Revista de Sistemas de Informação da FSMA
n. 21 (2018) pp. ??-??

SISTEMAS DE INFORMAÇÃO
http://www.fsma.edu.br/si/sistemas.html

TRILHA PRINCIPAL

Integration of Oceanographic Numerical Modelling Systems and Autonomous Computational Systems Using Web Services

Patrick Serpa[1,2], Leila Weitzel[1], Leandro Calado[2]
[1]Computer Science Department - Sciente and Technology Institute - UFF - Rio das Ostras Campus
[2] Admiral Paulo Moreira Sea Studies Institute - IEAPM, Arraial do Cabo - RJ, Brazil

***Abstract*—Operational Oceanography is a reality nowadays, making it necessary for computer systems to improv the numerical models and the feasibility of numerical oceanographic systems. The main goal of this research is to study and design a system that uses Web Services paradigm in order to integrate oceanographic numerical modeling systems and autonomous computational systems. The system aims to provide an abstraction layer between applications such as Matlab, Fortran, Python, Java Script and the underlying network infrastructure. It is based on the REST architecture and the JavaScript Object Notation (JSON) protocol. Thus, both internal modules and external systems can communicate in a distributed, low-coupling and transparent manner, regardless of their location, development language and hardware platform. For Operational Oceanography, these characteristics bring some advantages including, for instance, the fact that it provides greater processing power for the execution of numerical models using specific hardware and it makes it possible to integrate new numerical models. Hence, the proposed system enables the evolution of the Numerical Oceanographic Prediction System, specially core operations of the Almirante Paulo Moreira Institute of Sea Studies (Brazilian Navy), as well as other oceanographic researches in general.**

***Index Terms*—Web Service, REST, Oceanography, Numerical Forecast**

I. Introduction

Oceanography (or Sea Sciences) is a branch from the geosciences that is dedicated to the stduy of the oceans and coastal zones, in their physical, chemical, biological and geological aspects [1].

Atmospherical conditions forecast have improved markedly since the beginning of the XX centry, from Richardson's pioneer work [2], until today, given the best knowledge about the meteorological and oceanographic phenomena, the refining of the numerical solution techniques and to the exponential increase of the computational capacity.

The existance of new technologies and equipments in this field, such as research ships, meterorological and oceanographic buoys, orbital data (from satellites) and Global Positioning Systems (GPS) have widened the context and the width of an investigation in the maritime environment, making it an interdisciplinar study in synchrony with the technological changes [3].

Hence, since the end of the 20th century, oceanography has transitioned from research to operational mode. Operational Oceanography (OO) is made of the following three basic components:

- Ocean monitoring system through both in situ and remote sensors data gathering;
- Oceanographic Numerical Forecasting Systems (SPNO), which are able to offer diagnostics and forecastings about the ocean; and
- Post processing and data storage operational system, and the availability of the data to the whole community, using the Internet.

The first component of OO may be defined as supplying scientific information and forecast on the state of the sea in a routine basis and quickly enough so that the users can use this information to make decisions before there are significative changes in the environmental conditions [4]. Nowadays, those forecasts, both in a small time scale (a few days ahead) both in climatic scale (months and years) are part of the daily activities of most comunities.

The second componenet, a SPNO, uses a series of numerical models using the number supplied by the monitoring systems. We can define numerical modelling as the process by which a set of different equations and boundary conditions which together are transformed into systems of algebric equations that are solved using computational techniques [5]. These, in turn, generate the Oceanographic Numerical Models, which are computational tools used to investigate and forecast the sea physical conditions.

It is important to remeber that these routines have been fully functional for some years at the Paulo Moreira Sea Studies Institute (IEAPM)), having been developed by geologists,

geographers and other expert professionals. These researchers do not have the expertise in Computer Science, being, thus, unable to implement the numerical modelling through other paradigms such as Artificial Intelligence. Hence, this research becomes important in order to help orchestrate a set of routines to decrease the time necessary to forecast the sea physical conditions.

These models use measurements and theories about the ocean beahvoir, so that they make it possible to simulate and forecast the oceanic processes. Therefore, in order to forecast the sea and ocean conditions, there are measurements from many different characteristics, sch as the water surface temperature, the intensity and the direction of the wind, maritime currents, height and direction of waves, the waves period and many others. These characteristics are examples of data with variations in time and space, so that there are multiple variables with different magnitudes to represent.

Oceanic numerical models have evolved a lot in thelast years, not only because of the better understanding of the physical processes, but also because of the big increase in computational power, which allows not only for faster processing but also to a more faithful representation of the differential equation by its algebric equivalents. Such a revolution in computational power made ocean modelling practical, which, together with the availability of ever cheaper data gathered by remote sensors on satellites, results in a global data set [4].

The third component is part of an operation system whose purpose is to perform data post-processing, store the data gathered and generated and to allow for the distribution of the data and to provide a set of services with electronic documents which will be made available to the community through the Internet.

The Web Services technology allows both internal modules and external systems to communicate in a distributed way, with low coupling and in a transparent way, independent of their location, programming language and hardware platform. For OO, these characteristic bring several advantages, for they allow the existence of greater processing pwer for the execution of numerical models using specific hardware, the evolution to SPNO integrating new numeric models to improve the forecasting techniques and both gathering and distribution of data to be performed in a more efficient way.

