Supporting Smart System applications in Scientific Gateway environment

Krisztián Karóczkai[1], Anna Bánáti[3], Péter Kacsuk[1,2], Miklós Kozlovszky[3]
[1] MTA SZTAKI, H-1518 Budapest, Pf. 63., Hungary
[2] University of Westminster, 115 New Cavendish Street, London W1W 6UW
[3] Obuda University, John von Neumann Faculty of Informatics, Biotech Lab
Bécsi str. 96/b., H-1034, Budapest, Hungary
{krisztian.karoczkai,peter.kacsuk}@sztaki.mta.hu
{banati.anna,kozlovszky.miklos}@nik.uni-obuda.hu

***Abstract*—Smart system application are gaining significant attention especially concerning health monitoring researches. Data captured from sensors are arriving continuously and for an efficient handling of this large volume of data stream processing would give the ideal solution [6]. Stream processing means that without intermediate storage the data should be processed as they are coming from the sensors. Storing and downloading data is a time-consuming mechanism, and during our investigation we have proved that data downloading time has significant impact on the overall execution of our health monitoring smart application. We have investigated the most common science gateway solution to examine the path that data takes from the outer storage to the executor node or the infrastructure and we have defined an execution model for smart system applications where the data movement is minimal.**

I. INTRODUCTION

Smart systems are targeted to collect data from sensors positioned into the observed process, and from the collected data to analyze, operate and control the system in order to gain better efficiency. These systems incorporate functions of sensing, actuation, and control in order to describe and analyze a situation, and make decisions based on the available data in a predictive or adaptive manner, thereby performing smart actions. Concerning smart applications the size of the data in most cases are less than a kilobyte, but taking into account the sampling frequency (which can be 600 Hz/source of the digital signal) and the number of the sources which can range up to more thousands, than we face a real-time big data problem. The sensorization of real world processes and most relevantly healthcare applications lead to novel requirements of real-time data analyses. At the beginning of the wide-spread use of smart systems the main issue was the storage of data, but nowadays mainly the real time low latency processing of the high-volume data faces us big challenge. For the execution of compute intensive applications different kind of DCIs (Distributed Computing Infrastructure), clusters, grids and supercomputers are used. The aim of our research is to develop an execution model that can effectively support the data analyses in smart system applications irrespectively from the selected DCI.

II. RELATED WORK

In SMART systems during a short period of time high volume of data can be collected which must be stored and processed. In most cases these data are not the results of real measurements but the results of some computations on measurement data. The later they are processed the longer they should be stored. The ideal solution would be when the data coming from the sensors could be processed immediately and continuously. Classic procession of high volume data result in very bad performance; they require new approach [9] and also traditional SQL queries does not scale well, i.e.: with an increasing data set the SQL queries take more and more time[8] to be processed.

The medical diagnosis requires more than just simple collect and visualize real-time sensor data. The sensor data should be pre-processed (data filtering, cleaning and analysis of basic sensor data patterns) right before the medical examiner's manual analysis. This pre-processing and analyses is a calculation intensive method that occurs mainly in the background, and it is by definition a complex task [11]. Different sensor types are producing different amount and type of data, and the way of processing should be adapted to their requirements. The processing of health care smart system's generated data is a special Big Data problem both for engineers and for data processing systems [12]. The elementary data size can vary from the very small size such as a few bytes to a few thousands of bytes. If we look at the number of already available data sources and their data sending frequencies, and the intensity of measurements of potential data sources, it turns out, that even simple health care monitoring tasks (e.g.: measure pulse, ECG, or SpO2) require more processing resources than what conventional systems can effectively handle.

III. HOW DO SMART SYSTEMS WORK AND PROSESS DATA

Concerning SMART system applications the sensors positioned at the appropriate places are continuously and automatically transmitting data. Most of these data are not bigger in size then a kilobyte but in a very short time interval these sensors can transmit large amount dataset.

This dataset which consists of small pieces of data should be processed by several short running-time applications and the results should be accessible as soon as possible. In other words continuous data processing is required.

We have investigated some common computing middle-layer solutions how they can cope with these challenges.

Currently the health monitoring researches have gained significant attention concerning SMART Systems applications. The efficient and general processing of such data would result a great leap forward not just from a medical point of view but also from an information technology perspective.

