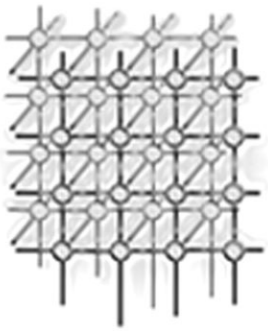
A taxonomy of grid workflow verification and validation



Jinjun Chen[*,†] and Yun Yang

Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology, P.O. Box 218, Hawthorn, Melbourne, Australia 3122.

SUMMARY

In a grid architecture, a grid workflow management system is a type of high-level grid middleware which is supposed to support modelling, redesign and execution of large-scale sophisticated scientific and business processes in many complex e-science and e-business applications. To ensure the correctness of grid workflow specification and execution, grid workflow verification and validation must be conducted. However, current research on grid workflow verification and validation is at the infancy stage and very few projects focus on them. Therefore, a systematic investigation and an overall classification of key issues in grid workflow verification and validation is helpful and should be presented so that we can keep on the right track and reduce unnecessary work as much as possible. As such, in this paper, we analyse the grid workflow verification and validation and present a taxonomy. Especially, we identify some important open points which are not discussed by the current research and hence need further investigation. The taxonomy is aimed at providing an overall picture of grid workflow verification and validation.

KEY WORDS: Grid workflow management systems; Grid workflow verification; Grid workflow validation

---

[*]Correspondence to: Jinjun Chen, Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology, P.O. Box 218, Melbourne, Australia 3122.
[†]E-mail: jchen@ict.swin.edu.au.
Contract/grant sponsor: Australian Research Council Discovery Project under grant No. DP0663841 and Linkage Project under grant No. LP0669660.

1. INTRODUCTION

In a grid architecture, a grid workflow management system is a type of high-level grid middleware which is supposed to support modelling, redesign and execution of large-scale sophisticated scientific and business processes in many complex e-science and e-business applications such as climate modelling, astrophysics, medical surgery, disaster recovery, international finance and insurance [1, 2, 3, 4, 5, 6]. A grid workflow management system normally contains a number of grid workflows in the form of their specifications and instances [2, 7, 8]. Generally speaking, a grid workflow management system works at three stages: build-time, run-time instantiation and run-time execution [9, 10, 11, 12]. At build-time stage, complex scientific or business processes are modelled or redesigned as grid workflow specifications [9, 11, 13, 14]. According to [9, 10, 15, 16], conceptually, a grid workflow contains a lot of computation or data intensive activities as well as dependencies between them. These activities are implemented and executed by corresponding grid services [9, 11, 17]. The dependencies define activity execution orders [9, 11, 17]. At run-time instantiation stage, grid workflow instances are created, and especially grid services that are specified in the build-time definition documents are discovered. This could include an instantiation service that is a high-level grid service [9, 11]. At run-time execution stage, grid workflow instances are executed. The execution is coordinated between grid services by the grid workflow engine which itself is also a high-level grid service, hence automatically grid aware [9, 11].

To ensure the well-performing of large-scale sophisticated scientific and business process support, we must ensure the correctness of grid workflow specification and execution. According to the IEEE Standard Glossary of Software Engineering Terminology [18], the correctness is defined as freedom from faults, meeting of specified requirements, and meeting of user needs and expectations. Furthermore, based on the IEEE Software Verification and Validation Standard [19], the correctness can be more commonly described as grid workflow verification and validation. Both grid workflow verification and validation are important. Verification failure results in the grid workflow specification and execution containing faults or flaws. Validation failure constitutes a breach of contract between the complex scientific and business process developer and the client. Clearly, neither is desirable. Figure 1 further depicts the relationship between correctness and verification & validation.

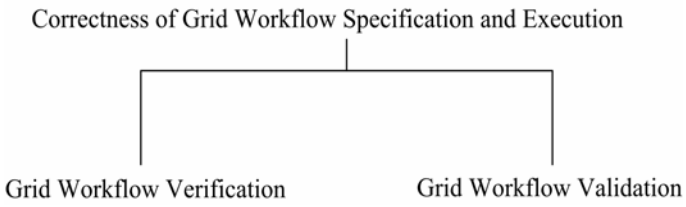


Figure 1. Correctness of grid workflow specification and execution vs grid workflow verification and validation

Some research related to (grid) workflow verification and validation has been done. [20, 21] discuss how to use Petri Net to model workflow processes. [22] presents twenty workflow patterns to represent the workflow expressive power. [23] analyses the structure errors in workflow specifications using Petri Net-based techniques. [24] uses the simulation method to compute the performance parameters. [25, 26] analyse resource constraints in the workflow specification and execution, and present some verification algorithms. [27, 28] discuss the specification, control, and enforcement of authorisation constraints. [29] assigns maximum and minimum durations to each activity and proposes some verification algorithms to check the consistency of relative and absolute temporal constraints. [30] discusses temporal dependency and its impact on the temporal verification effectiveness and efficiency in grid workflow systems. [31, 32, 33, 34] discuss checkpoint selection for temporal verification at run-time execution stage. [17] introduces four consistency states to a fixed-time constraint and develops corresponding verification algorithms. [3, 14, 35] discuss QoS (Quality of Service) scheduling issues and presents a grid economy based architecture with corresponding grid middleware. [36] discusses how to analyse workflow process models by using graph reduction techniques.

