International Journal of Computational Intelligence Systems, Vol. 4, No. 4 (June, 2011), 672-679

# Optimization of workflow scheduling in Utility Management System with hierarchical neural network

**Srdjan Vukmirovic*, Aleksandar Erdeljan, Imre Lendak, Darko Capko**
*Computer and control department, University of Novi Sad, Trg Dositeja Obradovica 6*
*Novi Sad, Serbia*
*E-mail: srdjanvu@uns.ac.rs, ftn_erdeljan@uns.ac.rs,lendak@uns.ac.rs,dcapko@uns.ac.rs*

**Nemanja Nedic**
*Telvent DMS, LTD*
*Novi Sad, Serbia*
*E-mail: nemanja.nedic@telventdms.com*

Received 24 October 2010
Accepted 22 March 2011

**Abstract**

Grid computing could be the future computing paradigm for enterprise applications, one of its benefits being that it can be used for executing large scale applications. Utility Management Systems execute very large numbers of workflows with very high resource requirements. This paper proposes architecture for a new scheduling mechanism that dynamically executes a scheduling algorithm using feedback about the current status Grid nodes. Two Artificial Neural Networks were created in order to solve the scheduling problem. A case study is created for the Meter Data Management system with measurements from the Smart Metering system for the city of Novi Sad, Serbia. Performance tests show that significant improvement of overall execution time can be achieved by Hierarchical Artificial Neural Networks.

*Keywords*: Utility Management Systems, Hierarchical Neural Network, Grid Computing.

## 1. Introduction

Thanks to advances in wide-area network technologies and the low cost of computing resources, computational grids, emerging as attractive computing platforms, enable the sharing, selection, and aggregation of geographically distributed resources for solving large-scale problems in science, engineering, and commerce. One primary issue associated with the efficient and effective utilization of heterogeneous resources in a grid is job scheduling. Unlike scheduling problems in conventional distributed systems, this problem is much more complex because of the high degree of heterogeneity of resources, their connection with heterogeneous network, the high degree of dynamics, etc. [1]

With the rapid development of manufacturing technology associated with central processing elements and the ever-popular Internet, applying computational grids in various fields has garnered considerable attention. Computational grids employ network devices to connect an enormous amount of different computational resources on the Internet, thereby constituting a low-cost, high-performance computing platform capable of parallel processing. [2]

Supervisory Control and Data Acquisition (SCADA) systems are becoming more and more resource demanding because their scope is becoming wider. This trend is especially visible in distribution systems for utilities - Utility Management Systems (UMS), like power distribution systems or gas and water distribution. Processing and storage of measured data becomes a problem, since more and more measured values are introduced in the controlled system. This problem could be solved by using the computer Grid.

UMS have some special requirements, mainly because they have to communicate with end devices (sensors and actuators) and have to store very large volumes of time-series data about variable values. This makes workflow schedule control for UMS a special problem.

The development of the proposed workflow scheduling system was driven by requirement to develop a scheduling system for a commercial Meter Data Management (MDM) system. An MDM system has to manipulate a large number of workflows in a distributed environment, so the proposed architecture could be used to reduce hardware requirements by optimizing recourse usage.

A workflow is loosely defined as the automation of a process to co-ordinate people, data and tasks. Business workflows had been researched and utilized over many years; more recently science recognized the need for workflows, and several specialized workflow engines have been developed. [3]

Scheduling is the decision process that assigns application components to available resources to optimize various performance metrics. Grid workload management and scheduling subsystems enable the efficient distribution of tasks in Grid systems and allow their transparent execution by hiding the complexity of the Grid infrastructure.

A novel hierarchical neural network model is proposed in this paper. It will solve the problem of workflow scheduling in large scale Utility Management Systems (UMS). The neural model is made up of two feed forward neural networks – one on top of the other.

In general, the considered scenario has several phases. UMS clients initiate a request for data processing by calling appropriate server functions. The workflow manager running at the server decomposes these functions into tasks and makes an execution schedule (see Figure 1). As system resources are limited an efficient task mapping (i.e. execution scheduling) becomes a fundamental concern. The aim of this work is to provide novel system architecture and combine it with enhanced algorithms in order to boost the efficiency of the solution.