We have investigated how big is the dataset that we have to process in case of smart systems. Our investigation is part of a health monitoring research. Within its frames we wanted to analyze ECG curves of humans to determine the number of the pulse during a one minute long time interval. For a similar test during one football match the 5 channel ECG curve at 600 Hz sampling rate of all football players on the field would mean 5*600*60*90*22=356.400.000 data to be processed. When these data were handled as a 4 byte integer value than we would need to store 1.425.600.000 Byte data/match. In the CERN datacenter between the local servers 10GB data can be transmitted in one second. Such a high volume of data would be generated on 3156 concurrently monitoring football math. Also in CERN there are more than millions of applications executed day by day to process the collected data. Such a high volume of data would be generated during 505 football match concerning the pulse of the football players.

A. Development and of a Pilot Application

During the health monitoring research in the Óbuda University Biotech Lab we fixed 4.415.100 ECG-data (with a sampling rate of 300Hz). Within the frames of this research program we wanted to define the pulse values on the basis of continuous ECG data. The pulse is the number of hearth-beats occurring in a single minute; in other words it is the number of the so called R peaks detected in a one minute long interval of the continuously arriving ECG curve. The pulse values must be determined in every second according to the data captured in the previous one minute.

As a starting point we simplified the model of our pulse defining smart system application. The pulse counter application (App) gets the ECG signal as its input, then it counts the R peaks and gives back the result as an output Fig. 1 The real problem is somewhat more complicated: The processing of the data is not carried out at the generation point and the results are not used at the calculation point. So the input data and the calculated results must be delivered to the appropriate place. Moreover the input data stream must be transformed in a form that can be processed.

Fig. 2 shows the typical structure of Smart systems application.

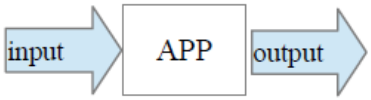

Figure 1. The theoretical structure of the pulse counter application

In order to provide a suitable support for processing, it is inevitable to characterize the behavior of these types of applications. We defined variates for the execution of the complete task and for the different parts of it.

The execution time of the complete program (Xapp) consists of four parts:

- data delivering from the central database to the computing infrastructure (Xget),
- conversion of the received data to a processable form (Xjson)
- determining the pulse (Xpulse),
- restoring the results to the central database (Xput).

During our research work we have investigated the Xapp, Xget, Xjson, Xpulse, Xput variates and the correlation between them.

In the experience we have processed all the available ECG data, which structure is completely identical with data captured by any other ECG measurement. The processing algorithm scans the input data and searches for a specified sample (R peak). The program in all cases verifies the whole data stream so its behavior does not change decisively depending on the content of the data.

The Time Intervals needed for the different ECG processing program parts are handled as independent experiments. The execution of the program may end during any particular interval. For this reason the variates are all considered to be continuous variates.

We determined using F-test whether there is a relationship between two normally distributed statistical multitudes. The critical value – with 1 per cent mistake – which belongs to the size and precision of our experiment is F=1.55957690 (tabular data).

We have examined whether the following null-hypothesis is fulfilled:

$$H0\text{-}_{get} = D(X_{get})=D(X_{app}). \quad (1)$$

$F_{get}=s^2_{app}/s^2_{get}$ =1,479 since $F_{get}<F$, we can declare with 99 per cent certainty, that there is a correlation between the deviation of the download time of the input data and the deviation of the execution time of our application.

In a similar way we have examined the fulfillment of the appropriate null-hypothesizes according to the different time intervals needed for other parts of the application:

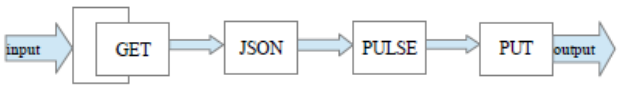

Figure 2. The real structure of the pulse counter application

TABLE I.
EXPECTED VALUE AND VARIANCE OF EXAMINED RANDOM VARIABLES

Random variable	Expected value (ms)	Variance (ms)
X_{app}	1549,96	217,5
X_{get}	871,74	178,84
X_{json}	157,45	24,46
X_{pulse}	381.44	68,91
X_{put}	139,32	24,01

$$H0\text{-}_{json} = D(X_{json})=D(X_{app}) \quad (2)$$

$F_{json=}s^2_{app}/s^2_{json}=1,479$ since $F_{json}>F$, we can declare with 99 per cent certainty, that there is no correlation between the deviation of conversion time and the deviation of the execution time.