Hence, given the context described in the previous paragraphs, we can state that the goal of this work is to present a case study of using Web Services to integrate oceanographic numerical modeling sustems with autonomous computationals systems, so that these applications can interact.

With the systems integration, we expect to offer subsidies for the operations of the Admiral Paulo Moreira Sea Studies Institute (IEAPM), an organ from the Brazilian Navy, and to the oceanographic researches performed in the southeastern region of Brazil, by offering a oceanographic forecast model. Besides, the applications integration will allow for more efficiency and autnomy for all researches in this domain.

This paper is organized as follows. In the second section, we offer a brief theoretical foundation on Web Service and REST. In the third section we present the application domain, that is, the SPNO processes. In the fourth section we present an experimental design and finally, in the fifth and last section, we offer a discution of the processes and sugest future works.

II. IEAPM's SPNO

The research concerning SPNO at the Admiral Paulo Moreira Sea Studies Institute (IEAPM) is performed by the Numerical Modeling Research Group, where many professionals, from graduate students up to invited forein professors, contribute to the evolution of the system.

The studies are based in two components: the data gathering system, which work as a data gathering and distribution platform, and the SPNO, which includes a set of methods which forecast the evolution of the oceanographic conditions.

Nowadays, the process of gathering, processing, analyzing and distribution data is performed by experts. The forecasting is perform in an isolated way with no communication between the steps, requiring that in each step the input and output variables depend on the expert actions, which causes a dependence on his availability in order for the process to be made in a timely way.

The whole process to perform the sea conditions forecasting at the IEAPM is easy to explain. First, the unprocessed data is gathered from servers from environmental institutes physically scattered around the world. These data are collected by sensors of different kinds. Processed data are algo gathered. All these data thgether form the imput for several numerical forecasting calculations. Next, the calculations are performed locally or remotely in several platforms which use different programming languages (such as Fortran and Matlab, among others), each one generating its own output formats. Everything must be "orchestrated" by a system that automatically seeks the data (or the processing results) and sends them to other processing steps (either in the same or in another server) and receive as result the forecasting of climatic conditions for a specific time period and next pubishes (and/or stores) the results in a server for the other researchers to access them.

In Section V, "Processes from the Oceanographic Numerical Forecasting Systems (SPNO)", we describe in details how the forecasting is performed, adding some figures to make it clearer for the reader. Figure 5 shows schematically how the webservices were "orchestrated" in this research and how the steps described above were modelled.

The gathered and generated data by all the OO components are fundamental not only to the SPNO itself but also to the external public (from different research areas). At the IEAPM, for instance, the Submarine Acoustic Division uses the forecast in its acoust forecast models and in submarine communications. On the other hand, the Meteorology area also uses the information generated by the SPNO in its forecasting models and in its analysis, in order to make it easier to understand wheather phenomena.

III. Related Work

We found no papers in the context of the application of Web Services to Oceanographic Numerical Forecasting Systems in order to manage the communication and interaction among modules. Nevertheless, Ferreira [6], in his dissertation "Middleware architecture for Pervasive Internet", found out that it was possible to control and monitor the state of devices, bringing about the necessary interoperability, scalability and mobility for the involved parties using the UPnP (Universal Plug and Play), REST and ZigBee (IEEE 802.15.4) technologies over IP networks.

Ono et al. [7], on the other hand, presented the Cytoscape application, which allows the analysis, visualization and publicatoin of complex networkds, available in several domain types, including bioinformatics, social network analysis and semantic web (with the CyREST module).

CyREST is a RESTful module which allows the communication with applications in different languages, such as Python, R, Matlab and Java Script. In the requests and respondes, the data are transmitted in the JavaScript Object Notation (JSON) format.

In the model presented by the authors [7], the Cytoscape layer represent tools that perform the requests and receive results from cyREST, which communicates with libraries developped to be a middleware for the passing of parameters from the upper layer to the available local systems.

Poulter, Johnston e Cox [8] study the use of the MEAN tool stack (MongoDB, Express, Angular e Node.js) in the development of backend services and web interfaces for devices connected to the Internet of Things.

These authors implemented a RESTFul API allowing for four operations (POST, GET, PUT & DELETE) on all data and, just like Ferreira [6] presented the Slave Controller element, they used a dedicated micro controller to act as an interface to the lower level hardware and from this modulra format, they derived more flexibility for a future device replacement.

These authors also proposed that the software should not be restricted to sending data from the sensor to the central repository, but that there should also be reverse interactions, allowing for change of parameters, such as sensor reading frequency and even the sending of a firmware update to the device.

From the point of view of SPNO and its intergration with its modules, the Web Services technology presents itself as a solution in which systems can interoperate, that is, it is possible for a system or products to interact with another system or product without a specific user intervention [9].

IV. Theoretical Framework

A. Service Oriented Architecture (SOA) and Web Services

According to the SOA reference models, published by the Organization for the Advancement of Structured Information Standards (OASIS) [10], SOA is defined as a paradigm (concept) for the organization and use of distributed resources that are under control of different proprietary domains. It offers an uniform way to offer, discover, interact and use habilities to generate the desired effects that are consistent with pre-conditions and have measurable expectations [10].

SOA is an architecture that does not depend on the technology to define, register and invoke services [11]. A service is a mechanism to offer access to one or more resources, in which the access is offered through an interface (service register) described with restrictions and policies specified in the service description and which is offered by an entity (service provider) for the use of a third party (service clients) [10].