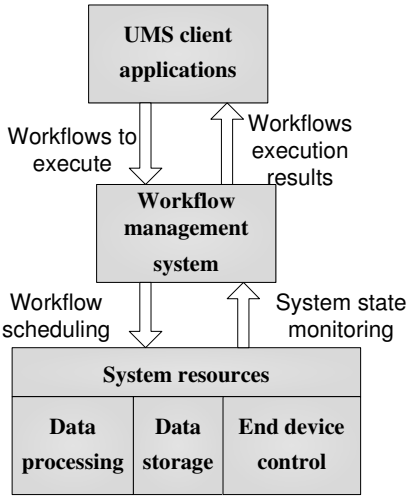


Fig. 1. Overall system architecture

The rest of the article is organized as follows. In Section 2, we overview related work; Section 3 describes specific characteristics of the UMS system. In Section 4 we describe the architecture of the proposed system; Section 5 shows ANN basics and describes the architecture of the network used in this paper. Section 6 describes test results and future works, and Section 7 concludes the article.

## 2. Related work

Grid computing is a new approach in scientific applications. Recent advances in grid infrastructure and middleware development have enabled various types of applications in science and engineering to be deployed on the Grid. The applications include those for climate modeling, computational chemistry, bioinformatics and computational genomics, remote control of instruments, and distributed databases [4]. Computational grids have recently attracted considerable attention as cost-effective platforms for parallel processing. [2]

Some researchers argue that Grid computing is the future computing paradigm for enterprise applications. Large scale grids are complex systems, composed of thousands of components belonging to distributed domains [5]. With the development of large-scale high-

speed networks, the Grid has become an attractive computational platform for high-performance parallel and distributed applications [6].

The Grid infrastructure is used both to share expensive and centralized resources among many scientists, as well as to integrate experimental data sources with the simulation codes necessary to analyze them [7].

The Grid connects computers, databases, instruments, and people in a seamless web, supporting computation-rich application concepts such as distributed supercomputing, smart instruments, and data-mining. However, its use has been limited to specialists, principally due to the lack of usability [8].

At present, in many modern Grid infrastructures, scheduling relies only on static properties and pre-determined states of resources. Therefore we argue that resource utilization can be enhanced by the addition of run time information and of forecasting capabilities. For this reason, we propose an approach to Workflow Management and scheduling that extends the existing solutions with a reasoning activity on Grid properties and states. The proposed approach relies on the availability of monitoring information, offering a snapshot of the system. This information, analyzed by the Workflow Scheduler, allows the prediction of future system states.

Criteria can pursue different goals: the minimization of a single task's execution time, the minimization of workflow execution time, the fairness of load distribution, maximum time of execution per workflow type, etc. Optimization rules are based on quantifiable metrics like: the workflow reliability and distribution fairness, workflow average execution time, etc. [6]

SCADA systems have come a long way from the simple visualization of the process. Distributed SCADA systems are very well described in scientific papers [9]. Process data found their way to the Internet [10] or even mobile phones [11].

The upcoming need for a Grid approach could be envisioned by observing the volumes of data that have to be stored and processed. This is especially visible in Utility Management Systems like power distribution systems, where in Distribution Management Systems (DMS) the number of process variables exceed tens of thousands [12]. Even more control variables are used in Meter Data Management (MDM) systems, which are responsible for controlling smart meters for individual power consumers. The latest customer requirements shows that a few millions of these smart meters will be implemented inside complex MDM systems, so this number will have to be multiplied by dozens of controlling variables per meter [13]. This kind of requirements calls for a new breed of UMS systems with a new approach to the architecture, and the Grid approach seems like the right approach that promises good performance.

The hierarchical neural network approach was successfully used for a wide range of real word problems from load forecasting to speech recognition and image processing [14]. In this paper we propose the Hierarchical Artificial Neural Network for workflow scheduling in Utility Management Systems.

## 3. UMS Architecture

In most general scientific workflow scheduling approaches only two parameters are considered for the workflow schedule: the computing power of the node and the network bandwidth between nodes. In a large UMS workflow there is an extended set of Grid parameters that should be considered. We have analyzed the architecture and requirements for large scale distributed UMS systems (electric energy distribution systems, gas and water distribution systems etc.) We have found that grid nodes in this type of UMS systems could be one of the following types:

**Processing node** – used for business calculation and data preprocessing, mainly for reporting and offline analysis of the system.

