High Performance Pre-Computing: Prototype Application to a Coastal Flooding Decision Tool

Romain Chailan, Anne Laurent, Frédéric Bouchette, Colin Dumontier, Olivier
Hess, Olivier Lobry, Héloïse Michaud, Sophie Nicoud, Gwladys Toulemonde

HAL Id: lirmm-00736797

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00736797v1

Submitted on 30 Sep 2012

High Performance Pre-Computing: Prototype Application to a Coastal Flooding Decision Tool

Romain Chailan*, Frederic Bouchette†, Colin Dumontier§, Olivier Hess§, Anne Laurent*,
Olivier Lobry†, Heloise Michaud†, Sophie Nicoud*, Gwladys Toulemonde‡
*LIRMM - †OSU/OREME - ‡I3M
University Montpellier 2
France
Email: {romain.chailan,frederic.bouchette,anne.laurent,
olivier.lobry,heloise.michaud,sophie.nicoud,gwladys.toulemonde}@univ-montp2.fr
§SMARTER WATER Montpellier
IBM France
Email: {olivier.hess,cdumontier}@fr.ibm.com

Abstract—After defining the High Performance Pre-Computing —referred as HPPC— concept, the aim of the present study is to develop a prototype whether to approve or not the benefits of this concept. Our application case tries to answer the geophysical issue of coastal flooding. This is an example of an alert system based on the HPPC architecture, thus on pre-computed scenarios. The prototype provides the scientists with an ergonomic and *on-demand* tool allowing the run of scenarios of any implemented numerical models. These runs are available through a web application which submits the corresponding jobs on the remote french public cluster of HPC@LR. In this study we simulate the waves propagation over a Mediterranean grid using the wave model WaveWatch III®. A reference simulation using usual conditions is approximated using the k-NN algorithm over 12, 98 and then 980 pre-computed scenarios. This simple experiment demonstrates how useful the pre-computing of scenarios is for alert systems as far as enough and relevant scenarios are pre-computed. This is the reason why searches continue in each critical points of the HPPC architecture such as the design of experiment, the approximation of the results by meta-models and the research of the closest scenarios in this big data context.

I. Introduction

Floods like in India in july 2009, windstorms like Xynthia in Europe in february 2010, tsunamis like Fukushima events in Japan in march 2011,... Unfortunately each year gets its own disaster(s). To prevent the risk and limit the damages caused by these terrible events, alert systems are built all around the world. In this context the American National Oceanic and Atmospheric Administration (NOAA)'s Tsunamis Warning Centers built a decision tool mixing live tsunami forecast simulations and pre-computed scenarios of seismic propagation [1], [2]. The European Flood Alert System (EFAS) [3] is also a decision tool reporting the probability of having a flood event using a statistical approach [4]. Other techniques are focused on data ubiquity such as the web-based flood forecasting tool presented by [5]. These efforts are made in the common objective of understanding quickly and accurately what the live conditions are and what will be their consequences. In this paper, we are following these approaches but introducing a technique of pre-computing scenarios instead of simulating online -i.e. in live- the time-. Generally, an alert system is based on models which require huge resources and are extremely time consuming. Therefore the major contribution to the area of alert system is to get rid of the dependancies on these constraints to provide a forecast once a crisis arises.

In [6], we established that the HPPC architecture presents the potential to provide the scientists with an ergonomic and generic tool to run any model of their choice as the GenScn tool presented in [7]. The HPPC architecture allows the end-user to perform statistical analyses on previously computed scenarios. The aim of this process is to forecast the outputs of a run without any live simulation. Therefore two main innovative concepts are added: the automation of the choice of scenarios to pre-compute coupled with the use of a web based application negociating with a remote cluster to run the simulations. The development of the prototype in this study is used as a validator of this innovative approach of pre-computed scenarios.