SOA is an architectural concept, a way of thinking and designing the integration among business services. The integration among several services (where each service can be a part of a system) is the basis for the business process.

In SOA, these services are connected to a central bus called ESB (Enterprise Service Bus). This bus has a series of connectors that allow for the communication with different technologies, from SOAP (WebService), CORBA, RMI to legacy technologies. The most widely used technology used in this model is Web Services [12], [13].

Web services are a service that communicates with the clients through a standardized set of protocols and technologies. There protocols and technologies are implemented in platforms and products, allowing for clients and services to communicate in a consistent way even if they are in different operational environments [13].

According to the W3C [14], the definition of Web Services is given as a software system developed to allow for machine-machine interactions, through a network, It supplies a service description interface through which the systems interact with the Web Services, using SOAP messages, usually sent through HTTP together with ther web related standards. Nevetheless, this definition is not restrictive and does not assume that SOAP is the olny processing model format.

The SOAP (Simple Object Access Protocol) protocol was created with the first definitions of Web Service, but the REST (REpresentational State Transfer) architectural style has been gaining popularitiy among Web Services developers. Companies such as Google, Facebook, Twitter and Amazon already make available REST style Web Services to allow for access to their services [15].

We chose to use REST in this work because of the reasons described in the previous paragraphs. Also, ERDDAP[1], a system which is integrated with more than 50 environmental institutions, also makes available the data used in this research through RESTful Web Services, an implementation that follows the principles of the REST meta-architecture.

[1] O ERDDAP (Environmental Research Division's Data Access Program) is a data server that offers a simple and consistent way to download scientific datasets in common file formats and to create graphs and maps. This particular installation of ERDDAP has oceanographic data, such as satellites and buoys.

B. Rest - Representational State Transfer

The REST architecture style was developed as a Web architecture abstract model based on HTTP and URI (Uniform Resource Identifier), having been presented not only as an architecture, but also as a way to evaluate architectures. This idea was proposed by Roy T. Fielding in his PhD thesis, titled "Architectural Styles and the Design of Network-based Software Architectures", published in 2000. Fielding is also the main author of HTTP / 1.1 and co-author of the Internet HTTP and URI standards.

Fielding e Reschke [16] defined HTTP as a stateless application level protocol for distributed, collaborative and hypertext information systems. In this protocol, the format and the negotiation of the data representation allows systems to be build independently of the transfered data.

According to Fielding e Reschke [17], the HTTP protocol methods must have the same semantics when applied to any resource, even though each resource determines by itself whether those semantics are either implemented or authorized. A series of standardized methods that are commonly used in HTTP is described in Table I.

The *Internet Engineering Task Force* (IETF), the main organ to establish standards in the Internet, through its *Request for Comments* (RFC), describes the standards for each Internet protocol. Besides the eight methods described in RFC 7230 [16], the PATCH method was also standardized according to RFC 5261 [18], which describes a patch format for XML documents and in RFC 6902 [19], which describes a similar format for JSON documents [20].

TABLE I
Methods from the HTTP protocol

Method	Description
GET	Get a representation of the resource.
HEAD	Get the headers sent together with the representation of the resource.
POST	Create a new resource based on the supplied representation.
PUT	Replace the state of a resource by the one described in the supplied representation.
PATCH	Modify only some specific informations of the resource.
DELETE	Delete the resource.
CONNECT	Forward some other protocol through its HTTP proxy.
OPTIONS	Discover the HTTP methods to which the resource responds.
TRACE	Debug proxies.

According to Fielding [21] and Richardson, Amundsen and Ruby [20], URI is the simplest and most important element from the Web architecture. There were hypertext systems before HTML and Internet protocols before HTTP, but they did not communicate with each other. The URI has interconnected all those protocols into a single web.

The initial Web architecture defined the URI as documents identifiers. The authors were instructed to define identifiers in terms of the document location in the network. The Web protocols could then be used to retrieve this document. Nevertheless, this definition proved its self not satisfactory because, for instance, of the need to change the identifier whenever the content was changed or even when the URI corresponded to a service instead of a document [21].

With REST, it is possible for the reference to remain unchanged, even though the results received when we access this reference may change over time. Thus, we define a resource with the semantics the author wants to identify, instead of identifying it as the value corresponding to teh semantics when the reference is created. Hence, we leave it to the author to make sure that an identifier chosen for a reference really identifies the intended semantics [21].

URI offers a simple and extensible way to identify a resource. The specification of the URI syntax and semantics is derived from concepts that were first presented by the World Wide Web [22].

In the REST and W3C recommendations, a resource and its URI must be intuitively correlated, keeping a structure and changing in a predictable way. Each URI designs exactly one resource, but one resource can have more than one URI. For instance, if the current software version is 1.3, we can assign two different URIs such as http://www.example.com/releases/1.3.tar and http://www.exa mple.com/releases/latest.tar, that can refer to the same file for a specific time frame [20].

REST offers a set of architectural restrictions that, when applied as a whole, emphasize scalability of the component interactions, generality of the interfaces, implementation independence from middleware components in order to decrease interaction latency, reinforce security and encapsulate legacy systems [23].

The RESTful architecture in an implementation of the concept of the REST meta-architecture: a set of simple directives for the implementation of typical services that can provide to the potential of the Web [20].