However, as far as grid workflow verification and validation is concerned, current research is still in infancy. To the best of our knowledge, very little work has been done and very few projects focus on grid workflow verification and validation issues. Therefore, a systematic identification and an overall classification of key issues in the grid workflow verification and validation field are needed which can help us find out main points in the field and avoid some unnecessary work as much as possible. Hence, in this paper, we investigate the key issues in grid workflow verification and validation, and present a taxonomy for them. Especially, we identify some important points which are not stated by the current research and hence need further investigation. The taxonomy depicts an overall picture for grid workflow verification and validation.

The remainder of the paper is organised as follows. Section 2 discusses the taxonomy of grid workflow verification and validation. Section 3 conducts a survey of existing projects and presents a further discussion. Section 4 concludes our contributions and points out future work.

2. TAXONOMY

We investigate grid workflow verification taxonomy in Section 2.1 and grid workflow validation taxonomy in Section 2.2.

2.1. Grid workflow verification

Specifically speaking, based on [19], grid workflow verification is mainly concerned with the specific correctness of grid workflow specification and execution such as no deadlock, no livelock, no temporal violation or no resource conflict. It aims at no faults in grid workflow specification and execution under the condition where complex scientific and business process requirements have already been correctly supported in grid workflow specification by the selected grid workflow management system. Correspondingly, as shown in Figure 2, grid

workflow verification taxonomy consists of six elements: (a) structure verification, (b) performance verification, (c) resource verification, (d) authorisation verification, (e) cost verification and (f) temporal verification. In this section, we look at each element in detail.

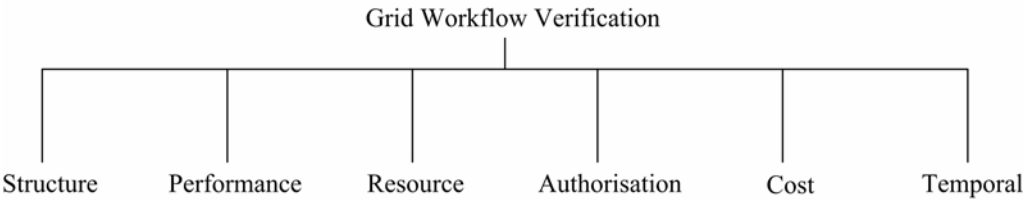


Figure 2. Elements of grid workflow verification

2.1.1. Structure verification

Structure verification aims to verify structure consistency [23, 24]. It consists of syntactic structure verification and semantic structure verification [23, 24, 36].

In a grid workflow specification, main syntactic structure inconsistencies include deadlock, livelock, lack of synchronisation, misuse of modelling objects and constructs, active end, dead activity and etc. [23, 24, 36]. Some methods such as Petri Net or directed graph with graph reduction technologies have been presented to model the structure and verify whether there are any structure inconsistencies [20, 21, 23].

Current research of structure verification mainly focuses on syntactic structure verification. However, semantic structure verification is also very important. For example, suppose we have a grid workflow specification which is correct in terms of syntax. So, it is using the selective structure correctly. Suppose at a decision activity point A, it has two branches B and C. Syntactically, it can go to B or C. However, semantically, at run-time, according to the execution results of A, it may need to go to D. Apparently, the grid workflow execution will stop because no D can be selected. That is to say, semantically, the structure of the grid workflow specification is incorrect. Therefore, semantic structure verification should also be investigated in addition to syntactic structure verification.

2.1.2. Performance verification

Performance verification aims to verify whether user defined performance parameters can be met [24]. A grid workflow specification or execution is normally attached with some user defined performance parameters such as average activity completion time, mean capacity utilisation rate, mean activity queuing time, mean activity synchronisation delay, mean activity set-up delay or mean resource allocation. [24, 37]. Therefore, we need to verify whether these parameters can be met by the selected grid workflow management system and current system run-time status. To conduct performance verification, we need to compute system performance parameters, and then compare them with user defined performance values. Currently, people

try to use Markovian chain theory, queueing theory or simulation tools for system performance parameter computations [24, 37]. Based on Markovian chain theory, we try to incorporate some properties into a Markovian chain, and then use the analysis methods provided by the Markovian chain theory to calculate system performance parameters. However, not every aspect of a grid workflow can be incorporated into the analysis and the Markovian chain analysis can also be very time-consuming [24, 37]. The Queuing theory can provide a good statistical analysis. However, many of the assumptions used in the queuing theory are not valid for grid workflow processes [24]. For example, in the presence of parallel routing, it is often impossible to apply the results obtained from the queuing theory because the distribution of activity completion time is normally not the one required for applying the queuing theory. Based on the simulation, we often convert a grid workflow specification into a simulation model, and then use some simulation tools to simulate grid workflow execution [24]. However, the challenge is that sometimes we cannot map a grid workflow specification into a simulation model without any semantic loss.

