because the results of the test were needed to evaluate how powerful this computer should be. Thus, tests took place in a development computer, that of one of the authors of this paper.

The actual production of ADA will take place in the regular computers used by CTTC personnel, as it has always been done

in other production projects. No extra equipment will be acquired thus for this task. Again, the same machine used for the benchmarks related to the ADA server was used for the workflow tests.

The computer was a Dell Latitude 5410, Intel I7 Intel® Core™ i7-10610U CPU @ 1.80GHz / 2.30 GHz, 16 GB RAM, 512 GB SSD.

4.1 The Performance of the Data Production Workflow

The authors already had figures about the performance of the first component included in the data production workflow, ADAfinder, and the prior results thus obtained seemed to point that this tool would be no bottleneck at all. However, and since these results were a bit outdated, new ones were computed using the computer described above.

No actual data coming from the EGMS was yet available, but an existing data set consisting of half million point scatterers was used instead, since this is a rather typical size in these situations. The new results reported that processing these data with ADAfinder takes about 65 seconds. This result was in line with those from older benchmarks, which took a bit more time (about 80 seconds) in an older computer. This conformance of results let us anticipate that even processing data sets with one or two million PS will suppose no problems for a data production chain like this.

No former results, however, were available for ADA2PGIS since this tool has been explicitly developed for this project. Taking as input the ADA produced in the test above, ADA2PGIS needed only 15 seconds to complete the task. Again, this is not a blocker. Such results were, in fact, anticipated. ADA2PGIS is a much simpler tool than ADAfinder (the complexity of ADAfinder's algorithm is much higher) so the authors did not expect performance problems at this point.

A different question was the time that the last step in the workflow would take, the actual insertion of the ADA data into the PostgreSQL + PostGIS database using its built-in command line tool psql. This was a true unknown since the authors had no prior experience in tasks like this. However, the database proved that its reputation is well deserved, since inserting the ADA produced by this benchmark took only one second. This meant inserting just about 1400 ADA—those output by ADAfinder in the first step of the workflow.

The figures above (81 seconds in total) state that there should be no worries about the capacity to process the incoming PS, at least when it comes to processing time.

However, and since tests with big amounts of data had to be performed to check the response times of the ADA server, a synthetic ADA data set was produced. It consisted of 100.000 circular ADA (Figure 6 depict a sample of these); each ADA was drawn using 200 points and every deformation time series included 300 values. The ADA were evenly arranged in a grid, covering a total surface of more than 900 km^2 and a density of more that 11 ADA/km^2, which is absolutely unrealistic, since no concentration of so a big amount of so closer ADA happens in real life. However, it was designed this way to test the server in very heavy conditions (see section 4.1 below).

Inserting this data set using psql took about 2 minutes. Such a result confirms that the insertion process is not an issue.

4.2 The Performance of the ADA Server

The authors believe that the performance results described later in this section must be considered as a worst case of what should be obtained from the actual server hosting the software stack described in previous sections. The reason of such belief is that, although the computer where the tests took place is a rather powerful one to perform development tasks, it may not be compared with the hardware that will be acquired for this purpose. Said that, all benchmarks were made using the synthetic data set described in section 4.1 and the full setup (database, JRE, Apache Tomcat + HTTPD server, GeoServer) of the ADA server.

Using the WebGIS to browse via an WMS layer the area where the ADA had been inserted resulted in refresh times that clearly depended on the zoom level in use: low zoom factors (i.e. far from the ground level) meant, in the worst case, response times of about 7–10 seconds, including the update of the base map layer, to obtain a crisp (not blurred) image. On the contrary, as expected, high zoom levels (i. e., close to the ground) changed the response times to less than two seconds, less than one in most cases. This behavior, however, was expected, and is a clear consequence of the way spatial databases and queries work. When approaching the terrain, the number of features included in the viewport (visible area) decrease, so the amount of information to retrieve, transfer and display is much smaller. Obviously, the opposite is also true.

