Towards a User-Friendly Platform for Building Language Resources based on Web Services

Marc Poch, Antonio Toral, Olivier Hamon, Valeria Quochi, Núria Bel

UPF, DCU, ELDA, CNR, UPF
Spain, Ireland, France, Italy, Spain
marc.pochriera@upf.edu, atoral@computing.dcu.ie, hamon@elda.org, valeria.quochi@ilc.cnr.it, nuria.bel@upf.edu

Abstract

This paper presents the platform developed in the PANACEA project, a distributed factory that automates the stages involved in the acquisition, production, updating and maintenance of Language Resources required by Machine Translation and other Language Technologies. We adopt a set of tools that have been successfully used in the Bioinformatics field, they are adapted to the needs of our field and used to deploy web services, which can be combined to build more complex processing chains (workflows). This paper describes the platform and its different components (web services, registry, workflows, social network and interoperability). We demonstrate the scalability of the platform by carrying out a set of massive data experiments. Finally, a validation of the platform across a set of required criteria proves its usability for different types of users (non-technical users and providers).

Keywords: service, workflow, interoperability

1. Introduction

The EU-FP7 PANACEA[1] project (7FP-ITC-248064) addresses one of the most critical aspect of Machine Translation (MT) and other Language Technologies (LT): the language-resource bottleneck. Although most statistical MT engines are language independent, they depend on the availability of Language Resources (LRs) for the language pairs and domains that they cover. The objective of the project is to build a platform that serves as a factory of LRs that automates the stages involved in the acquisition, production, updating and maintenance of language resources required by MT systems (as well as by other LT). Web crawled data and corpora could be processed in the factory to obtain new LR. On the other hand, already existing LR could be enriched using the platform tools.

LR production requires complex language processing chains. Based on a workflow manager, the PANACEA platform allows the user to combine different LR processors in order to build LRs. These processors are deployed as web services (WSs) that may be distributed on different servers and can be used by different users regardless of their location. A big advantage of using WSs is that users do not need to install any tool (LR processors); they can simply send requests and data to the services and obtain the results. From a technical point of view, users can use the tool without access to the source code. The platform is developed in three stages, i.e. there are three development cycles, so that improvements can be made on the basis of subsequent phases of experiments and tests. This paper will describe the second prototype and its validation.

2. Other projects and related work

The research area of Natural Language Processing (NLP) has advanced notably in the last two decades. However, there exists an access barrier as it is not straight-forward to use most of the tools derived from research, at least for non experts. Issues include the use of different formats in different tools (often incompatible), the difficulty to find the appropriate tools, and their complex installation (e.g. dependencies), lack of an easy to use framework to deploy WSs, to mention just a few.

There have been different initiatives to tackle these issues. On the one hand, integrated toolkits and frameworks have been devised. Two examples of these are GATE (Cunningham et al., 2011) for Information Extraction and corpus annotation and NLTK (Bird et al., 2009) for NLP. These toolkits offer an standardized ecosystem, from which the user can access a pool of resources (e.g. NLP tools, corpora), and build upon them. However, they are closely tied to a specific programming language and therefore the interaction with tools written in a different language is not straight-forward (Bone, 2008) or they are not designed to deploy an external tool as a web service.

A related work is U-Compare (Kano et al., 2011)[2]. It provides a GUI that allows users to build workflows using UIMA components. However, it has two drawbacks compared to the current proposal: (i) it only allows to use UIMA components while the PANACEA approach allows to plug any SOAP web service and (ii) it uses a strongly typed system, specifically designed to tackle text mining, while the PANACEA approach is applicable to broader applications (Kano et al., 2010). The PANACEA platform eases the process of deploying WSs independently from the programming language used by the NLP tool to be deployed and does not require a specific client to run the WSs. This can be very important for users and service providers with no expertise in computer science.

The Language Grid (Bramantoro et al., 2010) is another project aiming at the easy integration of WSs that has tackled many different aspects of creating an infrastructure: se-

[1] Platform for Automatic Normalized Annotation and Cost-Effective Acquisition of Language Resources for Human Language Technologies

[2] http://u-compare.org

curity and IPR issues (Murakami et al., 2010), special user interfaces, a registry of services, etc. However, as with some other cases, deploying a tool as a web service can be difficult for non-experts and the whole infrastructure is not designed for processing large amounts of data (corpus processing). The PANACEA platform will show a scalable model that can easily deploy different tools as WSs which can handle large corpora and that is focused in usability for both users and service providers.