Further research might be conducted on how to extend the Markovian chain theory or queuing theory or simulation tools to allow for grid workflow-specific characteristics, and on the extent to which a grid workflow can be simplified to meet the requirements of the use of these theories or tools.

2.1.3. Resource verification

Resource verification aims to verify whether there are any resource conflicts between different activities or workflows [25, 26]. In a single grid workflow or among multiple ones, different activities may compete for the same resource such as a machine or a human being. Given that the grid computing infrastructure normally encompasses a number of heterogeneous and autonomous hosting environments, such resource competitions can happen often [7]. Hence, we must verify them to make sure that resource sharing proceeds correctly. Currently, some fundamental research has been done [25, 26]. However, in grid workflow management systems, run-time execution environments are very dynamic. One resource may serve more than one activities at the same time while sometimes more than one resource may serve one activity simultaneously. Therefore, we should incorporate the statistical property of grid workflow execution into verification mechanisms. For example, two activities may compete for the same machine during the same time interval. It may or may not lead to a resource conflict. We need to conduct some statistical analysis to judge whether there are really any conflicts. The further research on the resource verification based on statistical analysis is a challenge because, as discussed in the above performance verification section, some statistical analysis theories or tools cannot be equally applied to grid workflow management systems.

2.1.4. Authorisation verification

A grid workflow management system often needs to deploy heterogeneous and distributed hardware and software systems to execute a given grid workflow [6, 7, 38]. This gives rise to

decentralised security policies and mechanisms that need to be managed. To ensure the security, authorization verification must be conducted to check the consistency between resource access and participants or roles. Authorisation means that no resources are accessed by unauthorised participants or roles at anytime [27]. Therefore, some authorisation constraints are set for different participants and roles, which show which kinds of resources can be accessed by them and how to access authorised resources [27, 28, 39]. For the grid workflow specification, authorisation verification is conducted to judge whether access rules for corresponding participants or roles are correctly defined. For the grid workflow execution, authorisation verification is conducted to show whether there is any unauthorised resource access so that grid workflow management systems can take some protective measures immediately. One of key challenges for authorization verification is that we need to figure out what the grid workflow security is since the underlying grid infrastructure has already supported the security to some extent [2, 38].

2.1.5. Cost verification

In the real world, scientific and business processes are normally budget constrained [3, 14, 35]. Therefore, we must verify the cost aspect of grid workflow specification and execution to check whether grid workflow specification and execution can meet corresponding budget requirements. Some grid workflows have many local cost constraints, while others have only one end-to-end cost constraint. To the best of our literature review, no research has been done about cost verification. Although some important and valuable work concerning cost scheduling has been done [14, 35], cost scheduling cannot be used to judge whether grid workflow specification or execution is definitely going wrong when the scheduling assignment is not followed. This is normal as the judgement is one of the responsibilities of grid workflow verification. To conduct cost verification, we need to represent activity cost information in a grid workflow. We may need to enrich grid workflow modelling languages so that they can accommodate activity cost information. We then need to compute the cost between two activities and compare it with cost constraints to check the consistency. In some cases where only one end-to-end cost constraint exists, to more efficiently control the activity cost for grid workflow execution, we may need to investigate how to dynamically assign, verify and adjust some local cost constraints within the frame of the overall cost constraint. In fact, a grid workflow normally consists of a number of activities and consequently lasts a long time. Therefore, only one end-to-end cost constraint is not sufficient to control the budget aspect of grid workflow execution. We may find the end-to-end cost constraint is violated at the last activity. Then, it is too late to take any compensating or precaution measures. Therefore, if there is only one end-to-end cost constraint, we must investigate how to divide it into some smaller local cost constraints. Then, by these local cost constraints, we can better control the budget aspect of grid workflow execution. With local cost constraints, we need to figure out how to compute the cost between two activities for verification purpose. The run-time uncertainty of activity cost must be taken into consideration so that corresponding statistical features can be considered.

2.1.6. Temporal verification

Similar to the cost, in the real world, scientific and business processes normally stay in a temporal context and are often time constrained [3, 33]. Therefore, temporal verification must be conducted to check whether there are any temporal constraint violations. We need to investigate four factors to conduct temporal verification: temporal consistency states, assignment of fine-grained temporal constraints, selection of checkpoints, and verification of temporal constraints.