We are focusing our study on the question of the coastal submersion. [8] defines the coastal submersion mechanism as the intrusion of marine salt water into the (usually) dry flood plain, by over-wash of the sand barrier, by digging of storm inlets through the sand barrier or by any similar process that originates in a significant elevation of the mean sea level. The experimentation consists in studying the behavior of the waves such as the wave-height, the wave-length and the wave-period in a configuration representing the Mediterranean sea. Basically a scenario defines a simulation —with a set of parameters— using the WaveWatch III® model [9]. The scenario S_{ref} is set as the reference scenario for the queries studied in this paper. The goal is to approximate the best as possible the result of S_{ref} using only other pre-computed scenarios. The approximation is re-established after having respectively populated the database by 12, 99, and 980 pre-computed scenarios to observe whether the HPPC approach is a good way to proceed or not. Thus, it will justify the need —or not— of a large optimized infrastructure. By validating

the HPPC infrastructure on a little set of experiments, we ensure the continuity and the investments on the project. These investments are concretely more experimentations and searches around the critical points of the architecture. It is translated as developing a performant and relevant automation of the design of experiment, providing the tool with the best clustering algorithms and building meta-models to mimic the simulations.

The other benefit of the prototype is to develop technical skills which can be used in a latter and more industrial scaled solution. Indeed in this prototype, many concepts of abstractions, design patterns and software engineering are intensively used. These concepts make the development and the tool more ergonomic and contribute to the use of the HPPC architecture by the scientists.

The paper is organized as follows. Section II introduces the technical infrastructure built for the HPPC prototype, while Section III details the implementation and the choices made to obtain a maintainable and efficient tool. Finally Section IV explains the experiment and proposes an interpretation of the results.
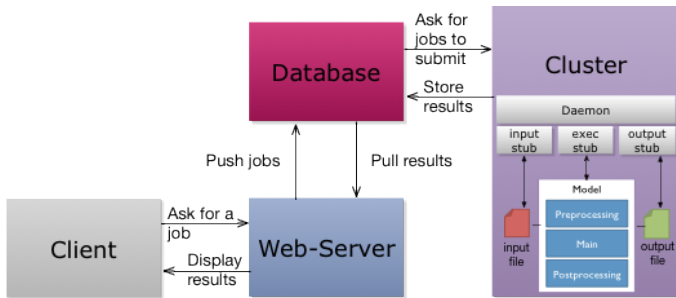
II. TECHNICAL INFRASTRUCTURE

Fig. 1. Technical infrastructure of the HPPC prototype.

The HPPC prototype is based on an architecture Client-Server-Cluster as illustrated by Figure 1. The web server is in charge of the front-end of the application. The user only interacts with the web-server. The web-server pushes jobs to perform into a database and pulls from the database the progress of a job and its results. The cluster is the back-end of the prototype. Its only purpose is to pull the jobs from the database, submit them and push the results —and status changes— back into the database. Section III-A specifies how jobs are submitted and results retrieved by the cluster.

The web-server and the database are stored on the same machine but remotely from the cluster (managed by a dedicated node). The communications between the client and the web-server follow the web standard HTTP protocol. The cluster communicates with the remote database through the JDBC driver.

In this first stage of our development, this configuration has been chosen instead of the web-service implementation in order to be focused on the HPPC concept validation. Indeed, the handling of jobs submissions, the discovering of inputs and the retrieving of outputs can theorically be performed by a web-service technology in a less coupled way. However, it should involve a cascading of different web-services that we were not confident to quickly deploy to validate the HPPC concept. This approach will be considered in future development.

III. PROTOTYPE

The prototype designed is in charge of validating the concept of the HPPC architecture. For the first steps only few modules of the global architecture —but the main ones— have been developed. Figure 2 illustrates the entire workflow embedded. In the next sections the choices of implementation for each module included in the prototype are detailed.

A. Simulation

We developed the prototype keeping a great genericity over the models we used [6]. For instance, it could have been possible to run a financial model as far as this later model is set up on the HPC cluster of the prototype. To make the link with the case study, we can state that a "job" in Figure 1 corresponds to a "a set of parameters" chosen by the client and which is given as input of the corresponding wave model. As illustrated in Figure 1, the cluster is in charge of submitting and running the jobs. Along the processing, the cluster sends back status of jobs to the web server via the database. In order to perform these tasks, a daemon containing three stubs —respectively *input*, *exec* and *output* stub— for each model is launched on the cluster.

