AUDIT ON NETWORK WORK PROCESS ADMINISTRATION FRAMEWORK

[1]DEEPA.B
[1]GUEST FACULTY IN COMPUTER SCIENCE
[1]AMUV,TORVI, VIJAYAPUR,INDIA

Abstract : Unique A work process application comprises of a piece of part applications that must be executed in the request controlled by their control and information reliance These can have various segment applications that can be executed simultaneously, or once in a while only a solitary segment which can be executed in parallel on numerous processors. Such parallelization brings about efficient usage of the figure energy of Network. The primary issues engaged with work process specification are the work process portrayal model and data about the applications sent on lattice. Likewise, for the execution of work process, the fundamental issues to be tended to are application disclosure and determination, co appointment of work process execution and giving adaptation to internal failure instruments. Network middleware, which tends to the previously mentioned issues in work process specification and execution, has been named as Framework work process administration framework {GWMS}.

Keywords—work process, framework, GWMS.

I. INTRODUCTION

Grid computing coordinates computational and date resources that are not subject to centralized control. This coordination is achieved with the help of standard, open, general purpose protocols and interfaces to deliver on trivial qualities of service such as response time, throughput, security, etc[1]. Scientists from different fields, such as high energy physics and gravitational physics [2][3], meteorology [4]), bioinformatics [5] [6], etc. are using grid technology for compute intensive applications. Such applications are generally not single standalone applications, but they are workflow applications. A workflow application consists of a composition of component applications that must be executed in the order determined by their control and data dependence These can have multiple component applications that can be executed concurrently, or sometimes just a single component which can be executed in parallel on multiple processors. Such parallelization results in efficient utilization of the compute power of Grid. The main issues involved in workflow specification are the workflow representation model and information about the applications deployed on grid. Also, for the execution of workflow, the main issues to be addressed are application discovery and selection, coordination of workflow execution and providing fault tolerance mechanisms. Grid middleware, which addresses the above-mentioned issues in workflow specification and execution, has been termed as Grid workflow management system {GWMS} [7]. Such grid middleware allows the user to compose workflows using a workflow specification model and also automates the workflow execution on grid. Thus, the user is not concerned with lower level details like application discovery, grid—specific commands, etc. This allows even a non expert grid user to use the compute power of grid to solve his problem.

A workflow specified by a user using the new workflow model may have structural errors like deadlock or lack of synchronization. These structural errors will result in erroneous execution of workflow. Hence these errors need to be detected before a workflow is submitted for execution.

II. MAJOR ELEMENTS OF GRID WORKFLOW MANAGEMENT SYSTEM

A Grid Workflow Management system comprises of two main components based on functionality: Workflow editor which allows user to compose workflows and workflow engines which manages execution of workflow on grid. Following are the major elements of a GWMS [7]:

- Workflow specification model: A workflow can be modelled using the basic patterns. A DAG—based workflow will allow modeling of sequence, parallel and choice patterns while a non—DAG based model will allow modeling of all the four patterns shown in the figure.
- A workflow can be specified using a mark—up language such as XML (GridAnt [8]). However, this approach requires that the user must know the language—specific syntax. Also this approach becomes very inconvenient when workflows are very complex. Another approach which is more user—friendly is graph based modeling, where a user can create a workflow by using a graphical interface, which allows the user to add workflow tasks and then connect them to reflect the control and data dependences between the tasks. Different graphical models like DAG (Pegasus [5], Taverna [6]), Petri net (GridFlow [8], eXeGrid [9]), Activity diagrams (Askalon [10]), etc. have been used for workflow specification.

  In grid context, workflows can be divided into two types, which are abstract and concrete workflows [5]. A workflow which does not specify the binding of tasks to a specific resource on grid during workflow composition has been termed as abstract workflow. Such abstract workflows allow the binding of tasks to resources being delayed until run—time to handle the dynamic nature of grid where machines can join and leave the grid. If a workflow specification involves binding of task nodes to a specific grid resources, the corresponding workflow is called a concrete workflow. A concrete model allows the user to explicitly specify the particular grid resource for each workflow task.

In addition, Quality of service (QOS) constraints like execution time, cost, etc. can be specified either for each individual task or for the entire workflow. Such user—specified constraints are taken into account during selection of a particular grid resource during workflow execution.

Information Retrieval: A workflow management system needs to maintain and retrieve static information such as the applications deployed on grid and dynamic information such as current application availability. The static information like the I / O helping the user in workflow specification.

Scheduling : A grid scheduler attempts to create a mapping between the workflow tasks and applications available on the grid with the intention to optimize some objective function, such as execution time of workflow or throughput.

Fault Tolerance : A Grid workflow management system should include fault tolerance functionality in order to handle the failures that can occur in a dynamic grid environment. Some of the failure reasons are non—availability of required applications, faults in underlying fabric components, etc. Fault tolerance techniques working at task-level address the effects of execution failure of individual workflow tasks. Some of these techniques include retrying the task execution on same compute server after failure [11], look for an alternate compute server, check pointing/restart mechanisms [12], etc. Fault tolerance techniques working at workflow-Zeoet modify the workflow structure such as use of alternate tasks to deal with errors occurring during workflow execution [13].

III. RELATED WORK

In this chapter, we describe the non—DAG based workflow specification models of some other GWMS. These systems allow the user to specify a workflow using a graphical model. Each workflow specification model is described in terms of the constructs provided to model the four workflow patterns mentioned above. Also, we mention the analysis techniques, if any, used to analyze workflows by each system.