To verify temporal constraints, temporal consistency states must be defined. Some research on timed workflow modelling has been done such as [29, 40, 41, 42, 43, 44]. Two temporal consistency states have been defined by them. [17] argues that in grid workflow management systems, two states are too restricted as grid workflow execution environments are very dynamic. Intermediate states exist and different exception handlings should be triggered to handle them [45]. Accordingly, [17] has introduced four temporal consistency states and developed corresponding verification algorithms. However, it does not investigate how different exception handlings should be triggered to handle different states.

In many complex scientific and business processes such as a climate modelling process, users often only set a few coarse-grained temporal constraints rather than a large number of them [1, 3]. However, only a few coarse-grained temporal constraints are not sufficient to control grid workflow execution in terms of time as we are not able to control grid workflow execution locally at various activity points. Therefore, we need to investigate how to assign a number of fine-grained temporal constraints based on a few user-defined coarse-grained ones. To do so, we need to solve two problems. One is where to assign some fine-grained temporal constraints. The other is how to assign them. [46] has solved the second problem. Further research is needed for solving the first problem.

At build-time and run-time instantiation stages, temporal verification is static because there are no any specific execution times. Each temporal constraint needs to be verified only once with the consideration of all covered activities. Therefore, we need not decide at which activities we should conduct the verification. At run-time execution stage however, activity completion durations vary and consequently, we may need to verify each temporal constraint many times at different activities. However, conducting the verification at every activity is not efficient as we may not need to do so at some activities such as those that can be completed within allowed time intervals. Hence, we need to figure out where to conduct temporal verification. The activities at which we conduct temporal verification are called *checkpoints* [29, 30, 43]. Correspondingly, a research topic comes into the picture which is Checkpoint Selection Strategies (CSS). Currently, some checkpoint selection strategies have been proposed [29, 31, 32, 33, 34, 43]. Some of them often select some unnecessary checkpoints or ignore some necessary ones. Although [34] develops a strategy which can select necessary yet sufficient checkpoints, it does not operate in the whole context of temporal verification. For example, it does not take into consideration temporal dependency between temporal constraints. With temporal dependency, we may be able to select fewer checkpoints. Hence, further research is needed.

Once we know where to verify temporal constraints, we can start to do it. Some

conventional temporal verification work has been done such as [29, 44]. Some other work is related to temporal verification from other perspectives such as time QoS scheduling, project management, etc., [14, 35, 47, 48, 49]. They try to apply the technologies from their fields to grid workflow temporal verification. However, it might not be consistent with grid workflow management systems very well because different fields have different characteristics. For example, a project management system normally only has one instance at run-time, while a grid workflow management system normally has a number of grid workflow instances at run-time. So, the project management temporal analysis is relatively static and hence cannot be equally applied to more dynamic grid workflow management system environments. Another example is about time QoS scheduling. Time QoS scheduling is important for ensuring that grid workflow execution is going smoothly in terms of time [14, 35, 50, 51]. However, time QoS scheduling cannot be used to judge whether a grid workflow execution is definitely going wrong when the scheduling assignment is not followed. Hence, grid workflow specific temporal verification is needed. For example, as analysed above, a grid workflow normally needs a series of temporal constraints. When a series of temporal constraints are verified, they are often dependent on each other in terms of overall temporal verification effectiveness and efficiency. This is because the later verification may make the previous verification ineffective and also the later verification may utilise the previous verification results to save current verification computation for better efficiency. Therefore, temporal dependency between temporal constraints must be taken into consideration when temporal verification is conducted so that we can further improve overall temporal verification effectiveness and efficiency. [30] has discussed temporal dependency based on conventional two consistency states. Further research is needed for temporal dependency based on multiple temporal consistency states which, according to [17], allow for grid workflow specific characteristics.

2.2. Grid workflow validation

Specifically speaking, based on [19], grid workflow validation is mainly concerned with the consistency between complex scientific and business processes and grid workflow specifications. When we model or redesign a complex scientific or business process as a grid workflow specification based on models and constructs provided by the selected grid workflow management system, we must ensure that all complex scientific and business process requirements are modelled or redesigned in the grid workflow specification. Otherwise, the grid workflow specification is incomplete and incorrect from the perspective of user requirements and needs. For example, some grid workflow management systems or architectures cannot support temporal constraint modelling [38]. Then, for those time-critical scientific and business processes such as climate modelling processes for weather forecast, the corresponding grid workflow specifications are incomplete or even incorrect as temporal information will be ignored. In fact, different grid workflow management systems provide different basic models and constructs for complex scientific and business process modelling. And different models and constructs have different expressive power [38]. A complex scientific or business process which can be modelled by one grid workflow management system may not be completely modelled by another one. Therefore, a question is raised between complex