The comparison of all these projects, tools and frameworks would require a deeper analysis that exceeds the scope of this paper. During the design phase of the project, different tools and approaches were studied but limited resources and time constrains made it impossible to make all tests and study all possible scenarios (non-experts users, highly skilled users, naive Web Service Providers (WSPs), professional WSPs with security demands, WSPs with massive job requests, etc.). Therefore, one of the main goals of the platform is to use tools, protocols and data formats that can be used by as many other projects and frameworks as possible trying to benefit from everyone's best features depending on the situation.

3. The platform

The factory is designed as a platform of WSs where the users can create and use these services directly or combine them in more complex chains. These chains are called workflows and can represent different combinations of tasks, e.g. extract the text from a PDF document and obtain the Part of Speech (PoS) tagging or Crawl this bilingual website and align its sentences. Each task, e.g. crawl the web, "get PoS tags", etc. is carried out using NLP tools deployed as WSs in the factory.

WSPs are institutions (universities, companies, etc.) who are willing to offer services for some concrete tasks. One of the project objectives is to provide tools and guidelines to WSPs that facilitate the task of providing services (software, temporary files, interfaces, etc.). These guidelines and documents can be found on the PANACEA website[3].

The platform has been implemented as a proof of concept. It can be used to process different kinds of corpora depending on the WSs. Most of the workflows are based on the web crawlers (Mastropavlos and Papavassiliou, 2011) that have been deployed as WSs. From this point of view, most workflows have been designed to process monolingual and bilingual data that is the output of those WSs although most WSs can process other kinds of data (i.e. PDF, DOC, TXT, CSV, XML, etc.).

Another important thing is that although the machine resources of the different WSPs (considering only the project partners and not other external WSPs) are very limited the platform scales to process large amounts of data.

The platform is based on a set of tools developed by myGrid[4] team which are used for the Bioinformatics research. The aim of these tools is to help researchers work with e-Science. Figure 1 summarizes these tools and the role they play in the platform. These tools have successfully been used in different projects and fields (i.e. social science, astronomy, music, chemistry). All different technologies used to develop the platform are explained in this section.

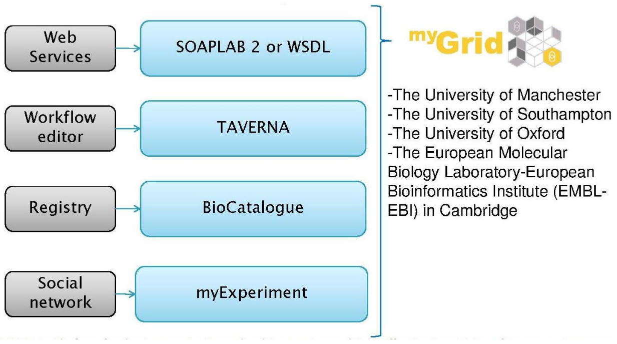
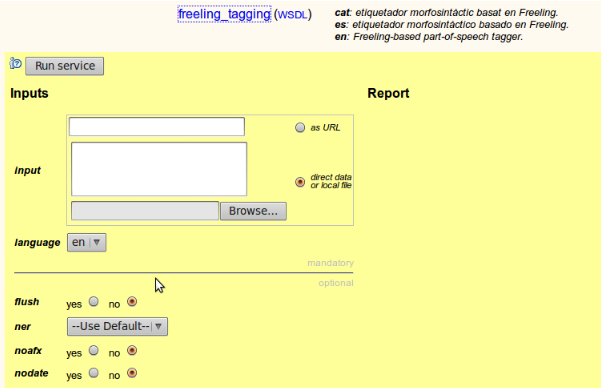

Web Services
SOAPLAB 2 or WSDL
Workflow editor
TAVERNA
Registry
BioCatalogue
Social network
myExperiment
myGrid
-The University of Manchester
-The University of Southampton
-The University of Oxford
-The European Molecular Biology Laboratory-European Bioinformatics Institute (EMBL-EBI) in Cambridge

Figure 1: MyGrid tools

3.1. Web Services

Soaplab (Senger et al., 2003) has been used to deploy WSs.[5] This software allows a WSP to deploy a command line tool as a WS just by writing a metadata file that describes the parameters of the tool. Soaplab is an easy to use tool that automatically takes care of all the typical issues regarding WSs, including temporary files, protocols, Web Service Description Language (WSDL) file and its parameters, etc. Moreover, it automatically creates a Web interface (called Spinet and showed in Figure 2) where WSs can be tested and used with input forms.


freeling_tagging (WSDL)

cat: *etiquetador morfosintàctic basat en Freeling.*
es: *etiquetador morfosintáctico basado en Freeling.*
en: *Freeling-based part-of-speech tagger.*

input: ○ as URL ● direct data or local file

language: en

flush: yes ○ no ●

ner: --Use Default--

noafx: yes ○ no ●

nodate: yes ○ no ●

Figure 2: Spinet web client to run a web service

Soaplab is also adapted to be able to run long-lasting jobs avoiding the web service clients timeouts. When a web service client (i.e. a program, a workflow engine, a web browser, etc.) makes a request to a WS a timer is activated. If the web service does not give the answer in less than the specific timeout the execution will fail (i.e. a web browser produces an error message when a web is not responding after a few time). Soaplab clients can avoid these timeouts by sending periodic requests to check whether or not the job has finished.