V. PROCESSES OF THE OCEANOGRAPHIC NUMERICAL FORECASTING SYSTEMS - SPNO

In this work, the data gathering is done in the ERDDAP database, which is a scientific database kept by the NOAA[2] (National Oceanic and Atmospheric Administration).

The next step, generating the initial field, using data assmiliation techniques, makes the satelitte images create tridimensional fields using the orbital and climate data. This way, we generate, for instance, temperature and salinity tridimensional fields.

The processes are executed with Matlab and Python generate the initial field, which occurs after the orbital and climate data are available.

[2] The NOAA is an organization that belongs to the United States Department of Commerce, which is supported by the United States National Meteorological Services. Its mission is to "understand and forecast climate, oceaninc and coastal change, in order to share this knowledge and information with other people and to conserve and manage the coastal and maritime ecosystems and resources. " [24].

Figure 1 shows the graphical representation of the orbital data extracted in November, 24th, 2017 with the following coordinates: Latitude (-40º e -15º) and Longitude (-50º e -30º), using the ERDDAP system.

The data is a product from the Sea Surface Temperature Analysis (SST) performed by the Jet Propulsion Laboratory (JPL) in the NASA MeaSUREs program. The orbital data was extracted from the Multi-scale Ultra-high Resolution (MUR) SST Analysis fv04.1, global, 0.01°, 2002-present, Daily database. The bar with values 0 to 32 indicate the temperature variation in ºC.

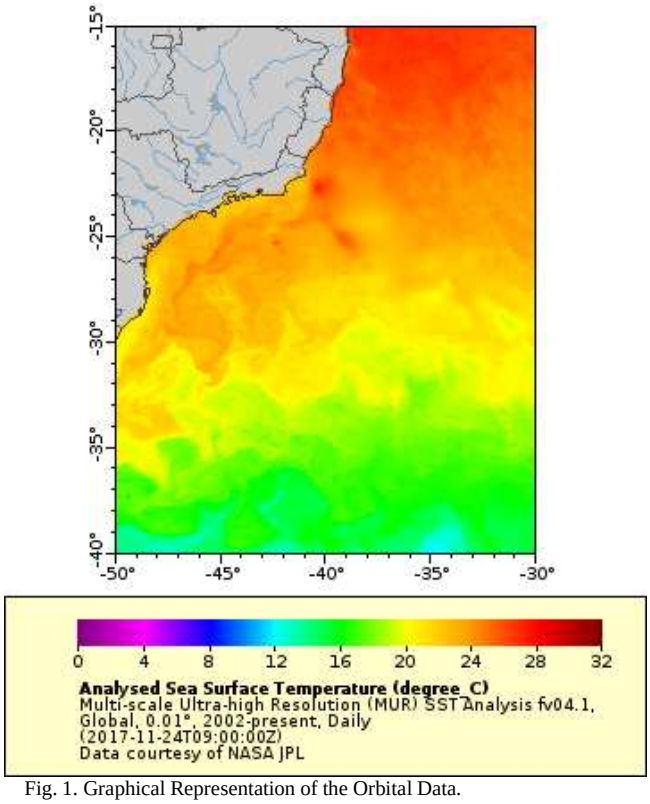

Fig. 1. Graphical Representation of the Orbital Data.

In the process blocks of the Fortran server are the processing steps of the Forecasting System, that is, where the numerical model Regional Ocean Modeling System (ROMS) will be performed in three steps, as shown in Figure 4. Each execution step of the ROMS numerical model works in a specific way.

The first operational model is responsible for smoothing the tridimensional fields based on the initial fields. This module initializes the process and its results are transfered through the new initial field to the next operational model.

The second operational model allows for temperature and salinity forcing fields to remain invariant in time as the velocity changes. It is called Nowcasting (current wheather forecast), which is the result of the process of adjusting a velocity field to the temperature and salinity fields, as shown in Figure 2.

The last module uses the Nowcasting field and evolves in time with the velocities, temperatures and salinities interacting among themselves. This way, we can generate the forecasting result with reliability for three days, as shown in Figure 3.