Connecting QGIS to the same WMS layer resulted in loading, zooming and panning times that might be considered negligible—i.e. almost instantaneous. This result must be interpreted under the light of the software used to display data in both cases. With the WebGIS, the responsibility of loading and displaying map data relies on the OpenLayers library (plus some snippets of supporting HTML code); QGIS takes care by itself of this task. The authors therefore suspect—but cannot state—that the reason for such disparate performances should be explained by the use of the said library, since in both setups (WebGIS vs. QGIS) it is the only different component; the software stack making the ADA server is exactly the same in both cases, so it should make no difference at all.

Switching now to WFS, another test consisted of connecting QGIS to the WFS layer published by GeoServer. In this case, the initial load of the data (that is, showing the full area covered by the grid of ADA) took about 90 seconds. Then, the panning and zooming operations took times very similar to those reported above for the WebGIS (7–10 or 1–2 seconds depending on the zoom level), so both tools show a similar performance. Opening the attribute table—a typical operation in QGIS—took about 20 seconds, which is a rather good result specially considering the huge amount of data (100.000 ADA with time series made of 300 values each) to fetch and display.

Then, QGIS was used to export the WFS layer as a GeoPackage (Open Geospatial Consortium, 2022). The goal was to check the performance of QGIS using the same data set but, in this case, loading the ADA from a local file stored in a popular format. Now, the initial load of the whole data set was much faster, taking about 5–6 seconds. The zooming and panning operations took about half the time than those required by the WFS layer stored in the database. Such an improvement in performance was expected by the authors, since in this context, data are read locally, by means of no intermediaries (PostgreSQL + PostGIS, GeoServer relying on Tomcat, a JRE and an HTTPD server).

These results, taking into account the fact that the computer used for the benchmarks is presumably worse than the server that will be finally used, are promising. Also, the characteristics of the synthetic data set, with an unrealistic extreme density of ADA, are very adverse. However, it must not be forgotten that other components playing an important role in the equation have not been evaluated due to the early stage of the project. The most important of them are the number of simultaneous users and the available bandwidth required to transfer the data between the server and its users. Unfortunately, the authors have no mechanisms to evaluate their impact, so they must remain unknown by now.

5. DISCUSSION AND CONCLUSION

This paper presents the very early stages of a project that must produce and publish a rather big amount of complex information, the Active Deformation Areas of all Europe, which will be accessible either by means of a WebGIS or WMS and WFS layers.

The tool for producing the ADA, ADAfinder, has been used regularly in production at CTTC since 2018 for various projects. Therefore, the main problem to be solved was where the resulting ADA should be stored so that these would be publicly accessible. The natural solution, using the cloud, has been discarded because the cost of such solution exceeds by far the limits set by the budget for the said project. This lead to consider the implementation of an in-house, self-made, low-cost solution.

Such solution comprised two different components: a server able to host and publish the ADA, on the one hand, plus the software chain producing the ADA and inserting these into such server on the other.

A setup including the PostgreSQL + PostGIS spatial database, the Apache Tomcat and HTTPD servers and the GeoServer geospatial data server has been devised and implemented to check whether such an arrangement is able to solve the storage and publishing aspects of the problem. All the software stack is composed of open-source, gratis components, so, besides the cost of the server itself, this approach is ideal from the economical standpoint. This is possible, however, because the CTTC already runs its own data center and that the personnel in charge or maintaining and backing up the server are already on its payroll. Should this not be the situation, these costs should be taken also into account; the amount of these costs has not been assessed. The ADA2PGIS tool has also been developed to implement the insertion of the ADA produced by ADAfinder into the PostGIS database. Finally, an HTML + JavaScript WebGIS relying on the WMS layer published by GeoServer has also been developed.

Testing the architecture above from the functional standpoint has shown that it fulfills the needs of the project. The performance aspect, however, had to be checked to ascertain whether such a setup would match the expected results. For this reason, several benchmarks using a heavy synthetic test data set was used. The figures describing the results using a regular development computer—not a real full-fledged server—show that the performance depends on how the information is accessed (e.g., WebGIS vs. direct inclusion of the WMS layer in QGIS) or the source of the data (e.g., using the WFS layer in QGIS vs. saving it first as GeoPackage and then using this local version of the data). At any rate, these preliminary results show that the system is feasible as it is described by this paper. Such an assessment is not taking into account other conditions like the number of users that might access the system simultaneously or the bandwidth required to provide data to all of them—however, not that many simultaneous users are foreseen.