scientific and business processes and grid workflow modelling power. That is whether current complex scientific and business process requirements can be supported by the selected grid workflow management system. This is particularly important when e-scientists or e-business people develop or purchase corresponding grid workflow management system products to support their complex scientific and business processes. To answer this question, grid workflow validation must be conducted. We need to describe a complex scientific or business process as a proper representation. Then, we represent the expressive power of the selected grid workflow management system. Finally, we need to develop some efficient approaches to validate the consistency between them. Hence, as shown in Figure 3, the grid workflow validation taxonomy consists of three elements: (a) representation of complex scientific and business processes, (b) representation of expressive power of grid workflow management systems, and (c) validation approaches.

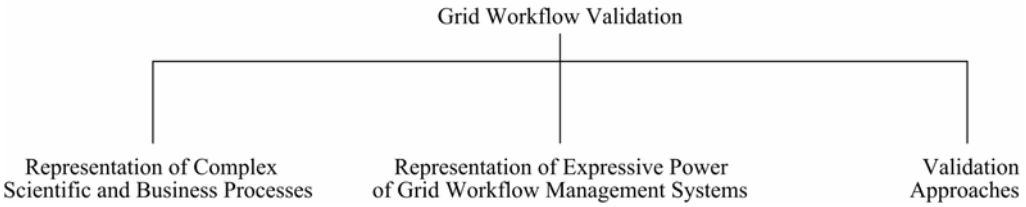


Figure 3. Elements of grid workflow validation

2.2.1. Representation of complex scientific and business processes

To check the consistency between complex scientific and business processes and corresponding grid workflow specifications, firstly we need to capture and represent complex scientific and business processes.

Some methods such as Petri Net or process algebra (mainly pi-Calculus) might be used to represent complex scientific and business processes [20, 21, 52, 53]. However, a complex scientific or business process normally changes often [6, 7, 38]. Changes in process requirements may lead to significant state adjustments in the Petri Net based representation or significant equation adjustments in the process algebra based representation [54]. Some other models such as state machine or concurrent transaction logic also have similar problems [55]. How to extend the expressive ability of Petri Net, process algebra or some other models to allow for the complexity and dynamics of sophisticated scientific and business processes needs further investigation.

2.2.2. Representation of expressive power of grid workflow management systems

By representing the expressive power of a grid workflow management system, we can get to know how complicated scientific and business process requirements can be supported by it [22]. Currently, most grid workflow management systems support four basic control structures:

parallel, selective, sequential and iterative [9, 11, 38]. However, it is not enough for us to only use these four control structures to represent the grid workflow expressive power. On one hand, a grid workflow management system may support other control structures. On the other hand, the diversity of complex scientific and business process requirements is calling for new control structures. Therefore, we need to tackle how to extend the control structures to allow for profound scientific and business process requirements and how to use them to evaluate the expressive power of the selected workflow system. Although [22] presents twenty workflow patterns, they are mainly for business processes. For scientific processes, the requirements are often different because scientific processes are normally computation or data intensive and lack intensive dependency between different activities [7, 9]. It is still a problem whether we can equally apply the twenty patterns to complex scientific processes. And if not, how to extend or modify them to allow for complex scientific requirements would be open.

2.2.3. Validation approaches

Based on the representation of complex scientific and business processes and based on the expressive power of grid workflow management systems, we can fall to developing some validation approaches. These approaches can be used to check whether all scientific and business process requirements can be supported by the selected grid workflow management system based on its control structures. Here, a key challenge is how to map the representation of a complex scientific or business process based on Petri Net or process algebra or some other models to the control structures. If the approaches can conduct the mapping, then if some part of the representation has no corresponding control structures provided by the selected grid workflow management system, it means that we can not correctly capture the scientific or business process as a grid workflow specification. In other words, the grid workflow validation fails. Then, we may consider changing the grid workflow management system to another one which has stronger expressive power.

3. SURVEY AND FURTHER DISCUSSION

Currently, very few projects focus on grid workflow verification and validation. This is not surprising as grid workflow has been investigating for just a few years. Many efforts have been made on the research and development of functional issues such as modelling languages, grid workflow engines and so on [9, 11, 13, 38]. Gradually, more and more attention has been paid to QoS scheduling and overall management [35, 51, 56]. However, very little work has been done on grid workflow verification and validation which is an important non-functional issue. More or less, the very few projects related to grid workflow verification and validation mainly include DILIGENT (Digital Library Infrastructure on Grid Enabled Technology) [57], CROWN (China R&D environment Over Wide-area Network) [58], Discovery Net [59], SwinDeW-G (Swinburne Decentralised Workflow for Grid) [60] and CAT (Composition Analysis Tool) [61]. We show the comparison of these projects based on grid workflow verification taxonomy in Table 1, and based on grid workflow validation taxonomy in Table 2.