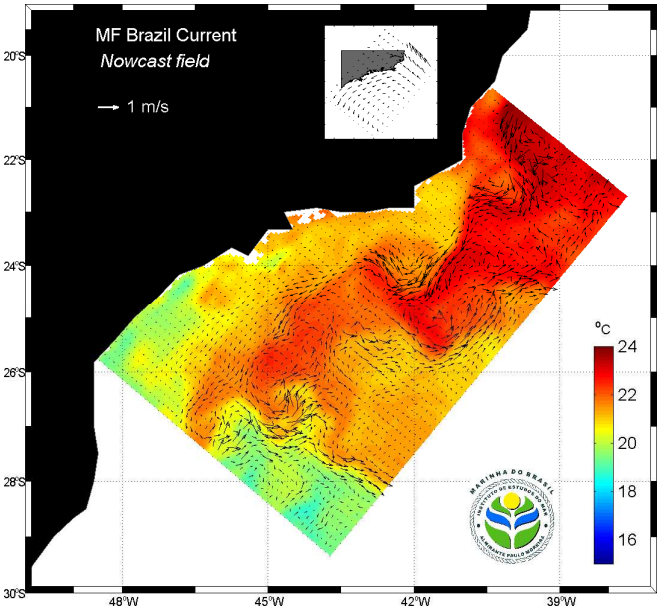

Fig. 2. *Nowcast*: result of the present wheather

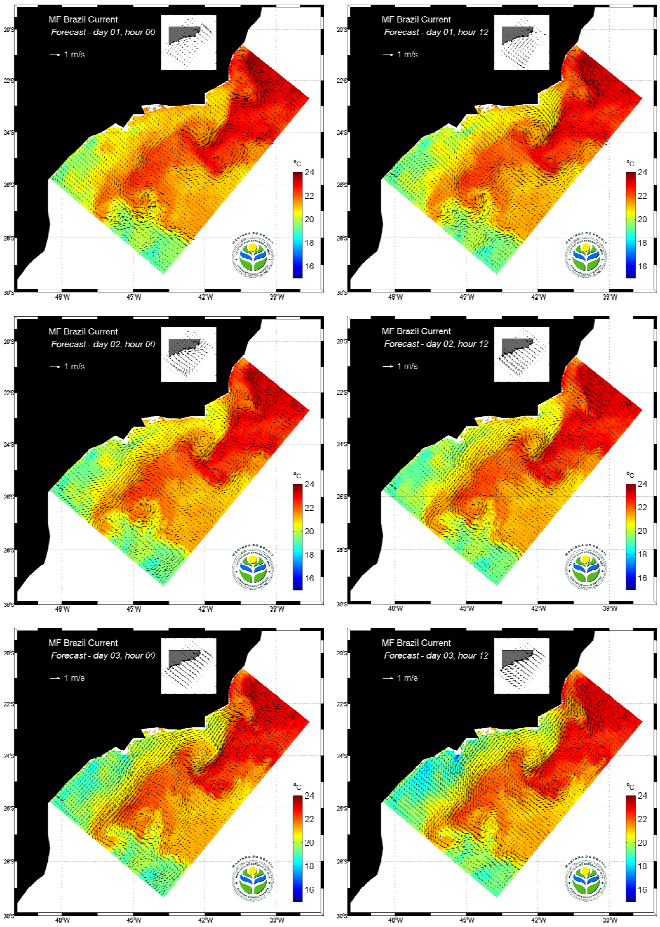

Fig. 3. Resul of the forecasting for three days ahead with 12 hours intervals.

VI. EXPERIMENTAL DESIGN

The expert systems used in the Oceanographic Numerical Forecast process are updated according to the methods used internationally to generate forecasts, but also allow for specifici expert configurations according to the knowledge of techniques or specificities of the region where the studies are performed.

We need flexibility and independence for individual adjusts such as time period, grid size and number of processing cycles in order to improve the results, and even for the replacement of a new module.

This work intends to focus on the simplicity and eficiency of the Web Services application in the presented problem. Hence, we use in each one of the models the Restify[3] implementation, which is a library developed in JavaScript for Node.js, which allows for the construction of Web Services based on the RESTful architecture.

In order to keep the systems independent, the files processed by each system are store in a directory structure, so that each can be accessed locally and remotely, allowing for the whole process to proceed.

The files maipulated by the system are mostly in the NetCDF[4] format, a standard used to access and share matrix oriented scientific data.

Given the file size and the network availability, these files are compressed. As an example, we used the result file from the third step of the numerical model processing, which contains the forecast in the NetCDF format. Figures 2 and 3 are graphucal representation of the results generated from this file. In the evaluation of the compressing, the file shrunk to 54% of its original size, going from 810 Mb to 439 Mb.

The data processing structure does not restrict access within the local network, beibg able to use higher capacity hardware from other institutions. Hence, the compressing prcess is important to increase the file transfer efficiency and for the forecast results to be available in an acceptable time frame.

Table II shows the structure of the Web Services directories and indicates a process flow. The directories that start with "Sis" represent the expert systems and those that start with "Dir" represent the directories taht store files generated by the systems. These directories are made available through Web Services, using the GET and GET:ID requisitions, in which we execute the modules fs.readDir, to read a directory content and fs.readFile to read a file content.

TABLE III
Web Services Directory Structure

	WebService1		**WebService2**		**WebService3**
1	SisDownload				
2	DirDownload	→	DirDownload		
3			SisCampoInicial		
4			DirCampoInicial	→	DirCampoInicial
5					SisROMS
6			DirROMS	←	DirROMS
7			SisPrevisaoIMG		
8	DirPrevisaoIMG	←	DirPrevisaoIMG		

In Figure 4 we present a graphical representation of the process flow. At WebService1, its structure stores the data gathered from external systems and makes the results of the forecasting process available to eternal requisitions.

At the next WebServices , the data are transfered and processed through its routines, such as the generation of the Initial Field, which makes its result available for WebService3 and the image generation, which makes its result available to WebService1. At the WebService3, after the processing performed in the three steps of the ROMS model, the result is made available for both the WebService1 and the WebService3.