- The *input* stub generates the input files corresponding to its model by mapping the input parameter values pulled from the database.
- The *exec* stub runs the pre-processing subroutines, submits the job and runs the post-processing subroutines.
- The *output* stub retrieves the output values to the database by parsing the outputs of the model.

Even if two models have been installed and currently available to run jobs on the HPPC architecture —which is a proof of the application genericity—, we focus this study on the determinist wave propagation model: WaveWatch III® [9]. Our chose model WaveWatch III® is configured to run simulations on HPC platforms with optimal benefits thanks to the MPI library. This particularity represents one of its main advantage for our application. The simulations of wave propagation are made on a Mediterranean grid describing 141×220 computational nodes. Figure 5 is a preview of the computational mesh displayed on Google Earth tool. More details are provided in the section IV-C. The main purpose of the prototype is to validate the HPPC principle. This is why each run of the defined simulation is quite fast: from 9 to 26 minutes running on an Intel bi-processor hexacore machine. These relatively short jobs allow us to analyze them quickly. It is certain that from a geophysical point of view, the results of our simulations are not really relevant. However in case of good results we can extend

this analysis to more complicated configurations and models thanks to the generic implementation. It is also important to highlight that WaveWatch III® has several pre/post-processing subroutines. Therefore embedding this kind of models into the prototype is an additional challenge of this study.

B. Choice

The choice module implemented in the prototype is really simple compared to the one defined in the HPPC architecture [6]. We do not use any feedback loop on the computed scenarios to reveal which parameters contribute the more on the output of the model. This former analysis is done by ourselves and confirmed by advices of experts in the area. At the model definition via the user-interface, each input gets a minimum value, a maximum value and a step. The scenarios to be pre-computed, often referred in this paper as design of experiment, are simply built by a Cartesian product of each possible value for the inputs.

For instance, let us have a model f with two inputs 'a' and 'b'. We design the experiment with the following values:

- For a: $min = 1, max = 2, step = 1 \Rightarrow A = \{1, 2\}$
- For b: $min = 1, max = 3, step = 1 \Rightarrow B = \{1, 2, 3\}$

Therefore the design of experiment is defined by the set: $A \times B = \{\{1,1\}, \{1,2\}, \{1,3\}, \{2,1\}, \{2,2\}, \{2,3\}\}$ and $|A| \times |B| = 6$ scenarios are pre-computed.

In general, the model f takes n inputs $X = (x_1, x_2, \ldots, x_n) \in D_I = I_1 \times I_2 \times I_{\ldots} \times I_n$ and generates m outputs $Y = (y_1, y_2, \ldots, y_m) \in D_O = O_1 \times O_2 \times O_{\ldots} \times O_m$. The model f is an application from D_I to D_O and is defined as:

$$f : D_I \longrightarrow D_O \,;\, X \longrightarrow Y = f(X). \tag{1}$$

An additional remark is that for all i input, $i \in \{1, \ldots, n\}$, the ensemble I_i is defined as: $I_i = \{i_{min}, i_{min} + i_{step}, \ldots, i_{min} + k \times i_{step}, i_{max}\}$, with $k \in \mathbb{N}$ s.t. $i_{min} + (k + 1) \times i_{step} \geq i_{max}$. Thus $\prod_{i=1}^{n} |I_i|$ scenarios are computed.

C. Storage

In the prototype the current storage is a MySQL relational database. The engine innoDB is used to guarantee the respect of the ACID transactional rules. This database has two roles in the application and so hosts two semantic kind of tables. Indeed the database is on the one hand a resource to provide information on jobs to the end-user while on the second hand the system uses the database to display and exchange information relative to the numerical models. Therefore in addition to a) system tables that support the whole HPPC tool from jobs historic to dynamic contents of the web server, b) specific tables are designed to store the models, inputs and outputs. Because the prototype must be generic to host any model, the specific tables are generated dynamically by the system. This automation is realized with the help of an XML file provided for each model which is validated and parsed by the web server.