Table 1. Grid Workflow Verification Taxonomy Mapping

Project Name	Structure	Performance	Resource	Authorisation	Cost	Temporal
DILIGENT	Supported	Supported	N/A	N/A	N/A	N/A
CROWN	Partly Supported	N/A	N/A	N/A	N/A	N/A
Discovery Net	Supported	N/A	N/A	N/A	N/A	N/A
SwinDeW-G	N/A	N/A	N/A	N/A	N/A	Supported
CAT	Supported	N/A	N/A	N/A	N/A	N/A

Table 2. Grid Workflow Validation Taxonomy Mapping

Project Name	Representation of Process	Representation of Expressive Power	Validation Approaches
DILIGENT	N/A	N/A	N/A
CROWN	N/A	Supported	N/A
Discovery Net	N/A	N/A	N/A
SwinDeW-G	N/A	N/A	N/A
CAT	N/A	N/A	N/A

We now describe those projects briefly.

DILIGENT [57] aims at supporting this new research operational mode by providing a knowledge infrastructure that manages a network of shared resources (e.g., archives, database and software tools) and enables the creation of on-demand digital libraries. It supports process design by composing existing services. Performance and structure verification based on the process description are supported.

CROWN [58] is a grid toolkit developed by BeiHang University in China. It contains a grid workflow modeling language called GPEL (Grid Process Execution Language). Based on GPEL, CROWN supports part of structure verification to identify deadlock and loss of synchronisation. Besides, it also analyses the expressive power of GPEL. This is part of grid workflow validation.

Discovery Net project [59] is an EPSRC-funded (Engineering and Physical Sciences Research Council) project to build the world's first e-Science platform for scientific discovery from the data generated by a wide variety of high throughput devices at Imperial College London. Its workflow management supports structure verification.

SwinDeW-G [60] is a peer-to-peer based grid workflow management system. For the moment, it can only support grid workflow temporal verification. It is on the way to developing approaches for other types of verification and validation.

CAT [61] developed by University of Southern California is a tool that analyses grid workflows and generates error messages and suggestions in order to help users compose complete and consistent workflows. CAT supports structure verification.

From Table 1, we can see that each project can support at least one type of grid workflow

verification, but not all types. From Table 2, we can see that most of the projects do not support validation. Given that many grid workflow management systems have been developed and can well support functional issues such as modelling languages or execution engines [38], more efforts should now be devoted to grid workflow verification and validation so that we can develop a set of complete verification and validation approaches. Accordingly, we are able to check the correctness of grid workflow specification and execution. This correctness is important as it directly affects the applicability of a grid workflow management system.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we have systematically analysed grid workflow verification and validation, and classified their key issues into a taxonomy. Some of them are not investigated by current research such as mapping validation approaches or cost verification, while others may need further investigation. The taxonomy depicts a picture and provides some insights into further potential research points for grid workflow verification and validation. Our ongoing and future work is to investigate such points, develop and evaluate corresponding verification and validation approaches.

ACKNOWLEDGEMENTS

A preliminary version of this paper appeared in AusGrid2006 [62].

REFERENCES

1. Abramson D, Kommineni J, McGregor JL, Katzfey J. An Atmospheric Sciences Workflow and Its Implementation with Web Services. *Proceedings of the 4th International Conference on Computational Science, Part Ⅰ (Lecture Notes in Computer Science*, vol: 3036), Springer Verlag: Berlin, Krakow, Poland, June 2004; 164-173.
2. Buyya, R., Enugopal, S.V. 2004. The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report. Technical Report, GRIDS-TR-2004-2, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, http://www.gridbus.org/papers/gridbus2004.pdf, accessed on Feb. 1, 2007.
3. Buyya, R., Abramson, D., Venugopal, S. 2005. The Grid Economy. *Proceedings of The IEEE* 2005; 93(3): 698-714.
4. Foster, I., Kesselman, C., Tuecke, S. 2002. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of Supercomputing Applications* 2002; 15(3): 200-222.
5. Foster I, Kesselman C, Nick J, Tuecke S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Globus Project, 2002, http://www.globus.org/alliance/publications/papers/ogsa.pdf, accessed on Feb. 1, 2007.
6. Simpson DR, Kelly N, Jithesh PV, Donachy P, Harmer TJ, Perrott RH, Johnston J, Kerr P, McCurley M, McKee S. GeneGrid: A Practical Workflow Implementation for a Grid Based Virtual Bioinformatics Laboratory. *Proceedings of the UK e-Science All Hands Meeting 2004 (AHM04)*, UK Engineering and Physical Science Research Council, Swindon, UK: Nottingham, UK, Sept. 2004; 547-554.