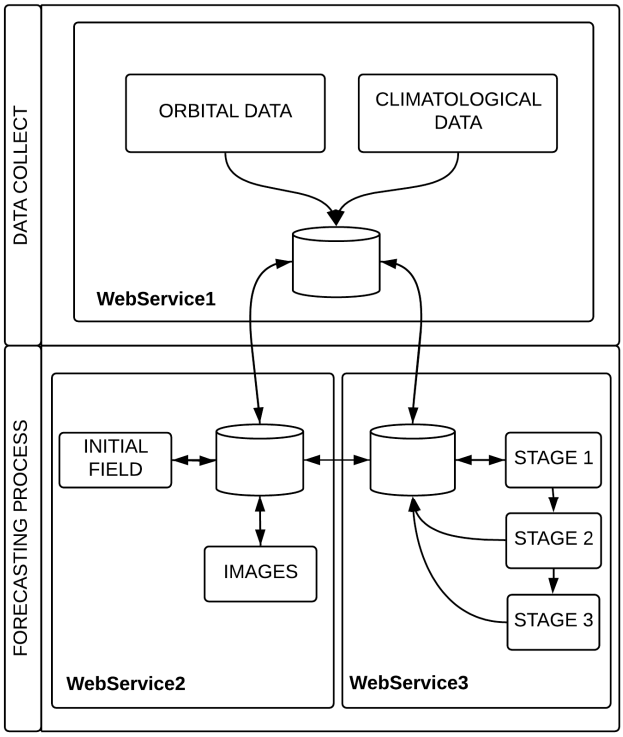

Fig. 4. Components for both the data gathering and the forecasting processes.

[3] Big companies such as Netflix, Napster, Pinterest, NPM and Joyent use the Restify module.

[4] Network Common Data Corm (NetCDF) is a set of software libraries and self describing data formats, machine independent which support the creation, access and sharing of matrix-oriented scientific data.

Each Node.js server, besides the implementation of routes for its services, uses a CRON[5] function to automatize its tasks and the child process module, through the child.process.execSync()[6] function, to generate a shell and execute a command for the execution of the processes. The specific functions for each server are the following:

- The WebService1 routine is responsible for making a GET requisition at the specificied server (ERDDAP) and, using the child.process.execSync() function, crompress the file. It also uses the GET method to perform the query and serach of the WebService2 results;
- The WebService2 routine has two different steps that are executed at different times.The first is responsible for searching the WebService1 files, decompress them and transfer them to the SisCampoInitial folder, with the predefined name. Then, it generates the Initial Field and compress the results into the DirCampoInicial directory. The thirs steps is to download the file to WebService3, decompress it and transfer it to the SisPrevisaoIMG folder with the predefined name. Then it generates the forecasting images and compress the results into the DirPrevisaoIMG folder;
- The WebService3 routine gets the Initial Field file from the WeService2, decompresses it, copies it with the predefined name into the SisROMS folder, executes the steps from the ROMS system and, after the result is generated, the file is compressed and made available at the DirROMS folder.

The functions described above are also presented in the activities diagram shown in Figure 5.

The management system allows us to monitor and control the modules, passing parameters, initiating processes and analysing the execution of the steps.

As shown in Figure 6, all the module communicate with each other. This communication is necessary so that the file transfer through the processing steps does not create barriers and in order to make the proposed solution more effective. Using REST architecture with files in the JSON format, besides allowing us to send and receive with no obstacles from firewalls, given that we work directly in the HTTP protocol, also intends to work with the resources as identified by their URI, allowing parameter passing to the other servers for adjustments in their local modules.

[5] Throught the CRON function it is possible to specify the time interval for which the remote file verification function and the internal processes must be executed.

[6] Child.process.execSync() is a symchronous version of child_process.exec(), which will block the Node.js event loop.

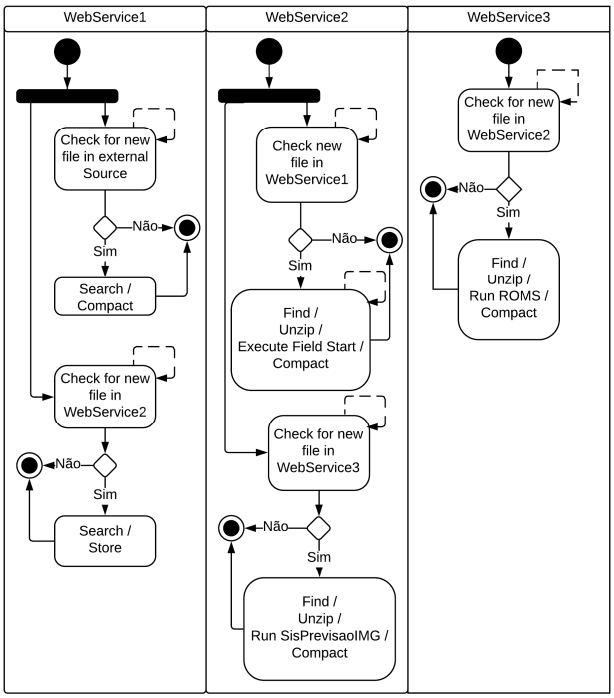

Fig. 5. Activities Diagram

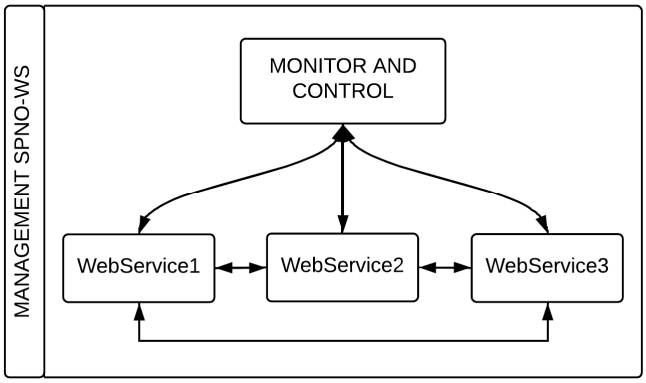

Fig. 6. Communication between the SPNO-WS modules.