The system can handle basic types such as integers, decimals, booleans and enumerations. Outputs can be vectors. However

no work has been made on the optimization of this storage and it remains a key point of research for the ongoing work.

D. Interrogation

The interrogation over the pre-computed scenarios already stored is done by the k-NN algorithm [10] considering the inputs of the model. The distance between two inputs is computed using for instance an Euclidian distance given by the following equation:

$$d(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}.$$
$$\text{where } A = (a_1, \ldots, a_n) \in \mathbb{R}^n$$
$$\text{and } B = (b_1, \ldots, b_n) \in \mathbb{R}^n. \tag{2}$$

For instance, lets take the model f previously defined and having two input variables. Lets define two scenarios of f, respectively S_1 and S_2 defined as:

$$S_1 = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \text{ and } S_2 = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}. \tag{3}$$

The distance between S_1 and S_2 is computed with the following formula:

$$d(S_1, S_2) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}. \tag{4}$$

N.B.: Before making the difference, the inputs a_i and b_i are normalized regarding their minimum and maximum values.

Then the answer is displayed immediately, and the results from the k-closest scenarios can be compared through the web interface as illustrated in the screenshot referenced as Figure 3.

E. Monitoring and features

A real work has been made around the monitoring of informations along the development of the prototype. This module allows to check the good behavior of the entire workflow: "What is running?", "What is waiting?", "What is done?". Moreover, we use the software-engineering concept of "contracts" — i.e. at each step corresponds a contract as pictured in Figure 4, from the job submission to the consultation of the results. This technique allows the user to identify from where the issues are raising. This is a real value for a tool like this prototype which interacts a lot with different external components without having any feedback to directly display to the end-user. This last feature will certainly be implemented on the large scaled industrial solution regarding the benefits provided.

F. Littoral Model

Another feature of our prototype is to use a littoral-model [8] which takes outputs of the WaveWatch III® model as inputs to compute accurately the waves for the last hundred meters before reaching the shoreline. We have two reasons to use this model. Firstly the wave propagation is perturbed by

the presence of a closer seabed. Unfortunately the oceanographic models such as WaveWatch III® use to not/badly consider this phenomenon. Secondly the littoral model creates a map which indicates to the end-user where along the coast a submersion is forecasted. This map is built by making the difference between the wave-height computed along the shoreline and the height of the coastline. The littoral-model is an online tool and neither required a lot of resources nor a lot of time to perform the computations. However this model is still under development by the specialists thus no analysis can be performed on it, even if a mock map is currently presented to the end-user to validate the workflow.

IV. Experience

In our experience, we are simulating the wave propagation over a grid defined in the following section. Each simulation represents a 3-days run decomposed into steps of 3 hours, so 24 time series.

A. Inputs

To perform a simulation, the end-user must define the following inputs:

1) Do not vary for each scenario:

- A computational grid. In our case the input grid (named 'MED' and described in Section IV-C) has 141×220 nodes as illustrated in Figure 5.
- The bathymetry of the corresponding zone. In this experiment the bathymetry is provided by the LEGOS -*French Laboratory fulfilling Spatial studies of Geophysics and Oceanography*-.
- We only consider the waves generated by the wind, therefore no initial conditions of the sea water level are used for the simulation.

2) Varying for each scenario:

- The wind conditions between the starting date and the stopping date. These data are provided for each node of the grid. For each time step, we set the wind contribution parallel to the latitude and then the wind contribution parallel to the longitude. For the tests, the contributions are fixed and remains constant during the simulation in order to simplify the addressed problem.