**Objects database node** – This node is used for storing static data from the distributed UMS system. Most commonly it hosts some kind of a relational database for better search performances.

**Time-series node** – This node hosts data about process variables. Since the value of the process variable could change very often the number of variables could be very large.

**Communication node** – This computer node is responsible for communication with end devices. In many cases this node communicates by a wireless network, which means that the current status of communication media could significantly impact the performance of the whole Grid.

This means that for developing an optimal strategy for a workflow scheduling system feedback signals will have to be used for all four kinds of performance indicators. In this way the Workflow Scheduler will be able to

create an optimal strategy of the dynamic state of the system.

Some other variables that could be used for developing an optimal strategy are: an average execution time for previous instances of the same workflow, the priority of the workflow (for alarms purposes), and cancellation workflows – if some new event in the system could cancel the execution of a previously stated workflow. These new optimization constraints could be easily introduced in the proposed architecture.

UMS workflows could include various processing of data, storing and querying data about devices and about process variable values, communication with end devices in order to get current values or send a command. Workflows chosen for this paper are defined on the basis of real UMS use cases. The following conditions are implied:

- All workflows are independent of each other.
- All workflows have the same priority.
- Every node processes only one workflow at a time.
- Every workflow is processed on one node at a time.
- Workflow of the same type has the same execution time on each node.

For this paper we have implemented five workflows which use different types of Grid nodes:

**1. Direct Command** - this workflow is responsible for sending commands to actuators. When executing it will send commands to actuators through a Communication node, and after that write command results to the time series database in a Time-series node.

**2. Command with preprocessing** - this type of command requires data preprocessing before issuing commands. These commands could be used when business logic has to be applied before an actuator is used. In this workflow, first a Processing node is called in order to prepare data, and after that a Communication node is used to send commands to actuators.

**3. Read Variable values** – in this scenario, variables are read from cache (previously read from devices). Workflows execution of this type starts in an Objects database node in order to filter end devices that should be read, and after that execution is transferred to a Time-series node to read selected variables values.

**4. Read Variable values From Device** – contrary to the workflow of type 3, workflows of this type have to read values from sensors on demand. This use case is used when the user needs the latest values. When executing this, workflow is migrated to an Objects database node in order to filter end devices that should be read, and after that a Communication node is used to send reading to sensors on demand.

**5. Reporting Inquiry** – this workflow covers various types of data processing, from reporting to device usage statistics. When executed, this workflow is migrated to an Objects database node in order to retrieve data needed for the calculation and after that it is migrated to a Processing node which is responsible for the calculation.

Figure 2 presents the execution plan of workflow migration for a set of previously defined UMS workflows. In order to make the optimization goal more obvious different Grid nodes are colored differently. The optimization goal in this paper is to rearrange incoming workflows in order to get maximum coverage of all colors; this means that all Grid nodes should be evenly loaded in time. This workflow scheduling plan will result in minimum overall execution time of all workflows.

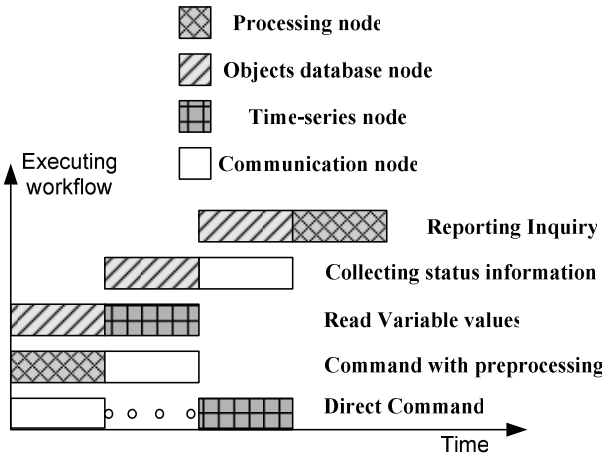


Fig. 2. Workflow execution migration between nodes

All workflows in the described systems have two migration stages. In more complex systems some workflows could have more migration steps. In some cases one workflow could be migrated back to the same node (if, for example, processing is done prior and post to communication with end devices). We anticipate that in these cases the usage of soft computing methods will generate even more performance improvement.