[3] http://panacea-lr.eu

[4] http://www.mygrid.org.uk

[5] http://soaplab.sourceforge.net/soaplab2

As its name indicates, Soaplab is based on the widely used SOAP[6] protocol making WSs compatible with different workflow engines, clients and programming languages (e.g. Perl, Ruby, Java, Python, etc.) thus fostering interoperability.

All these features make Soaplab a very suitable software tool for our project. Moreover, its numerous successful stories make it a safe choice; for example, it has been used by the European Bioinformatics Institute (EBI)[7] to deploy their tools as WSs.

PANACEA developers designed a technique to be used with Soaplab to limit the amount of concurrent executed jobs in the WSP server. With a growing number of users all WSPs must pay attention on the machine resources that can be used by users. This technique allows WSPs using Soaplab to easily establish a maximum amount of concurrent jobs and a queue for waiting requests. This technique has been distributed as a software patch[8].

One of the prices to be paid when using WSs is the network usage. SOAP makes use of XML messages to transfer data between computers. Reducing the size of these messages can improve the overall performance of the whole architecture. One of the advantages of Soaplab is that it can use direct data (the data itself is transfered inside the SOAP message) and reference data (only a URL is transfered inside the SOAP message) as input or output. PANACEA developers modified Soaplab with a new configuration parameter that allows WSPs to limit the amount of direct data that SOAP messages can transfer. Therefore, by reducing this limit, users must use URLs to transfer data between computers and the network usage and the memory usage of the workflow editor is reduced drastically. This technique has also been distributed as a software patch[9].

Many different tools have been deployed as WSs by all project partners (WSPs): from Python tools to UIMA components (Prokopidis et al., 2011), all have been successfully integrated and used thanks to Soaplab.

3.2. The Registry

Once the WSPs have deployed their WSs, users need to find those WSs. Biocatalogue (Belhajjame et al., 2008)[10] is a registry where WSs can be shared, searched for, annotated with tags, etc. It is used as the main registration point for WSPs to share and annotate their WSs and for researchers and users in general to find the tools they need. Biocatalogue is a user-friendly portal that automatically monitors the status of the WSs deployed and offers multiple metadata fields to annotate WSs. A very important feature is that users can rate WSs. WSPs share their WSs in the registry but they are responsible of the quality of that service. The catalogue can lists all WSs (high and low quality) and thanks to the rating system users can rate WSs depending on the quality of the documentation, the speed of the WSs, etc. always considering that the WSP may or may not be the tool developer. WSPs with best rated WSs are those with more options to apply for fundings or able to charge users for the service.

PANACEA Registry[11] is an adapted and modified instance of the Biocatalogue. Changes have mainly been focused on the interface layout to suite the PANACEA graphic style. A link to the Spinet form of the WSs has also been added. Then, a few minor corrections have been made within the source code, notably concerning the tool enabling the WSDL parsing. The configuration has also been set up to fit the PANACEA needs, like the status monitoring frequency of the WSs. The PANACEA Registry has 103 registered WSs.

3.3. Workflows

After being able to find WSs in the Registry and testing them with some client, users will be interested in joining WSs to create complex chains. Taverna (Missier et al., 2010)[12] is the myGrid tool devoted to design and run workflows. It makes use of a user-friendly graphical interface in which users can design workflows with drag-and-drop arrows and mouse clicks. Such workflows can be seen like in the Bioinformatics field (and others) as experiments which can be reproduced, tuned and easily shared with other researchers.

An advantage of using workflows is that the researcher does not need to have background knowledge of the technical aspects involved in the experiment. The researcher creates the workflow focusing on high level functionalities (each WS provides a function).

3.4. Social Network

MyExperiment (De Roure et al., 2008)[13] is a social network used by workflow designers to share workflows with the rest of the community. Users can create groups and share their workflows within the group or make them publicly available. Workflows can be annotated with several types of information such as description, attribution, license, etc. Users can use myExperiment not only to share their workflows, but to find examples showing how to build other workflows or how to run some WSs.