7. Deelman E, Blythe J, Gil Y, Kesselman C, Mehta G, Vahi K. 2003. Mapping Abstract Complex Workflows onto Grid Environments. *Journal of Grid Computing* 2003; 1(1): 9-23.
8. Marinescu, D. A Grid Workflow Management Architecture. Global Grid Forum White Paper, 2002, http://www.gridforum.org/mail_archive/gce-wg/2002/Archive/pdf00003.pdf, accessed on Feb. 1, 2007.
9. Cybok, D. 2006. A Grid Workflow Infrastructure. Concurrency and Computation: Practice and Experience, Special Issue on Workflow in Grid Systems 2006; 18(10): 1243-1254.
10. Huang Y. 2003. JISGA: A JINI-BASED Service-Oriented Grid Architecture. *The International Journal of High Performance Computing Applications* 2003; 17(3): 317-327.
11. Krishnan S, Wagstrom P, Laszewski GV. GSFL: A Workflow Framework for Grid Services. Technical Report, Argonne National Laboratory, Argonne, U.S.A., 2002, http://www-unix.globus.org/cog/papers/gsfl-paper.pdf, accessed on Feb. 1, 2007.
12. Workflow Management Coalition: The Workflow Reference Model. TC00-1003, 1995.
13. Fahringer T, Pllana S, Villazon A. A-GWL: Abstract Grid Workflow Language. *Proceedings of the 4th International Conference on Computational Science, Part ⅢⅠ (Lecture Notes in Computer Science*, vol: 3038), Springer-Verlag: Berlin, Krakow, Poland, June 2004; 42-49.
14. Yu, J., Buyya, R. A Novel Architecture for Realizing Grid Workflow Using Tuple Spaces. *Proceedings of 5th IEEE/ACM International Workshop on Grid Computing (GRID'04) - Volume 00***,** IEEE Computer Society Press, Washington, DC, USA: Los Alamitos, USA, Nov. 2004; 119-128.
15. Amin K, Laszewski GV, Hategan M, Zaluzec NJ, Hampton S, Rossi A. GridAnt: A Client-controllable Grid Workflow. *Proceeding of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*, IEEE Computer Society Press, Washington, DC, USA: Hawaii, Jan. 2004; 210-219.
16. Cao, J., Jarvis, S.A., Saini, S., Nudd, G.R. GridFlow: Workflow Management for Grid Computing. *Proceedings of IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003)*, IEEE Computer Society Press, New York, USA, Tokyo, May 2003; 198-205.
17. Chen, J, Yang, Y. 2006. Multiple States based Temporal Consistency for Dynamic Verification of Fixed-time Constraints in Grid Workflow Systems. *Concurrency and Computation: Practice and Experience* 2006; in press, http://www3.interscience. wiley.com/cgi-bin/fulltext/113374790/PDFSTART, accessed on Feb. 1, 2007.
18. IEEE. Standard Glossary of Software Engineering Terminology. *In IEEE Software Engineering Standards Collection*, IEEE, Std 610.12-190, 1994.
19. IEEE-SA Standards Board. *IEEE Standard for Software Verification and Validation*, IEEE Std 1012, 1998.
20. Aalst, van der, W.M.P. 1998. The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems and Computers* 1998; 8(1): 21–66.
21. Adam, N., Aturi, V. and Huang, W. 1998. Modeling and Analysis of Workflows Using Petri Nets. *Journal of Intelligent Information Systems, Special Issue on Workflow and Process Management* 1998; 10(2): 131-158.
22. Aalst, van der, W.M.P., Hofstede, A.H.M.Ter, Kiepuszewski, B. and Barros, A.P. 2003. Workflow Patterns. *Distributed and Parallel Databases* 2003; 14(3): 5-51.
23. Van der Aalst WMP. Workflow Verification: Finding Control-Flow Errors using Petri-net based Techniques. *Proceedings of the International Conference on Business Process Management: Models, Techniques, and Empirical Studies (Lecture Notes in Computer Science*, vol. 1806), Springer-Verlag: Berlin, Berlin, Germany, 2000; 161-183.
24. Aalst, van der, W.M.P. 2002. Workflow Management, Models, Methods, and Systems. Cambridge, Massachusetts, London, England, The MIT Press.
25. Li, H., Yang, Y., Chen, T.Y. 2004. Resource Constraints Analysis of Workflow Specifications. *The Journal of Systems and Software* 2004; 73(2): 271-285.
26. Li, H., Yang, Y. 2005. Dynamic Checking of Temporal Constraints for Concurrent Workflows. *Electronic Commerce Research and Applications* 2005; 4(2): 124-142.
27. Bertino, E., Ferrari, E., Atluri, V. 1999. An Authorisation Model for Supporting the Specification and Enforcement of Authorisation Constraints in Workflow Management Systems. *ACM Transactions on Information System Security* 1999; 2(1): 65-104.
28. Wu, S., Sheth, A., Luo, Z. 2002. Authorisation and Access Control of Application Data in Workflow Systems. *Journal of Intelligent Information Systems* 2002; 18(1): 71-94.
29. Marjanovic O, Orlowska ME. 1999. On Modelling and Verification of Temporal Constraints in Production Workflows. *Knowledge and Information Systems* 1999; 1(2): 157-192.