## 4. Proposed architecture

The proposed architecture takes into account the dynamic nature of a real-world UMS and it uses the Grid environment approach for detecting and responding to the changing environment in UMS. The developed framework is shown in Figure 3 and it provides the required support for feedback from the scheduling process.

As the figure shows, the framework consists of three major components: the Grid Manager, the Workflow Scheduler, and the Application Manager.

The **Workflow Scheduler** is responsible for deciding which queued workflow will be executed next. This decision is based on control variables from the monitored Grid. This component is of the most interest in our research, since decision making is done there. We anticipate that significant performance improvements could be achieved by using soft computing optimization methods in this component.

The **Grid Manager** is responsible for monitoring the state of the Grid and controlling the execution of workflows in it. It collects values of control variables that are used inside the Workflow Scheduler.

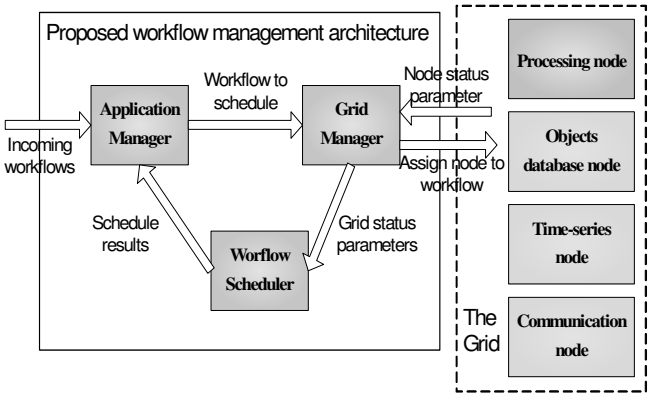


Fig. 3. Proposed architecture of the system

**Application Manager** is responsible for receiving workflows in execution time. It then queues and works with the Workflow Scheduler to determine the right time to send workflow to execution in the Grid. After receiving instructions from the Workflow Schedule it sends workflow to the Grid Manager, and at the end the results of the workflow execution are written into a database and sent to a client that initialized the workflow execution. The Application Manager is the only component visible from outside of the proposed system – thereby it hosts service interfaces for the initialization of the workflow execution by the client.

## 5. Neural network architecture

The classical methods for forecasting include regression and state space methods. The more modern methods include expert systems, fuzzy systems, evolutionary programming, Artificial Neural Networks (ANN) and various combinations of these tools. Among the many existing tools, the ANN has received much attention because of its clear model, easy implementation and good performance [15]. ANN have become popular in various real world applications including prediction and forecasting, function approximation, clustering, speech recognition and synthesis, pattern recognition and classification, and many others. [16]

The multi-layer feed forward network will learn to associate the given output vectors with input vectors by adjusting their weights, which are based on the error at the output. The weight modification algorithm is the steepest descent algorithm (often called the delta rule) to minimize a nonlinear function [17]. The algorithm is called back error propagation or back propagation because errors are propagated back through the hidden layers.

The weights can be learned by training the network using a training set of target states $x_p^T$ for the output for a given set of inputs where p is the training pattern. The steepest descent algorithm essentially seeks to choose the weights during training. The mean square error E between the target output $x^T$ and the actual output x of the network over all training data is minimized. Thus, the weights are chosen for n output neurons so that the back propagation algorithm minimizes the mean square error E for N training samples. The mean square errors can be calculated by (1)

$$E = \frac{1}{N}\sum_{p=1}^{N}\sum_{i=1}^{n}\left(x_{pi}^T - x_{pi}\right)^2 \quad (1)$$

The delta rule is then used to find the minimum of E by differentiating it with the weight $T_{ij}$. The weight $T_{ij}$ is changed by (2)

$$\Delta T_{ij} = -\eta\frac{\partial E}{\partial T_{ij}} \quad (2)$$

where $\eta$ is called the learning rate. After calculating $\Delta T_{ij}$, the new weight is given by

$$T_{ij,new} = T_{ij,old} + \Delta T_{ij} \quad (3)$$

In Workflow Scheduling applications, the main function of the ANN is to predict the duration of the execution of a specified type of workflow based on the current state of the Grid.