PANACEA myExperiment portal[14] is an adapted and modified instance of myExperiment. Likewise the registry, the PANACEA myExperiment has been modified according to the PANACEA graphic style. Also, going with the configuration of the portal, further database corrections have been made to fix some compatibility issues due to system or tool versions differences. The PANACEA myExperiment has 35 registered workflows.

3.5. Interoperability

Interoperability is a fundamental necessity for the platform (Poch and Bel, 2011). This interoperability need was foreseen on the design phase of the project. There are two levels of interoperability that need to be addressed in a factory based on WSs:

[6] Simple Object Access Protocol

[7] http://www.ebi.ac.uk

[8] http://myexperiment.elda.org/files/4

[9] http://myexperiment.elda.org/files/5

[10] http://www.biocatalogue.org

[11] http://registry.elda.org

[12] http://www.taverna.org.uk

[13] http://www.myexperiment.org

[14] http://myexperiment.elda.org

1. The data being transferred between components must follow an interoperable format. Tools must be able to process this format which is being transferred across the factory. This data object was called Travelling Object (TO) because of the distributed nature of the factory (i.e. WSs deployed in different locations).

2. The other aspect regards the parameters of the WSs. All WSs must use the same naming convention for parameters, not only to help developers but also for automatic processes to check compatibility, etc. However, some technical aspects of these parameters also need to be established. For example, if the parameter is optional or mandatory. To this aim, a Common Interface[15] (CI) was created for all WSs deployed to work in the factory.

The first TO designed is based on the XCES standard (Ide et al., 2000) and it was chosen because it is the format that requires the minimum amount of changes between the inhouse formats of the tools. At a later stage, in order to ensure wider standardization/interoperability, another TO has been adopted implementing the Graph Annotation Format (Ide and Suderman, 2007), which is the XML serialization of LAF (ISO 24612, 2009). It is used to create standoff annotations for some of the outputs of the platform.

The CI was designed during the design phase of the project. It was designed according to the first tools to be deployed as WSs for the first version of the platform: tokenizers, sentence splitters, Part of Speech (PoS) taggers, etc. The common parameters for the tools to be deployed for every functionality (tokenization, sentence spitting, PoS) were studied and considered to be the mandatory parameters for the CI. On the other hand, optional parameters can be used freely to configure the specific idiosyncrasies of the tools. This way of deploying WSs facilitates their usage and makes it really easy to chain them in workflows or to change a specific web service with another one performing the same functionality To summarize, the CI established which are the mandatory parameters for every functionality. The CI will be enriched with new functionalities for the final version of the platform.

4. Massive data

The aim of the so called massive data experiments is to prove that the whole architecture of the platform can scale and is robust enough to a growing input data and number of requests. Even that the partners' servers (test WSPs) are modest[16], the idea is to show the system is robust enough and that it can grow to handle more data and requests if provided with more machine resources.

To test the platform capabilities the idea was to stress a concrete WSP server. Every partner designed a workflow with different WSs all of them deployed on the server to be tested.

4.1. First version of the platform

The first workflows were designed using the first version of the platform. For the first version of the platform the main goal was simply to chain components and no special features or workflow design guidelines were used. As expected, long lasting WSs executions hit the Taverna timeout and the corresponding files were not processed. Some bugs were detected on Soaplab (i.e. error with parameter names with underscore). They were reported and fixed by Soaplab developers in its last version[17].

4.2. Second version of the platform

For the second version of the platform, different improvements and techniques were studied and implemented to handle large corpora. First, since the idea is to test a single server with numerous requests it is important to handle temporary files. As mentioned before, the typical used server is a Virtual Machine with an average of 50 GB of disk space. Every partner used its own way to automatically erase old temporary files, e.g. using automatic and periodic calls to a shell script program on Linux servers.

To avoid hitting the timeout limit of Taverna, workflows must be designed using the Soaplab ”polling” technique. Each Soaplab WSs must be configured with polling parameters to make sure Taverna makes periodic requests to the service. These requests will check the service status until the job is finished and finally Taverna will get the service result.

Another Taverna feature which is very valuable to add robustness to the workflow is the retries system. Every WS in a workflow can be configured to be retried in case of an error during execution. The workflow designer can set the amount of retries before considering that job erroneous and the time between those executions.

Taverna parallelization feature can be used to make parallel calls to a WS while running a workflow. This can be used to improve the whole workflow performance and reduce the total amount of time to process all input data. Moreover, parallelization adds robustness to the workflow: if for some reason a WSs execution hangs and takes too much time to finish, the rest of input data can be processed using the other parallel instances of that particular WS.