Figures 7 and 8 detail the requisitions and the responses from both processes that use REST and a way to transfer files and pass parameters.

As shown in Figure 7, the client performs a query on the available files to the Web Service through the GET method. The Web Service then executes an internal routine that is capable of listing the available files and returning a response in the JSON format. Based on this response, the client identifies through an internal routine if there is any file and executes once again the GET method informing the id of the file that must be copied.

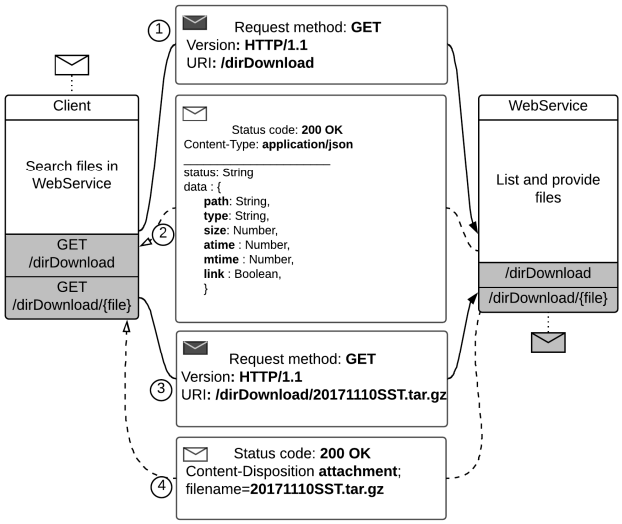

Fig. 7. Example of a REST communication in the file transfer steps.

As shown in Figure8, the client performs a query to the configuration file using the GET method to the Web Service, which executes an internal routine capable of reading the file and returning the current paramenter through a response in the JSON format. Based on this response, the client executes the PUT method, passing the previously read value and the new parameter.

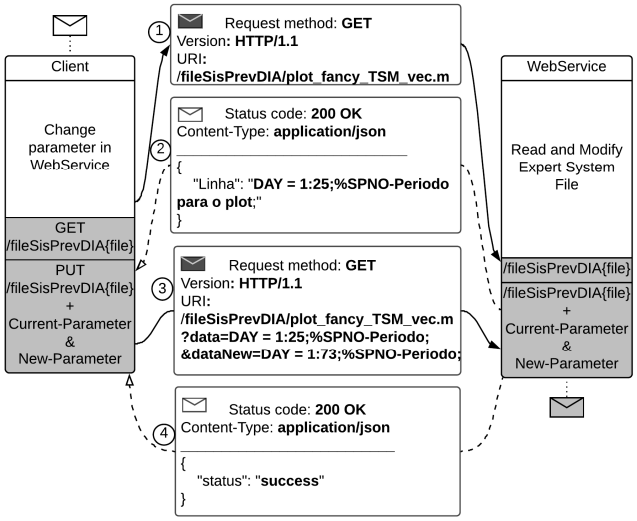

Fig. 8. Example of a REST communication while in the steps of parameter passing for the expert systems.

VII. DISCUSSION OF THE PROCESSING STEPS

The Java Script technology combined with NODE.js, brings simplicity and the possibility to interact with the Operational System in an easy and controlled way, something that was not possible with other languages. This way, the access to expert systems or even legacy systems that do not support recent technologies becomes possible allowing us to use newer technologies.

The RESTful architecture brings clarity and ease to the implementation of the REST concepts, allowing the implementation in many different way, from simple file search operations up to file manipulations that include reading, deletion and remote parameter changes in the expert systems. This way, the existance of several libraries for NODE.js, already mentioned in this paper, contribute significatively to the development agility, due to the great contribution of developpers from the whole Web.

The adaptation of new systems requires an adjustment period. Nevertheless, the modular structure allows for agility for the adjustments to new systems and parameters and also for a clear and objective view of the system. It also allows for the experts to be closer to propose system evolutions in order to contribute to the forecasting results. For instance, it is possible to develop systems that compare forecasting results, identifying by intermediate systems the equivalence of the forecasting at the current day as compared to a previously executed forecast. In thsi caso, the system elaboration is performed by researchers in a totally independent way, so that in the future, it can be added as a new module.

As future work, we intend to implement adjustments in the functioning model, allowing for parameters in the existing legacy routines to be adjusted automatically. For instance, adjust the forecasting window, which nowadays is still performed manually. There are other possible improvements to the model under consideration at this moment.

We would like to stree that this research is based on the internal models and procedures for numerical forecast at the IEAPM and they answer to its specific needs. Hence, in order to use it at another research institute, it is necessary to perform a complete study of all procedure, routines, use stategies and other factors so that we may adapt the developped system to the new reality.

REFERENCES

[1] SOUZA, Ronald Buss de. Oceanografia por satélites. 2. ed. Oficina de Textos, 2009.

[2] RICHARDSON, Lewis Fry. Weather prediction by numerical process Cambridge University Press. Monthly Weather Review, n. February, p. 219, 1922. Disponível em: <https://docs.lib.noaa.gov/rescue/mwr/050/mwr-050-02-0072.pdf>. Acesso em: 25 mar. 2017.