Either way, the work described here is not a theoretical exercise trying to decide whether this or that approach is valid. The budgetary (and temporal) limitations set by the project, in whose context this system has been developed, are the reasons why such a solution has had to be adopted to meet the conditions of the contract. If these conditions would change in the future—beyond the scope of the said project—the ADA could be then moved to a more powerful environment to facilitate their exploitation.

ACKNOWLEDGMENTS

This work is part of the Spanish Grant SARAI, PID2020-116540RB-C21, funded by MCIN/AEI/10.13039/501100011033.

REFERENCES

Barra, A., Solari, L., Béjar-Pizarro, M., Monserrat, O., Bianchini, S., Herrera, G., Crosetto, M., Sarro, R., González-Alonso, E., Mateos, R. M., Ligüerzana, S., López, C., Moretti, S., 2017. A Methodology to Detect and Update Active Deformation Areas Based on Sentinel-1 SAR Images. *Remote Sensing*, 9(10).

Crosetto, M., Solari, L., Mróz, M., Balasis-Levinsen, J., Casagli, N., Frei, M., Oyen, A., Moldestadk, D. A., Bateson, L., Guerrieri, L., Comerci, V., Andersen, H. S., 2020. The Evolution of Wide-Area DInSAR: From Regional and National Services to the European Ground Motion Service. *Remote Sensing*, 12, 2043.

EEA, 2021. (European Environment Agency). European Ground Motion Service – Copernicus Land Monitoring Service. `https://land.copernicus.eu/pan-european/european-ground-motion-service`. Accessed: 2022-02-18.

Janssen, K., Dumortier, J., 2007. *Research and theory in advancing spatial data infrastructure concepts*. 1st edn, ESRI Press, Redlands, California.

López-Vinielles, J., Fernández-Merodo, J., Ezquerro, P., García-Davalillo, J. C., Sarro, R., Reyes-Carmona, C., Barra, A., Navarro, J. A., Krishnakumar, V., Alvioli, M., Herrera, G., 2021. Combining Satellite InSAR, Slope Units and Finite Element Modeling for Stability Analysis in Mining Waste Disposal Areas. *Remote Sensing*, 13, 2008.

Open Geospatial Consortium, 2022. OGC GeoPackage. `https://www.geopackage.org/`. Accessed: 2022-02-18.

OSGEO, 2022a. GeoServer. `http://geoserver.org/`. Accessed: 2022-02-18.

OSGEO, 2022b. OpenLayers - Welcome. `https://openlayers.org/`. Accessed: 2022-02-18.

PostGIS Project Steering Committee, 2021. PostGIS — Spatial and Geographic Objects for postgreSQL. `https://postgis.net/`. Accessed: 2022-02-18.

The Apache Software Foundation, 2021a. Apache Tomcat© – Welcome! http://tomcat.apache.org/. Accessed: 2022-02-18.

The Apache Software Foundation, 2021b. Welcome! – The Apache HTTP Server Project. https://httpd.apache.org/. Accessed: 2022-02-18.

The PostgreSQL Global Development Group, 2021. PostgreSQL: The world's most advanced open source database. https://www.postgresql.org/. Accessed: 2022-02-18.

Navarro, J. A., Tomás, R., Barra, A., Pagán, J. I., Reyes-Carmona, C., Solari, L., Vinielles, J. L., Falco, S., Crosetto, M., 2020. ADAtools: Automatic Detection and Classification of Active Deformation Areas from PSI Displacement Maps. *ISPRS Int. J. Geo-Inf.*, 9, 584.

Tomás, R., Pagán, J. I., Navarro, J. A., Cano, M., Pastor, J. L., Riquelme, A., Cuevas-González, M., Crosetto, M., Barra, A., Monserrat, O., López-Sánchez, J. M., Ramón, A., Iborra, S., del Soldato, M., Solari, L., Bianchini, S., Raspini, F., Novali, F., Ferreti, A., Constantini, M., Trillo, F., Herrera, G., Casagli, N., 2019. Semi-Automatic Identification and Pre-Screening of Geological-Geotechnical Deformational Processes Using Persistent Scatterer Interferometry Datasets. *Remote Sensing*, 11(14).