Using the PANACEA software patch for Soaplab to limit the SOAP messaging is very important to reduce the network usage and to limit the amount of memory used by Taverna (memory used to process all Soap XML messages).

The first tests showed that long lasting executions could be processed and that parallelization added robustness and improved throughput.

However, for experiments of more than 500 inputs (in this case XML files) Taverna 2.3.0 could not save the results. This bug was reported to myGrid developers and fixed in Taverna 2.4. Another detected severe error was the ”waiting for data” message in any of the iterations of a workflow (most workflows have one iteration per input file). The ”waiting for data” message is used to show that a concrete iteration is waiting for a WS response. Due to a bug, some iterations were not able to escape from this waiting state

[15] http://panacea-lr.eu/en/info-for-professionals/documents

[16] most of them provide: 2 cores, 4GB RAM, 50 GB disk space usually deployed as a Virtual Machine

[17] Soaplab 2.3.2

even when the WSs had finished and sent its response. Instead of using the retry system to actually retry the job Taverna leaves that iteration on hold while the others are normally executed. When all data has been processed, iterations which are still waiting for data make the workflow execution never end. This was for some months the main reason for not fulfilling the massive data scalability tests and not being able to present satisfactory experiment results until now.

Working in collaboration with Taverna developers resulted in PANACEA developers testing Taverna 2.4 (a beta version not released to the public at that moment). This new Taverna fixed those bugs and our experiments confirmed the scalability of the platform. Figure 3 illustrates a typical scalability test designed to test server “iula04”. The workflow has two WSs: PoS and a converter to the TO. Both WSs are configured with 3 parallel processes generating a total 6 parallel processes on the server. The corpus being processed is the result of a crawling task with 60M tokens and 13K files stored in a different server. The whole corpus was downloaded and processed in approx. 5 hours.

4.3. Third version of the platform

The final report on scalability is under development since the third version of the platform is not final yet. It will contain experiments carried on different WSPs servers and using different kind of tools and workflows. EBI has been using Soaplab to deploy their massively used WSs for years (McWilliam et al., 2009) and with our massive data scalability experiments we will demonstrate that with more machine resources more jobs can be served from WSPs servers. With more machine resources the growing number of requests (more users or more parallel jobs) will be fulfilled by WSPs.

Videos, guidelines, tutorials and general documentation can be found on the PANACEA website[18] for more detailed information about all these topics and the platform in general.

5. Validation

This section describes the assessment done within the PANACEA project of the platform machinery/technology mainly from technical perspective. An evaluation of the impact on SMT of the resources produced using the services integrates into platform is reported in (Pecina et al., 2011). Also, an industrial evaluation of the platform and its technologies will be performed at the end of the project for the final prototype of the platform.

The validation of the platform is performed on the basis of a set of criteria previously defined (see 5.1.) by three validators (5.2.) in order to determine whether required criteria are compliant with expectations; therefore, there are no validation scores: a requirement is either validated or not on a binary scale.

5.1. Validation criteria: requirements to be checked

The criteria for validation had been defined at the start of the project on the basis of the expected requirements and functionalities of our platform (for the automatic creation of language resources). Here we briefly mention the most salient criteria for the scope of the paper, a complete list and specification of the various criteria can be found in the project deliverable $D7.3$[19].

The Registry. A set of criteria are defined for checking the availability of the registry, its searching mechanisms and the possibility of adding services by service providers and the like.

Web services: The set of criteria for checking the availability and accessibility of the platform processors, i.e. WSs; the availability of metadata and closed vocabularies for their description and categorization; and that error messages and exceptions are handled satisfactorily. Interoperability criteria have also been defined to check availability and compliance to the Common Interfaces and Traveling Object.

Workflows: a set of criteria defined for checking the functionality of a workflow editor; it’s handling workflows executions, of provenance (e.g. errors, timestamps, etc.) and intermediate data; and the possibility of remotely executing workflows, which is an relevant feature for long lasting workflows.

5.2. User profiles and validators

Validation of the platform has been organized on the basis of two prospective typical users: platform users, and WSPs.

Platform users aim at using WSs and workflows already designed, or at building new workflows using the available WSs. Service providers aim at deploying and sharing their tools within the platform, through WSs and workflows, they may also want to build workflows using their tools in pipeline with tools by other providers.