Figure 4 represents two types of ANN that are used for workflow scheduling in UMS. The simple ANN predicts the execution time of the specified workflow type based on the current state of the Grid nodes. The hierarchical ANN has two networks, one on top of another. The bottom ANN forecasts the next state of the Grid and the Top ANN predicts the execution time of workflow based on the current and predicted state of the Grid.

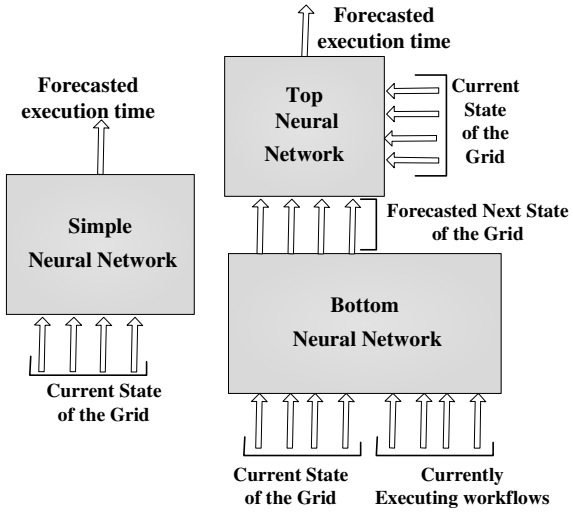


Fig. 4. Hierarchical ANN architecture

### 5.1. *Simple Neural Network*

Five inputs for Artificial Neural Network are selected for the Simple ANN that proved to be the most important for optimal scheduling. The Output of ANN is chosen in a way that will uniquely identify the type of workflow that should be started in the current Grid state. From the experiments conducted on experiment data we concluded that the optimal ANN has one hidden layer with ten neurons. Standard Windows Performance Counters are used for monitoring appropriate features of the Grid node.

The selected ANN inputs are:

- Current CPU Usage from Processing node [%]
- Current query execution time for Objects database node [%]
- Current read/write duration for Time-series node [%]
- Network bandwidth – Communication node [%]
- Workflow type - provided by an application [one of the five provided]

The selected ANN output is:

- Forecasted execution time [s]

### 5.2. *Hierarchical Neural Network*

Hierarchical ANN consists of two separate networks one on top of the other. The **Bottom ANN** has eight inputs: the first four represent the current state of the Grid nodes (the same as the first four inputs for the Simple ANN), and other four inputs are the types of workflows that are currently executing on Grid nodes. Four outputs of the Bottom ANN are the forecasted state of the Grid nodes in the next step.

The **Top ANN** has three groups of inputs: the first four inputs are forecasted performances of the Grid nodes (outputs of the Bottom ANN), the next four inputs are the current status of the Grid nodes (the same as the first four inputs for the Simple ANN), and the last input is the type of workflow. The output of the Top ANN is the forecasted execution time of the specified workflow type.

Both ANN have one hidden layer with ten neurons. The activation function for all layers was the Logarithmic sigmoid transfer function (4).

$$\text{logsig}(x) = \frac{1}{(1+e^{-x})} \quad (4)$$

## 6. Results and discussion

The test environment was developed based on the proposed architecture. One computer node was dedicated for each type of nodes specific for UMS systems. To achieve full experiment credibility we have pre-defined the set of applications (i.e. workflows) that has been provided as an input to the considered system under the test.

For testing purposes we have scheduled 5 types of UMS workflow schedules defined in Section 3. The number of workflows was from ten to a thousand. We

have test total execution times of all workflows with the following scheduling logic:

**No optimization** – workflows were scheduled in the order in which they arrived in the input queue. This scenario was used for comparison.

**Hard-coded optimization** – in this scenario we have used simple logic that says that workflows will execute more efficiently if they are of different types. So when the time comes for the new workflow to be executed, the Workflow Scheduler examines which workflow types are currently executing. Then the Workflow Scheduler tries to find the workflow of the type which is least deployed and tries to start executing that workflow.