B. Outputs

As outputs of the simulation the end-user obtain the wave-height, the wave-length and the wave-period for each arbitrary defined points given in decimal degrees. In our case we define six points as enumerated in the following:

- Point1 $latitude : 40.40, longitude : 2.86$
- Point2 $latitude : 41.04, longitude : 5.54$
- Point3 $latitude : 42.19, longitude : 7.43$
- Point4 $latitude : 41.09, longitude : 10.95$
- Point5 $latitude : 38.92, longitude : 11.37$
- Point6 $latitude : 38.08, longitude : 7.47$

We establish statistical analyses on the outputs point by point.

C. Area studied

The geographical area studied is delimited by the following fourth points given in decimal degrees:

- SouthWest $latitude : 31.00, longitude : -5.6$
- SouthEast $latitude : 31.00, longitude : 17.6$
- NorthEast $latitude : 45.00, longitude : 17.6$
- NorthWest $latitude : 45.00, longitude : -5.6$

This zone is referenced in this paper as the 'MED' grid illustrated in Figure 5. It corresponds to a large part of the mediterranean sea bordered with the shorelines of North-Africa and South of Western-Europe.

The mesh is clearly not accurate enough for such an area if we would like to provide a real alert system tool for coastal submersion. However the mesh satisfies our needs to validate our approach of pre-computed scenarios by providing a valid simulation.

D. Results

In this study we firstly define the design of experiments on the inputs V_x and V_y: respectively the contribution of the wind parallel to the latitude and the contribution of the wind parallel to the longitude. These inputs are both given in meter per second and set on the ranges [-20;20] for V_x and [-25;25] for V_y. We query the system after respectively 12, 99 and 980 pre-computed scenarios but excluding the S_{ref} scenario. Following the query step, the system provides the 4-closest scenarios —see III-D. We arbitrary set the value of k to 4 in the k-NN algorithm. This choice is based on the observation of several experiments which show that neither a greater nor a lower value of k does really improve the results. Further experimentations with a variable value of k will be the object of a future work. Then the system gives an approximation of the results by making a weighted average on the results of these selected scenarios. For each scenario, the weight is defined as the inverse of the distance to the scenario S_{ref}. Finally we compare the mean and the max of the differences between the six output points and the reference scenario S_{ref} —more details in Section IV-D2. The comparison is given for each output variables: the wave-height in meter named H, the wave-length in meter named L and the wave-period in second named T.

The following tables summarize the experiment:

1) S_{ref} results: Hereafter are presented results of the reference scenario simulation S_{ref}. For this scenario simulated, the contribution of the wind parallel to the latitude is set to $V_x = 7.87 \ m.s^{-1}$ and the contribution of the wind parallel to the longitude is set to $V_y = -9.31 \ m.s^{-1}$.

Variable	Pt1	Pt2	Pt3	Pt4	Pt5	Pt6
$H(m)$	2.28	2.63	2.96	2.04	2.06	2.82
$L(m)$	70.6	80.0	78.3	56.0	54.7	79.9
$T(s)$	6.47	6.94	6.88	5.81	5.69	6.90

TABLE I
RESULTS FOR THE SIMULATION OF THE SCENARIO S_{ref}

2) Details: In the following tables we defined the mean relative error and the max relative error for each variable: the wave-height, the wave-length and the wave-period. Lets have a closer look on the definition of these values by taking for instance the definition of the wave-height mean relative error by Equation 5 and the wave-height maximum relative error by Equation 6:

The wave-height mean relative error is defined as:

$$\overline{\Delta H} = \frac{1}{6} \sum_{p=1}^{6} \frac{|H_p - H_p ref|}{H_p ref}. \qquad (5)$$

where p is the output point number.

When we analyze the $\overline{\Delta H}$ column in the results tables, we have to note that:

- If for the row "Query" the value is close to 0, thus the approximation of S_{ref} by the model is "on average" good for all the points.
- If the value is too high, thus the approximation of the model is bad. For instance a value of 85 corresponds to a mean relative error of 85% regarding the S_{ref} scenario, which is a bad approximation.

The wave-height maximum relative error is defined as:

$$\Delta H_{max} = \max_{p} \frac{|H_p - H_p ref|}{H_p ref}. \qquad (6)$$

where p is the output point number.