$$H0\text{-}_{pulse} = D(X_{pulse})=D(X_{app}) \quad (3)$$

$F_{pulse=}s^2_{app}/s^2_{pulse}=1,479 =1,479$ since $F_{pulse}>F$, we can declare with 99 per cent certainty, that there is no correlation between the deviation of the pulse defining time interval and the deviation of the execution time.

$$H0\text{-}_{put} = D(X_{put})=D(X_{app}) \quad (4)$$

$F_{put=}s^2_{app}/s^2_{put}=1,479$ since $F_{put}>F$, we can declare with 99 per cent certainty, that there is no correlation between the deviation of the upload time and the deviation of the execution time.

During the experiment I was able to verify Stonebraker's statement that download time is a critical factor concerning Big Data[6].

This information is very important because on the basis of this result we have to minimize the data movement.

B. Scientific Gateway configurations for SMART applications

We have investigated the data handling techniques of one of the most prevalent Scientific Gateway solutions, the WS-Pgrade/gUSE/DCI-Bridge system which was developed by the Laboratory of Parallel and Distributed Systems of MTA SZTAKI. The Grid User Support Environment (gUSE) is basically a virtualization environment providing large set of high-level DCI services (including workflow manager, storage, broker, etc.) by which interoperation among classical service and desktop grids, clouds and clusters, unique web services and user communities can be achieved in a scalable way. gUSE has a graphical user interface, which is called WS-PGRADE. gUSE is implemented as a set of Web services that bind together in flexible ways on demand to deliver user services and provide access to various Distributed

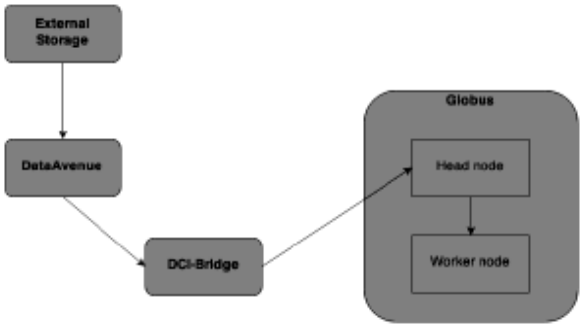

Figure 3. Data transfer with DataAvenue into Globus

Computing Infrastructure (DCI). WS-PGRADE hides the communication protocols and sequences behind JSR286 compliant (Liferay compatible) portlets and uses the client APIs of gUSE services to turn user requests into sequences of gUSE specific Web service calls. End users can access WS-PGRADE via Web browsers. WS-PGRADE/gUSE is used worldwide by many scientific communities, and numerous e-Science gateways based on gUSE. In our performance test we have used the WS-PGRADE/gUSE workflow management system to create workflows and submit the jobs to the DCI Bridge [4][13]. DCI Bridge is a web application (service) that provides standard access to the distributed computing infrastructures (DCIs) like grids (gLite, Globus, Unicore, ARC), clusters (PBS/LSF, XID, SGE, MOAB), clouds (EC2, Fedcloud) and service based computational resources by implementing the specification of the OGSA Basic Execution Service 1.0. The DCI Bridge web application creates a transparent layer between the users (workflow systems) and the DCI systems. The user can submit jobs to the various DCI systems using the OGSA Basic Execution Service (BES) interface. As a result, the users do not have to learn the access protocol of the various DCI systems since they are hidden behind the BES interface.

Based on the analysis of our algorithm it became emerged for us that the execution of our Smart System application is input transfer sensitive so ideal solution is based on minimal data movement. From the above mentioned computing infrastructures and their related execution methods and environments we have searched and for one that use minimal movement of the data.
gUse can store the tasks on its inner local store, on the inner store of the computing middleware or on an outside store that can be reached via the DataAvenue service. The DataAvenue service gives solutions for easy data transfer between various computing infrastructure storage resources.