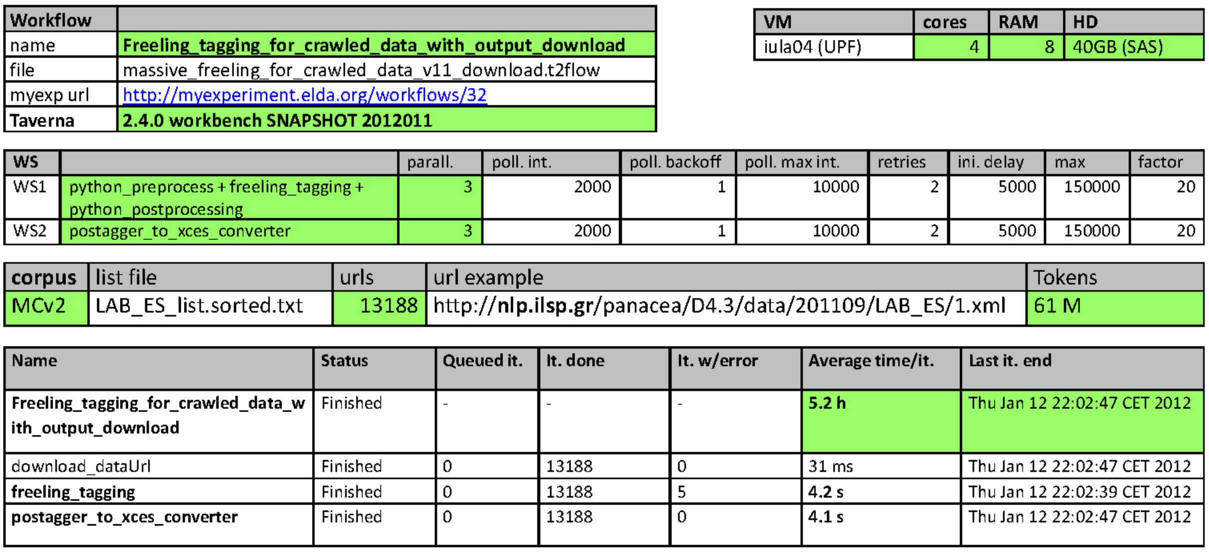
3 Validators were recruited so as to fit user types (i.e. platform user vs. service provider); two of them are researchers of the project active in the production of some components of the platform, but not directly involved in the platform design and development; while one validator, with the user profile, who had not been involved in service development has been selected to act as an ”external” validator.

5.3. Validation scenarios

The platform validation is based on scenarios. For the validation of this second version of the prototype, 5 scenarios have been defined taking into account both the user profiles (i.e. Service Provider and Platform User) and the criteria to be validated for the second version of the platform prototype. Each scenario aims at validating some of the criteria defined according to the user type and consists of a sequence of steps or tasks that the validator has to perform and a questionnaire the validator fills in at the end of the steps. The questions proposed to the validators required either yes/nos or open answers. Free feedback was also allowed in order to be able to make an overall qualitative assessment of the platform and to gather useful suggestions for further improvements.

[18] http://panacea-lr.eu

[19] Available at www.panacea-lr.eu



Workflow	
name	**Freeling_tagging_for_crawled_data_with_output_download**
file	massive_freeling_for_crawled_data_v11_download.t2flow
myexp url	http://myexperiment.elda.org/workflows/32
Taverna	**2.4.0 workbench SNAPSHOT 2012011**

VM	cores	RAM	HD
iula04 (UPF)	4	8	40GB (SAS)

WS		parall.	poll. int.	poll. backoff	poll. max int.	retries	ini. delay	max	factor
WS1	python_preprocess + freeling_tagging + python_postprocessing	3	2000	1	10000	2	5000	150000	20
WS2	postagger_to_xces_converter	3	2000	1	10000	2	5000	150000	20

corpus	list file	urls	url example	Tokens
MCv2	LAB_ES_list.sorted.txt	13188	http://**nlp.ilsp.gr**/panacea/D4.3/data/201109/LAB_ES/1.xml	61 M

Name	Status	Queued it.	It. done	It. w/error	Average time/it.	Last it. end
Freeling_tagging_for_crawled_data_with_output_download	Finished	-	-	-	**5.2 h**	Thu Jan 12 22:02:47 CET 2012
download_dataUrl	Finished	0	13188	0	31 ms	Thu Jan 12 22:02:47 CET 2012
freeling_tagging	Finished	0	13188	5	**4.2 s**	Thu Jan 12 22:02:39 CET 2012
postagger_to_xces_converter	Finished	0	13188	0	**4.1 s**	Thu Jan 12 22:02:47 CET 2012

Figure 3: Typical massive data scalability test

5.4. Validation material and procedure

Tutorials and videos prepared for documentation of the platform[20] were provided to validators who were required to read them at least once during the training phase. Also, material was prepared for use by the validators with the services in order to have controlled and comparable outputs.