[3] KALYVAS, Christos; KOKKOS, Athanasios; TZOURAMANIS, Theodoros. A Survey of Official Online Sources of High-quality Free-of-charge Geospatial Data for Maritime Geographic Information Systems Applications. Information Systems, v. 65, n. September 2016, p. 36–51, 2016.

[4] FLEMMING, Nicholas Coit. Strategic Planning for Operational Oceanography. Ocean Forecasting. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. p. 1–17. Disponível em: <http://link.springer.com/10.1007/978-3-662-22648-3_1>. Acesso em: 24 mar. 2017.

[5] BAPTISTA, António M. Modern Paradigms for Modeling and Visualisation of Environmental Systems. Encyclopedia of Physical Science & Technology, v. 5, p. 565–581, 2002.

[6] FERREIRA, Hiro Gabriel Cerqueira. Arquitetura de middleware para internet das coisas. 2014. 111 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade de Brasília, Brasília, 2014. Disponível em: <http://repositorio.unb.br/handle/10482/17251>. Acesso em: 20 maio 2017.

[7] ONO, Keiichiro et al. CyREST: Turbocharging Cytoscape Access for External Tools via a RESTful API. F1000Research, 2015. Disponível em: <http://f1000research.com/articles/4-478/v1>. Acesso em: 1 jun. 2017.

[8] POULTER, Andrew John; JOHNSTON, Steven J.; COX, Simon J. Using the MEAN stack to implement a RESTful service for an Internet of Things application. IEEE World Forum on Internet of Things, WF-IoT 2015 - Proceedings, p. 280–285, 2016.

[9] IEEE. Standards Glossary. Disponível em: <https://www.ieee.org/education_careers/education/standards/standards_glossary.html>. Acesso em: 24 mar. 2017.

[10] MCCABE, Francis et al. Reference Model for Service Oriented 1.0. p. 31, 2006. Disponível em: <https://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>. Acesso em: 20 abr. 2017.

[11] PAPAZOGLOU, Mike P.; VAN DEN HEUVEL, Willem Jan. Service oriented architectures: Approaches, technologies and research issues. VLDB Journal, v. 16, n. 3, p. 389–415, 2007. Disponível em: <https://link.springer.com/article/10.1007/s00778-007-0044-3>. Acesso em: 3 maio 2017.

[12] ERL, THOMAS. SOA Principios de Design de Serviços. São Paulo: Pearson Prentice Hall, 2009.

[13] ORT, Ed. Service-Oriented Architecture and Web Services : Concepts, Technologies, and Tools. 2005. Disponível em: <http://www.oracle.com/technetwork/articles/javase/index-139840.html>. Acesso em: 23 abr. 2017.

[14] W3C, World Wide Web Consortium. Web Services Architecture, 2004. Disponível em: < https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>. Acesso em: 25 mar. 2017.

[15] DA SILVA, Ricardo Frenedoso; GONÇALVES, Pablo Rodrigo. Web Services – Uma Análise Comparativa. Revista das Faculdades Integradas Claretianas–No5–janeiro/dezembro de, p. 8, 2012.

[16] FIELDING, Roy Thomas; RESCHKE, J. RFC 7230 - Hypertext Transfer Protocol HTTP/1.1: Message Syntax and Routing. 2014a. Disponível em: <https://tools.ietf.org/pdf/rfc7230.pdf>. Acesso em: 24 maio 2017.

[17] FIELDING, Roy Thomas; RESCHKE, J. RFC 7231 - Hypertext Transfer Protocol HTTP/1.1: Semantics and Content. 2014b. Disponível em: <http://www.rfc-editor.org/info/rfc7231.>. Acesso em: 25 maio 2017.

[18] URPALAINEN, J. RFC 5261 - An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors. RFC, 2008. Disponível em: <https://tools.ietf.org/pdf/rfc5261.pdf>. Acesso em: 11 jun. 2017.

[19] BRYAN, P.; NOTTINGHAM, M. RFC 6902 - JavaScript Object Notation (JSON) Patch. 2013. Disponível em: <http://www.rfc-editor.org/info/rfc6902.>. Acesso em: 11 jun. 2017.

[20] RICHARDSON, Leonard; AMUNDSEN, Mike; RUBY, Sam. RESTful Web APIs: Services for a Changing World. 5. ed. Sebastopol, CA: 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2013.

[21] FIELDING, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. 2000. 162 f. Doctor of Philosophy in Information and Computer Science, University of California, IRVINE, 2000. Disponível em: <http://www.ics.uci.edu/~fielding/pubs/dissertation>. Acesso em: 24 mar. 2017.

[22] BERNERS-LEE, T; FIELDING, R; MASINTER, L. RFC 3986 - Uniform Resource Identifier - URI: Generic Syntax. 2005. Disponível em: <https://tools.ietf.org/pdf/rfc3986.pdf>. Acesso em: 24 maio 2017.

[23] FIELDING, Roy Thomas; TAYLOR, Richard N. Principled design of the modern Web architecture. ACM Transactions on Internet Technology, v. 2, n. 2, p. 115–150, 2002. Disponível em: <https://www.ics.uci.edu/~fielding/pubs/webarch_icse2000.pdf>. Acesso em: 24 maio 2017.

[24] NOAA, Administração Nacional Oceânica e Atmosférica. Sobre. Disponível em: <http://www.noaa.gov/about-our-agency>. Acesso em: 26 mar. 2017.