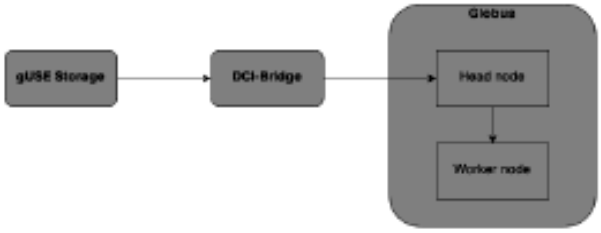

Figure 4. Data transfer with gUSE storage into Globus

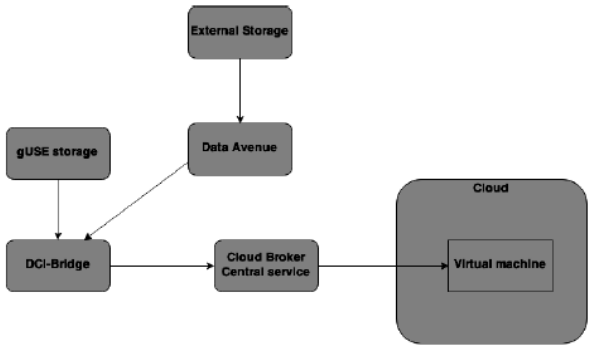

Figure 5. Data transfer with gUSE storage and DataAvnue into Cloud

When an input file of an input port of a job is to be accessed via DataAvenue service, then before running the job the DCI bridge sends this request to the DataAvenue service. As a response the DCI bridge gets the specified file and saves it to its local file system. From there it is copied to the head node of the Globus Site (fig. 3). The scheduler of the Globus system determines which worker node is responsible for the execution and copies the necessary files to the file system of the executor computer.

When we use the gUse inner local store, than the files from there at first are copied to the native local storage of the DCI-Bridge and then it is transported to the used computational middle layer (fig. 4).

From a file transfer perspective PBS/LSF batches operate very similar to the Globus system. With the use of GLite middleware the path of the data files will be longer because the DCI-Bridge transfers the files to the storage of the central GLite service and from there after the selection of the appropriate node the same path is taken as in the case of Globus systems (fig. 5).

There exist a solution when the files are stored inside the gLite system and from there a little wrapper travels and downloads the files directly from the gLite or the GUse storage.

The CloudBroker platform is a service of the Cloud Broker Inc., with the help of which more EC2 compatible cloud services can be accessed. Arbitrary applications can be executed on the virtual machines created by this platform. The files are delivered to the virtual machines through their own swiss center.

Concerning IaaS cloud based solutions in gUse environments direct cloud access provides us with real alternative. In this case the cloud resources can be reached directly with EC2 and FedCloud protocols. From data delivering perspective it is very similar to the direct file availability with gLite, but in this case instead of the wrapper script the JSDL JOB descriptor is delivered to the place of execution. This solution can be further optimized, if the Data Avenue service is implemented into the virtual machine (fig. 6).

IV. CONCLUSION AND FUTURE WORK

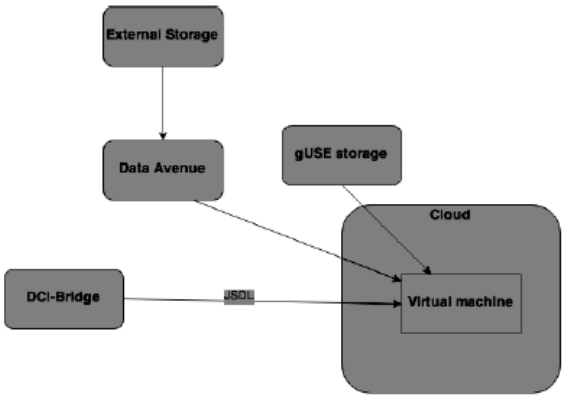

Figure 6. Data transfer with DataAvenue into Globus

We have investigated different SMART applications and defined an execution model to describe them. Within the framework of a health monitoring project we have analyzed the dependency between the time needed by the different part processes and the overall execution time. We have declared that from a processing point of view the data delivery is a critical factor. Thus concerning smart applications we have to try to minimize the inner data delivery with different scientific gateway and computational middleware environments. Using the WS-Pgrade/gUSE/DCI-Bridge Scientific gateway solutions which were developed by the Laboratory of Parallel and Distributed Systems of MTA SZTAKI we have investigated the data path in case of different computational middlewares. According to our experiments we found that for smart systems the direct cloud access provides ideal solution. To continue our research work we would like to investigate other scienific gateway solutions as well.