After the training phase, the scenarios with task descriptions and question forms were provided to validators. After the validation was done, validators returned their forms which have then been analysed so as to learn the lessons of the task and improve the PANACEA platform.

5.5. Validation results

A quantitative and qualitative analysis of the returned forms reveal an overall good performance of the platform: most of the requirements are validated and the platform realizes its main expectations. Table 1 gives an overview of the validated fulfilled and unfullfilled criteria.

Criteria	**Fulfilled**	**Unful.**
Registry searching mechanisms	X	
Adding services	X	
Components accessibility	X	
Common interface compliance	X	
Metadata description		X
Error handling	X	
Exception management	X	
Workflow execution monitoring	X	
Workflow execution provenance	X	
Workflow execution error messaging	X	
Workflow intermediate data inspection	X	
Remote workflow execution		X
Interoperability among components	X	
Common Interfaces design	X	
Proprietary data management	X	
Traceability	X	
Service bug reporting		X
User feedback	X	
Administrators' Documentation	N/A	N/A
Total (19)	**15**	**3**

Table 1: Summary of the second validation results

The Registry, *myExperiment* and the WSs are available and easily usable. Taverna, the workflow editor, can also successfully been used for running and creating workflows, although some problems and failures were reported. Thus, workflow management can be improved.

Regarding the registry, indeed validators were ease with its use and found its navigation natural. In particular, they mentioned the different views, the filtering options using categories and the WSs status as interesting and useful. However, improvements were suggested on the level of search functionalities for helping users in locating the desired WSs: for instance, providers should give more annotations to their WSs, or the search engine could be enriched with synonyms or new terms. The registration of a new WSs is not always clear: the distinction between SOAP and Soaplab is rather confusing and the URL to submit is not clear and well defined. Also, metadata descriptions were insufficient according to validators. Metadata guidelines were in fact not available at the time of validation and are still under construction.

For the PANACEA *myExperiment* portal, similar considerations apply: search functionalities should be improved, workflows more extensively tagged and categorized so that users can find them easier.

The use of *Taverna* was reported to be rather easy for processing a simple existing workflow, as well as for combining WSs into workflows. The error management and notification are altogether sufficient, especially the visual one within the workflow graph. The same applies to the Spinet

[20] http://panacea-lr.eu/en/tutorials/

error management. However, the display of errors could generally be improved, especially the Java error trace that may be hard to follow by a non-technical user.

Documentation comes out as the main issue to be improved. The current videos available to users proved useful and helped validators finding solutions and using the platform. However, the specific documentation of individual services and of workflows was still weak, which is mostly the responsibility of the service provider. This obviously does not concern all the WSs and it occurs at different levels but it appears to be the major factor hampering the usability of the platform. Web service and tool documentation should in particular address issues such as: input/output formats and tagsets, parameters setting, compatibility with other services or special requirement when put in workflows. While not damaging the overall technical operativity of the platform, this is instead an important recommendation to the (academic) community of tool developers, esp. if they want their tools be spread and used. This issue has been adressed since the validation results and PANACEA WSPs have improved their tools documentation.

6. Conclusion and future work

This paper presented the PANACEA project platform designed to create and process LRs. A set of tools from the Bioinformatics field are used to deploy WSs based on NLP tools, make workflows, share WSs and workflows. The paper presents these tools and their adaptation to suite the project requirements as well as the work developed to foster interoperability between the WSs. The platform is now a test case to study its possible exploitation and scalability. To this aim, “massive data scalability tests” are made to test developments and the presented workflow design methods. The platform has proven to be based on very usable and interoperable tools that can represent a change in the way the LRs are processed. WSs reduce the amount of resources users need to devote to tools. They are a specially interesting solution for proprietary software which cannot be freely distributed, but for which remote use can be allowed. Although the platform is still under development, the WSs, the Registry and myExperiment can be used.

The combined experience from all the relevant projects and initiatives in the field can foster this change if cooperation is promoted and specially if interoperability between tools and frameworks is promoted too. Users, institutions and companies could be interested in different aspects or tools from the different projects and frameworks presented. As in many other research fields, interoperability and usability are key aspects for the success of platforms like the one presented in this paper. Usability not only from the point of view of the user, but the WSP that has to deploy different WSs and maintain them. Therefore, the ideal scenario could be based on giving all kind of users and providers the best options for their needs: i.e. 1) a PhD student could benefit from the Registry and Soaplab to easily find and test tools deployed as WSs (even proprietary tools which code cannot be shared but can be shared as a service); 2) a company using UIMA could deploy payment WS which could be called from GATE, Taverna or any other tool; 3) SOAP services with massive data capabilities could be shared in the Language Grid, etc. These and many other situations based on interoperability and collaboration could foster the necessary critical mass to start a new culture based on distributed services that can represent new research and business opportunities.