When we analyze the ΔH_{max} column, we have to consider that:

- If for the row "Query" the value is close to 0, thus the approximation of S_{ref} by the model is good for all the points. Indeed every points are well approximated.
- If the value is too high, thus the approximation of the model is bad for at least one point. For instance a value of 85 means that the error of the approximated points are between 0 and 85% of the S_{ref} corresponding value, with at least one error equal to 85% which is too important to consider the results as good ones.

In the same way we define those values for the wave-length and wave-period variables. It provides the figures reported in the following tables, grouped in two kind of rows:

- The "Query" row is the results of the approximation made by the system using the weighted average over the 4-closest scenarios to the reference scenario S_{ref}.
- The K_i rows correspond to each i-closest scenario results and reported for information.

3) For 12 pre-computed scenarios: The design of experiments is defined on the range $[-20; 20] \times [-25; 25]$ with both steps of 20. Using a design of experiments containing only 12 scenarios equally distributed over the inputs, we observe in Table II that the approximation of the result is far from the one expected. Indeed the mean relative errors for the wave-height, the wave-length and the wave-period are respectively 149%, 126% and 40%. We can also remark that the maximum relative error point by point are close to the mean relative error,

Scenario	Inputs		Outputs (normalized to S_{ref}, in %)					
[distance]	Vx	Vy	$\overline{\Delta H}$	ΔH_{max}	$\overline{\Delta L}$	ΔL_{max}	$\overline{\Delta T}$	ΔT_{max}
Query	**7.87**	**-9.31**	**149**	**185**	**126**	**152**	**40**	**47**
K_1[0.215]	0	-5	75	86	55	70	25	44
K_2[0.315]	20	-5	223	358	214	316	79	104
K_3[0.370]	0	-25	288	455	265	385	93	125
K_4[0.436]	20	-25	371	431	236	280	85	94

TABLE II
RESULTS FOR A QUERY OVER 12 PRE-COMPUTED SCENARIOS

which states that the system badly approximates the result at the same level for each point.

4) For 99 pre-computed scenarios: The design of experiments is defined on the range $[-20; 20] \times [-25; 25]$ with both steps of 5. In the case of 99 scenarios, the approximation is

Scenario	Inputs		Outputs (normalized to S_{ref}, in %)					
[distance]	Vx	Vy	$\overline{\Delta H}$	ΔH_{max}	$\overline{\Delta L}$	ΔL_{max}	$\overline{\Delta P}$	ΔP_{max}
Query	**7.87**	**-9.31**	**18**	**25**	**18**	**30**	**6**	**10**
K_1[0.055]	10	-10	30	46	22	29	11	15
K_2[0.073]	5	-10	105	146	89	113	38	45
K_3[0.101]	10	-5	16	31	18	40	8	18
K_4[0.112]	5	-5	66	83	57	69	36	42

TABLE III
RESULTS FOR A QUERY OVER 99 PRE-COMPUTED SCENARIOS

more relevant as illustrated in Table III. The approximation still contains mean relative error of 18%, 18% and 6% for respectively the wave-height, the wave-length and the wave-period. But globally the accuracy of the results increases considerably and thus the confidence given by this statistical approximation is more important.

5) For 980 pre-computed scenarios: The design of experiment is defined on the range $[-20; 20] \times [-25; 25]$ with both steps of 1.5. These last tests are made once the database is pop-

Scenario	Inputs		Outputs (normalized to S_{ref}, in %)					
[distance]	Vx	Vy	$\overline{\Delta H}$	ΔH_{max}	$\overline{\Delta L}$	ΔL_{max}	$\overline{\Delta P}$	ΔP_{max}
Query	**7.87**	**-9.31**	**1**	**1**	**2**	**3**	**1**	**1**
K_1[0.021]	8.5	-10	14	18	11	13	5	6
K_2[0.023]	8.5	-8.5	9	17	6	14	3	7
K_3[0.026]	7	-10	9	16	11	16	5	7
K_4[0.027]	7	-8.5	17	20	13	16	7	8

TABLE IV
RESULTS FOR A QUERY OVER 980 PRE-COMPUTED SCENARIOS

ulated of 980 pre-computed scenarios. In that case we observe in Table IV a very good approximation of the results having an approximation where the maximum relative errors never reach more than 3% whatever the output variable considered. These results can be explained because the distances are this time really close to the scenario queried. Therefore the confidence in the result is very important.