**Round robin selection of workflow type –** this approach is proposed in [18]. It suggests that every time the workflow is about to be scheduled a different type of workflow should be scheduled in round robin manner. It is different than previous approach (hard-coded optimization) because in this case it doesn't matter which types of workflow are currently executing. This approach is easier to implement and it does not need information about currently executing workflows. However approach proposed in this paper shows better performances.

**Simple ANN optimization** – in this scenario we used the output of the simple ANN to select which type of workflow should be scheduled in the next step.

**Hierarchical ANN optimization** – the output of the Hierarchical ANN was used for selecting optimal workflow to be scheduled.

The most important part of any Neural Network is the learning set. The training data for ANN consisted of 1000 sets of values, 90% of which were used for training, and 10% were used for verification. The training of ANN was executed several times and in all cases the error on the training data was less than 0.1 percent, and the error on the verification data was less than 2 percent.

A case study is created for the Meter Data Management system with measurements from the Smart Metering system for the city of Novi Sad, Serbia. Workflows were executed on the Grid in the order in which they arrived in the system (without any optimization of the scheduling process). In this part the execution time of workflow was recorded as well as its type and the state of the Grid nodes in the moment of the workflow execution start.

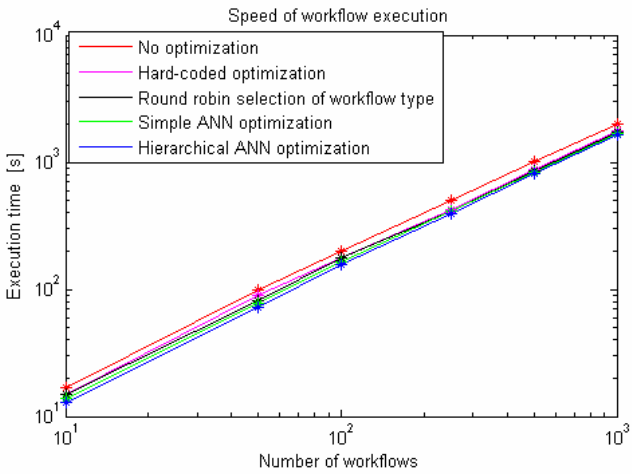


Fig. 5. Speed of workflow execution

The test results are shown in Table 1, where the execution time is shown in seconds. Figure 5 shows the same results in the graphical form. The test results clearly show that introducing the Hierarchical ANN in the workflow scheduling process for large scale UMS systems could improve overall performances of the Utility Management System. The percentage of performance improvement varies depending on the number of scheduled workflows.

Table 1. Test results for the speed of execution of workflows

| Number of workflows | Time of execution – No optimization [s] | Time of execution hard-coded optimization [s] | Time of execution Round robin selection of workflow type [s] | Time of execution – simple ANN optimization [s] | Simple ANN improvement rate [%] | Time of execution – Hierarchical ANN optimization [s] | Hierarchical ANN improvement rate [%] |
|---|---|---|---|---|---|---|---|
| 10 | 17 | 15 | 15 | 14 | 6.7 | 13 | 7.1 |
| 50 | 99 | 90 | 82 | 77 | 14.4 | 72 | 6.5 |
| 100 | 199 | 176 | 175 | 163 | 7.4 | 156 | 4.3 |
| 250 | 501 | 422 | 412 | 412 | 2.3 | 391 | 5.1 |
| 500 | 1007 | 868 | 863 | 827 | 4.7 | 809 | 2.2 |
| 1000 | 2018 | 1762 | 1709 | 1671 | 5.1 | 1625 | 2.7 |

## 7. Conclusion

In this paper we propose an architecture that allows optimal scheduling of Workflows in Utility Management Systems. The architecture is based on the feedback from the Grid based on the performance monitoring indicator. This way the Workflow Schedule can optimize the execution time of workflows.

This paper also explains specific features of large scale Utility Management Systems that make the proposed architecture different from standard Grid scheduling systems. A set of control variables is also proposed, which is an additional contribution of this work.

The workflow scheduling component, which is responsible for starting appropriate workflows, uses the Artificial Neural Network for choosing the optimal scheduling strategy. Performance analysis shows that this approach significantly boosts the performance of the whole system; the total execution time is reduced by approximately five percent. The introduction of the Hierarchical Artificial Neural Network could additionally reduce the overall execution time of workflows. This improvement is based on the fact that HANN can predict the future state of the Grid nodes and optimize scheduling based on that information.