We first mention some terms [14] used in this section. A node having two or more successor nodes, which can execute concurrently, is called an AND-split node.

An AND-join node has two or more predecessor nodes. It is used to synchronize the concurrent paths of execution resulting from a corresponding AND—split node. Similarly, a node that has two or more successor nodes, only one of which can be executed, is called an OR-sptit node. An OR-join node is used to join the alternative mutually exclusive execution paths resulting from an OR—split node into one execution path.

Triana

Triana [15,16] has been developed at Cardiff University. It is currently being used for different tasks such as signal, text and image processing. The workflow specification model of Triana allows the following modeling constructs.

- Parallel : Parallelism can be specified in a workflow by setting the number of I/O units of a task node. If a node has two or more two output units, then the successor tasks of this node can execute concurrently. If a node has two or more input units, the node will act as an AND—join node.
- Choice : Conditions are specified with the help of IF—split and IF—join nodes. Only one of the alternative execution paths arising from an IF—split node is followed during execution. All these possible execution paths are merged using an IF—join node. Another control construct supported is the Switch construct. It is used to specify a conditional expression, which is evaluated to decide which of the two inputs (of predecessor nodes of the switch node) is to be used by the successor node.
- Iteration : In order to specify a loop in a workflow, the sub—workflow to be iterated upon is grouped together. An explicit loop handler node is added automatically to this group. This loop handler node handles the iterations over the sub—workflow. An exit condition has to be specified with a loop handler node which decides when to exit the loop. Triana supports fired iterator construct and conditional while loops.

  The usage of a sub—workflow to be iterated upon as a group defines the entry and exit points of the loop explicitly. This simplifies the workflow representation in XML. However, this restricts the specification of complex interactions with the tasks which are part of the loop.

IV. CONCLUSION

A comparison of our workflow specification model with those of some other GWMS mentioned in Section 3. We do not give a comparison in terms of the other elements of a GWMS. Modelling of parallel, choice and loop patterns in a workflow: These patterns can be specified explicitly, i.e. by using nodes of other types than the nodes used to represent tasks in the workflow. Otherwise, they can be specified using tasks nodes only. The comparison with respect to the verification mechanisms suggests that verification of structural correctness has not been provided by any of the mentioned Grid workflow management systems. A workflow found correct by structural verification techniques can be executed correctly for all possible workflow instances. This gives the user a guarantee that a specified workflow which is found structurally correct, will execute correctly provided the data flow between the workflow tasks is valid.

REFERENCES

[1] Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid:Enabling Scalable Virtual Organizations. In International Journal of Super-Computer applications, volume 15(3), 2001.

[2] .GriPhyN home page. http://www.griphyn.org/.

[3] GridLab home page. http://www.gridlab.org/.

[4] A. R. Price, G. Xue, A. Yool, D. J. Lunt, T.M. Lenton, J. L. Wason, G. E. Pound, Cox S. J., and the GENIE team. Tuning GENIE Earth System Model Components using a Grid Enabled Data Management System. In Proceedings of the UK e-Science All Hands Meeting 2004, pages 593—600, Nottingham, UK, 2004.

[5] Ewa Deelman, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Sonal Patil, Mei—Hui Su, Karan Vahi, and Miron Livny. Pegasus : Mapping Scientific Workflows onto the Grid. In Across Grids Conference 2004, Nicosia, Cyprus, 2004.

[6] Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Mark Greenwood, Tim Carver, Matthew R. Pocock, Anil Wipat, and Peter Li. Taverna: a tool for the composition ows. Bioinformatics Journal, 16 June, 2004

[7] Yu, Jia, and Rajkumar Buyya. "A taxonomy of workflow management systems for grid computing." Journal of Grid Computing 3.3-4 (2005): 171-200.

[8] Amin, Kaizar, et al. "Gridant: A client-controllable grid workflow system." System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on. IEEE, 2004.

[9] Cao, Junwei, et al. "Gridflow: Workflow management for grid computing." Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on. IEEE, 2003.

[10] Andreas Hoheisel. User Tools and Languages for Graph—based Grid Workflows. Grid Workflow 2004 Special Issue of Concurrency and Computation: Practice and Experience, March 2004.

[11] Askalon home page. www.par.univie.ac.at/ project / askalon.

[12] Taverna User Manual(1.2). http://taverna.sourceforge.net/usermanual/docs.word.html.

[13] Condor manual. http://www.cs.wisc.edu/condor/manual/v64/Contents.html.

[14] Soonwook Hwang and Carl Kesselman. GridWorkflow: A Flexible Failure Handling Framework for the Grid. 12th IEEE International Symposium on High Performance Distributed Computing, 2003.

[15] Wasim Sadiq and Maria E. Orlowska. Analyzing process models using graph reduction techniques. Inf. Syst, 25(2):117—134, 2000.

[16] David Churches, Gabor Gombas, Andrew Harrison, Jason Maassen, Craig Robinson, Matthew Shields, Ian Taylor, and Ian Wang. Programming Scientific and Distributed Workflow with Triana Services. G GF Workflow Workshop, Special Issue of Concurrency and Computation: Practice and Experience, 2005.

[17] Majithia, Shalil, et al. "Triana: A graphical web service composition and execution toolkit." Web Services, 2004. Proceedings. IEEE International Conference on. IEEE, 2004.