7. Acknowledgements

We would like to thank the developers of myGrid, Soaplab, Taverna, myExperiment and Biocatalogue for solving our questions and attending our requests. This research has been funded by the EU project PANACEA (7FP-ICT-248064).

8. References

Khalid Belhajjame, Carole Goble, Franck Tanoh, Jiten Bhagat, Katherine Wolstencroft, Robert Stevens, Eric Nzuobontane, Hamish McWilliam, Thomas Laurent, and Rodrigo Lopez. 2008. Biocatalogue: A curated web service registry for the life science community. In *Microsoft eScience conference*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit.* O’Reilly, Beijing.

Paul Bone. 2008. Integrating NLTK with the Hadoop Map Reduce Framework, http://ww2.cs.mu.oz.au/ pbone/papers/nltk-hadoop.pdf.

Arif Bramantoro, Ulrich Schfer, and Toru Ishida. 2010. Towards an integrated architecture for composite language services and multiple linguistic processing components. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6).*

David De Roure, Carole Goble, and Robert Stevens. 2008. The design and realisation of the myexperiment virtual research environment for social sharing of workflows. *Future Generation Computer Systems*, 25:561–567, May.

Nancy Ide and Keith Suderman. 2007. GrAF: A Graph-based Format for Linguistic Annotations. In *Proceedings of the Linguistic Annotation Workshop*, pages 1–8, Prague, Czech Republic, June. Association for Computational Linguistics.

Nancy Ide, Patrice Bonhomme, and Laurent Romary. 2000. XCES: An XML-based encoding standard for linguistic corpora. In *Proceedings of the Second International Language Resources and Evaluation Conference. Paris: European Language Resources Association.*

Y. Kano, P. Dobson, M. Nakanishi, J. Tsujii, and S. Ananiadou. 2010. Text mining meets workflow: Linking u-compare with taverna. *Bioinformatics*, 26(19):2486–2487.

Y. Kano, M. Miwa, K. B. Cohen, L. Hunter, S. Ananiadou, and J. Tsujii. 2011. U-compare: a modular nlp workflow construction and evaluation system. *IBM Journal of Research and Development*, 55(3):11:1 – 11:10.

Nikos Mastropavlos and Vassilis Papavassiliou. 2011. Automatic acquisition of bilingual language resources. In *Proceedings of the 10th International Conference of Greek Linguistics*, Komotini, Greece.

Hamish McWilliam, Franck Valentin, Mickael Goujon, Weizhong Li, Menaka Narayanasamy, Jenny Martin, Teresa Miyar, and Rodrigo Lopez. 2009. Web services at the European Bioinformatics Institute-2009. *Nucleic acids research*, 37(Web Server issue):W6–10, July.

Paolo Missier, Stian Soiland-Reyes, Stuart Owen, Wei Tan, Aleksandra Nenadic, Ian Dunlop, Alan Williams, Thomas Oinn, and Carole Goble. 2010. Taverna, reloaded. In M. Gertz, T. Hey, and B. Ludaescher, editors, *SSDBM 2010*, Heidelberg, Germany, June.

Yohei Murakami, Donghui Lin, Masahiro Tanaka, Takao Nakaguchi, and Toru Ishida. 2010. Language service management with the language grid. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Pavel Pecina, Antonio Toral, Andy Way, Vassilis Papavassiliou, Prokopis Prokopidis, and Maria Giagkou. 2011. Towards using web-crawled data for domain adaptation in statistical machine translation. In Mikel L. Forcada, Heidi Depraetere, and Vincent Vandeghinste, editors, *Proceedings of the 15th Annual Conference of the European Associtation for Machine Translation*, pages 297–304, Leuven, Belgium. European Association for Machine Translation.

Marc Poch and Núria Bel. 2011. Interoperability and technology for a language resources factory. In *Proceedings of the Workshop on Language Resources, Technology and Services in the Sharing Paradigm*, pages 32–40, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Prokopis Prokopidis, Byron Geograntopoulos, and Haris Papageorgiou. 2011. A suite of nlp tools for greek. In *Proceedings of the 10th International Conference of Greek Linguistics*, Komotini, Greece.

Martin Senger, Peter Rice, and Thomas Oinn. 2003. Soaplab - a unified sesame door to analysis tools. In *All Hands Meeting*, September.