E. Interpretation

Regarding the results of the experiments, the main difficulty of the HPPC architecture remains in the process of pre-computing scenarios close to the scenario queried. It is

obvious that the larger the number of scenarios, the larger we probably get an accurate result. But this is not enough. The scenarios computed have to be "close" to the query. If like in real situation we do not have any clue of what will be the queries, those principles are dependent. Two perspectives are opened: a) trying to understand which variance of each parameter contributes the more on the variance of the outputs — i.e. the sensitivity analysis — in order to optimize the design of experiments. This will contribute to get scenarios representing any queries accurately. b) considering the similar and regular queries of end-users to populate the database with these scenarios. In other words the more a "class" of scenario is queried, the more the design of experiments has to run scenarios contained into this class. Because the end-product is an alert decisional tool, the system should also consider the extreme values which correspond to the unusual weather conditions at the origin of the natural disasters.

V. CONCLUSION

In this paper, we present the HPPC prototype that was developed and we set an experiment to validate the HPPC architecture. More specifically, we show how the concept of pre-computed scenarios can be used for alert system tools. A lot of works have been carried out during the development of the HPPC prototype and many computer sciences techniques and concepts embedded will be reused for a more industrial scaled solution. Behind technical skills, the prototype is complete enough to obtain good results on a specific academic configuration. This experiment validates the HPPC principle and allows us to keep on developing the general HPPC architecture: on the one hand, every module of the general HPPC architecture has to be implemented and improved. It concerns the meta-model creation, the sensitivity analysis, the optimization of scenarios storage, the way of aggregating the k-closest scenarios. On the other hand, the global technical architecture is going to be modified by using web-services, adding a cloud layer to perform runs within the technique of cloud bursting and finally keep developing the features embedded to provide the end-user with a very ergonomic and efficient decision tool.

REFERENCES

[1] E. Gica, V. V. Titov, C. D. Chamberlin, and J. C. Newman, "Development of the forecast propagation database for NOAA's short-term inundation forecast for tsunamis (SIFT)," 2008.

[2] ——, "Development of the forecast propagation database for NOAA's short-term inundation forecast for tsunamis (SIFT)," 2008.

[3] J. Thielen, J. Bartholmes, M. Ramos, and A. D. Roo, "The European Flood Alert System – Part 1 : Concept and development," *Environment*, no. December 2007, pp. 257–287, 2008.

[4] J. C. Bartholmes, J. Thielen, M. H. Ramos, and S. Gentilini, "The European Flood Alert System EFAS – Part 2 : Statistical skill assessment of probabilistic and deterministic operational forecasts," *Environment*, no. December 2007, pp. 289–322, 2008.

[5] X.-Y. Li, K. Chau, C.-T. Cheng, and Y. Li, "A Web-based flood forecasting system for Shuangpai region," *Advances in Engineering Software*, vol. 37, no. 3, pp. 146–158, Mar. 2006.

[6] R. Chailan, F. Bouchette, C. Dumontier, O. Hess, A. Laurent, O. Lobry, S. Nicoud, and G. Toulemonde, "High Performance Pre-Computing: Challenges and Directions of an innovative architecture mixing HPC and BI," 2012.

[7] J. L. Kittle, A. M. Lumb, P. R. Hummel, P. B. Duda, and M. H. Gray, "A Tool for the Generation and Analysis of Model Simulation Scenarios for Watersheds (GenScn)," 1998.

[8] F. Bouchette, F. Sabatier, and G. Sylaïos, "SUBDUNE tool : quasi-explicit formulation of the water level along the shoreline," *Paralia*, 2012.

[9] H. L. Tolman, "User manual and system documentation of WAVE-WATCH III," 2009.