ACKNOWLEDGMENT

This work was supported by EU project SCI-BUS (SCIentific gateway Based User Support). The SCI-BUS project aims to ease the life of e-Scientists by creating a new science gateway customization methodology based on the generic-purpose gUSE/WS-PGRADE portal family

REFERENCES

[1] P. Kacsuk: P-GRADE portal family for Grid infrastructures, Concurrency and Computation: Practice and Experience journal, Volume: 23, Issue: 3, 2012, pp. 235-245

[2] Peter Kacsuk, Zoltan Farkas, Miklos Kozlovszky, Gabor Hermann, Akos Balasko, Krisztian Karoczkai and Istvan Marton: WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities, Journal of Grid Computing, Vol. 9, No. 4, pp 479-499, 2012

[3] Tamas Kiss, Pamela Greenwell, Hans Heindl, Gabor Terstyanszky and Noam Weingarten Parameter Sweep Workflows for Modelling Carbohydrate Recognition Journal of Grid Computing, volume 8, number 4, pp 587-601, 2010 (SCI-BUS)

[4] Kozlovszky M, Karoczkai K, Marton I, Balasko A, Marosi A, Kacsuk P: Enabling generic distributed computing infrastructure compatibility for workflow management systems, COMPUTER SCIENCE JOURNAL 13:(3) pp. 61-78. (2012)

[5] P Kacsuk, Z. Farkas, M. Kozlovszky, G. Hermann, A. Balasko, K Karoczkai, I Marton: WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities, JOURNAL OF GRID COMPUTING 10:(4) pp. 601-630. (2012)

[6] Michael Stonebraker, U□ur Çetintemel, Stan Zdonik: The 8 requirements of real-time stream processing, ACM SIGMOD Volume 34 Issue 4, December 2005 Pages 42 - 47

[7] Don Carney, U□ur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Greg Seidman, Michael Stonebraker, Nesime Tatbul, Stan Zdonik: Monitoring streams: a new class of data management applications, VLDB '02 Proceedings of the 28th international conference on Very Large Data Bases Pages 215 - 226 VLDB Endowment ©2002

[8] Amin Zahedi-Khameneha, Raimar J. Scherera, Mehdi Zaréb: A non-parametric wave type based model for real-time prediction of strong ground motion accelerogram, Soil Dynamics and Earthquake Engineering Volume 49, June 2013, Pages 181–196

[9] Branzanti, M.; Colosimo, G.; Crespi, M. ; Mazzoni, A. :GPS Near-Real-Time Coseismic Displacements for the Great Tohoku-oki Earthquake, Geoscience and Remote Sensing Letters, IEEE March 2013 Page 372 - 376 DOI: 10.1109/LGRS.2012.2207704

[10] Jianghui Geng, Yehuda Bock, Diego, Melgar, Brendan W. Crowell, Jennifer S. Haase, Cecil H., Ida M: A new seismogeodetic approach applied to GPS and accelerometer observations of the 2012 Brawley seismic swarm: Implications for earthquake early warning, DOI: 10.1002/ggge.20144J. Clerk Maxwell, *A Treatise on Electricity and Magnetism*, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[11] Eva Sciacca, Marilena Bandieramonte, Ugo Becciani, Alessandro Costa, Mel Krokos, Pietro Massimino, Catia Petta, Costantino Pistagna, Simone Riggi and Fabio Vitello VisIVO Science Gateway: a Collaborative Environment for the Astrophysics Community Proceedings of the 5th International Workshop on Science Gateways. Zurich, Switzerland, 3-5 June, 2013

[12] Eva Sciacca, Marilena Bandieramonte, Ugo Becciani, Alessandro Costa, Mel Krokos, Pietro Massimino, Catia Petta, Costantino Pistagna, Simone Riggi and Fabio Vitello VisIVO Workflow-Oriented Science Gateway for Astrophysical Visualization 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2013 Processing (PDP)

[13] http://www.lpds.sztaki.hu/node/192 [acc.09.09.2013]