This scheduling mechanism can significantly reduce investments in hardware as the same results could be achieved with less Grid nodes.

## References


1. Kang Q., Hong H., Wang H., Jiang C., A Novel Discrete Particle Swarm Optimization Algorithm for Job Scheduling in Grids, The 4th International Conference on Natural Computation, Jinan, Shandong, China, 2008, pp. 401 – 405, October 2008
2. Chiang C., Applying an improved fast ANT system to DAG scheduling for computational Grids, Journal of the Chinese Institute of Engineers, Vol. 31, No. 7, pp. 1113-1125, 2008
3. Rygg A., Mann S., Roe P., Wong O., Bio-Workflows with BizTalk: Using a Commercial Workflow Engine for eScience, Proceedings of the First International Conference on e-Science and Grid Computing, 8 pp. - 123, January 2006
4. Abramson D., Lynch A., Takemiya H., Tanimura Y. et. al. Deploying scientific applications to the pragma grid testbed: Strategies and lessons. In Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid, Washington, DC, USA, pp. 241–248, 2006
5. Menasc A., Casalicchio E., A Framework for Resource Allocation in Grid Computing, 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, pp. 259 - 267, 2004
6. Zhang Y., Koelbel C., Cooper K., Hybrid Re-Scheduling Mechanisms for Workflow Applications on Multi-cluster Grid, 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, Shanghai, China, pp. 116 – 123, June 2009
7. Allen G., Angulo D., Goodale T., Kielmann T. et. al., GridLab: Enabling applications on the Grid. In GRID '02: Proceedings of the Third International Workshop on Grid Computing, London, UK, pp. 39–45, 2002.
8. Berman F., Casanova H., Chien A., Kennedy K. et. al. New grid scheduling and rescheduling methods in the grads project. Int. J. Parallel Program., 33(2) pp. 209–229, 2005.
9. Agrawal K., Merh B., Fatnani P., Yadav R., Gangopadhyay S., SCADA functionality for control operations of INDUS-2, 10th ICALEPCS Int. Conf. on Accelerator & Large Expt. Physics Control Systems. Geneva, p. 1-6, October 2005
10. Qiu B., Gooi H. B., Web-Based SCADA Display Systems (WSDS) for Access via Internet, Power Systems, IEEE Transactions on, Issue 2, pp. 681-686, May 2000
11. Ozdemir E., Karacor M., Mobile phone based SCADA for industrial automation, ISA Transactions, Volume 45, Number 1, pp. 67–75, January 2006
12. Srđan Vukmirović, Aleksandar Erdeljan, Imre Lendak, Darko Čapko, Extension of the Common Information Model with Virtual Meter, Electronics and electrical engineering, ISSN 1392 – 1215, 2011, (107), pp. 59-64.
13. Vukmirovic S., Lukovic S., Erdeljan A., Kulic F., Software architecture for Smart Metering systems with Virtual Power Plant, The 15th IEEE Mediterranean Electrotechnical Conference, Valletta, Malta, pp. 448 – 451, April 2010
14. Carpinteiro O., Lemeb R., de Souza A., Pinheiro C., Moreira E., Long-term load forecasting via a hierarchical neural model with time integrators, Electric Power Systems Research 77, pp. 371–378, 2007
15. Yamin H, Shahidehpour S., Li Z. Adaptive short-term electricity price forecasting using artificial neural networks in the restructured power markets, Electrical Power Energy Systems, 26 pp. 571–81, 2004
16. Mandal P., Senjyu T., Funabashi T., Neural networks approach to forecast several hour ahead electricity prices and loads in deregulated market, Energy Conversion and Management 47, pp. 2128–2142, 2006
17. Rumelhart D., Hinton G., Williams R. Learning internal representations by error propagation, Parallel Distributed Processing, Vol. 1, Chapter 8, MIT Press, pp. 318-362 1986.
18. Ungurean I., Job Scheduling Algorithm based on Dynamic Management of Resources Provided by Grid Computing Systems, Electronics and Electrical Engineering, 2010.–No. 7(103), pp. 57–60