[10] S. Arya, D. M. Mount, N. S. Netanyahu, and A. Y. Wu, "An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions," *Science*, vol. 1, no. 212, 1994.
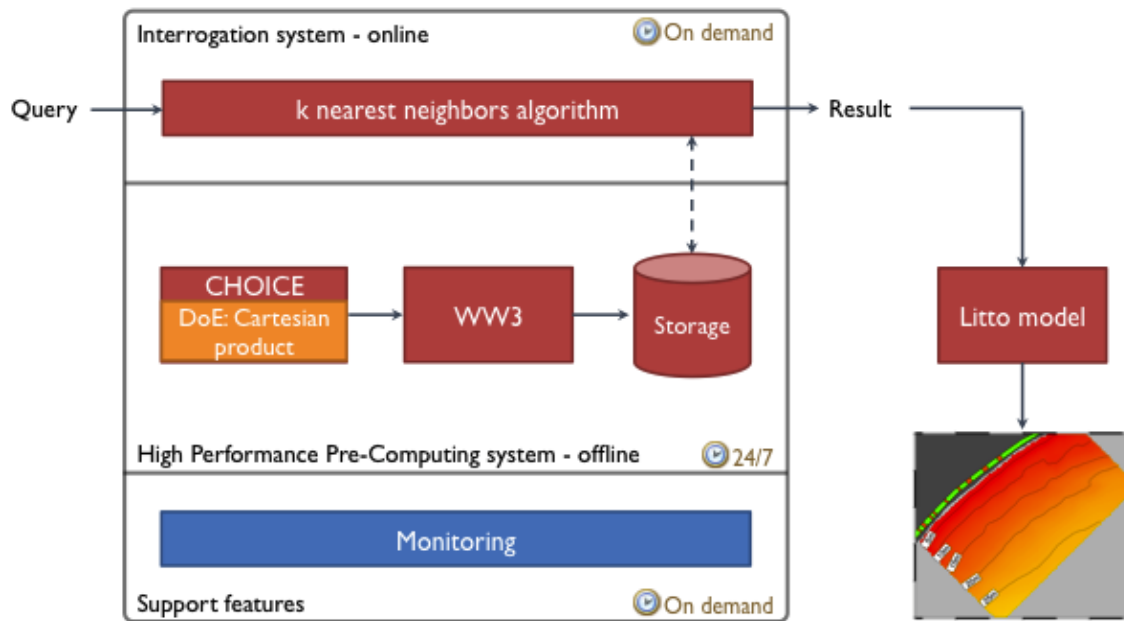
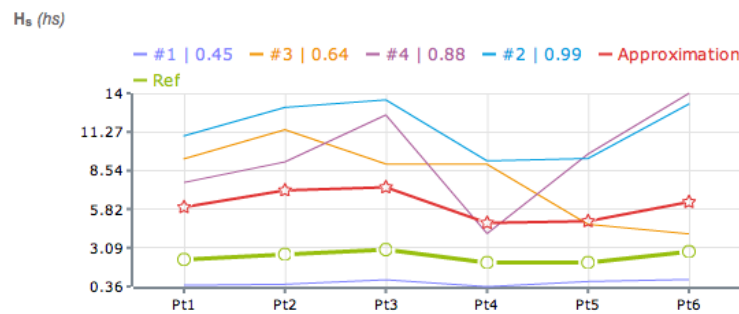Fig. 2. HPPC prototype architecture.

Fig. 3. Screenshot of the web interface: the dashboard sums up to the end-user the results of the 4-closest pre-computed scenarios from the query. Every job is identified by its 'id' number and its distance to the query. Moreover it gives an approximation by making the weighted average of these results. The weight corresponds to the inverse of the distance. We specifically added the reference line to compare with the real result of the simulation queried.

Fig. 4. Screenshot of the web interface: monitoring the evolution of the jobs by consulting the contracts.

Fig. 5. Representation of the structured computational grid (141×220 nodes) named *'MED'* and output-points analyzed, displayed on Google Earth.