30. Chen, J., Yang, Y. Temporal Dependency for Dynamic Verification of Fixed-date Constraints in Grid Workflow Systems. *Proceedings of the 7th Asia Pacific Web Conference (Lecture Notes in Computer Science, vol. 3399)*, Springer-Verlag: Berlin, Berlin, Germany, Mar. 2005; 820-831.
31. Chen, J., Yang, Y., Chen, T.Y. Dynamic Verification of Temporal Constraints on-the-fly for Workflow Systems. *Proceedings of Asia-Pacific Software Engineering Conference*, IEEE CS Press, New York, USA, Korea, Nov./Dec. 2004; 30-37.
32. Chen, J, Yang, Y. 2006. Activity Completion Duration based Checkpoint Selection for Dynamic Verification of Temporal Constraints in Grid Workflow Systems. *The International Journal of High Performance Computing Applications* 2006; Sage, in press, http://www.ict.swin.edu.au/personal/jchen/docs/Chen_Yang_IJHPCA.pdf, accessed on Feb. 1, 2007.
33. Chen J, Yang Y. A Minimum Proportional Time Redundancy based Checkpoint Selection Strategy for Dynamic Verification of Fixed-time Constraints in Grid Workflow Systems. *Proceedings of the 12th Asia Pacific Software Engineering Conference (APSEC2005)*, Dec. 2005. IEEE CS Press: Taiwan, Dec. 2005; 299-306.
34. Chen, J., Yang, Y. Selecting Necessary and Sufficient Checkpoints for Dynamic Verification of Fixed-time Constraints in Grid Workflow Systems. *Proceedings of the 4th International Conference on Business Process Management (BPM2006)* (*Lecture Notes in Computer Science*, Vol: 4102), Springer-Verlag: Berlin, Vienna, Austria, Sept. 2006; 445-450.
35. Yu, J., Buyya, R., Tham, C.K. QoS-based Scheduling of Workflow Applications on Service Grids. *Proceedings of 1st IEEE International Conference on e-Science and Grid Computing (e-Science2005)*, IEEE Computer Society Press, Washington, DC, USA: Melbourne, Australia, Dec. 2005; 140-147.
36. Sadiq, W., Orlowska, M.E. 2000. Analysing Process Models using Graph Reduction Techniques. *Information Systems* 2000; 25(2): 117-134.
37. Son, J.H., Kim, M.H. 2001. Improving the Performance of Time-constrained Workflow Processing. *The Journal of Systems and Software* (2001); 58(3): 211-219.
38. Yu, J., Buyya, R. 2005. A Taxonomy of Scientific Workflow Systems for Grid Computing. *Special Issue on Scientific Workflows, SIGMOD Record* 2005; 34(3): 44-49.
39. Fabio, C., Silvana, C., Mariagrazia, F. 2001. Managing Workflow Authorisation Constraints through Active Database Technology. *Information Systems Frontiers* 2001; 3(3): 319-338.
40. Chinn, S., Madey, G. 2000. Temporal Representation and Reasoning for Workflow in Engineering Design Change Review. *IEEE Transactions on Engineering Management* 2000; 47(4): 485-492.
41. Li, H., Fan, Y. 2004. Workflow Model Analysis Based on Time Constraint Petri Nets. *Journal of Software* 2004; 15(1): 17-26.
42. Li, J., Fan, Y., Zhou, M. 2003. Timing Constraint Workflow Nets for Workflow Analysis. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans* 2003; 33(2): 179-193.
43. Zhuge, H., Cheung, T., Pung, H. 2001. A Timed Workflow Process Model. *The Journal of Systems and Software* 2001; 55(3): 231-243.
44. Eder J, Panagos E, Rabinovich M. Time Constraints in Workflow Systems. *Proceedings of the 11th International Conference on Advanced Information Systems Engineering (CAiSE'99) (Lecture Notes in Computer Science*, vol: 1626), Springer-Verlag: Berlin, Heidelberg, Germany, June 1999; 286-300.
45. Hagen C, Alonso G. 2000. Exception Handling in Workflow Management Systems. *IEEE Transactions on Software Engineering* 2000; 26(10): 943-958.
46. Chen, J., Yang, Y. Dynamic Setting, Verification and Adjustment of Upper Bound Constraints in Grid Workflow Systems. *Proceedings of 2nd International Conference on Semantics, Knowledge and Grid (SKG2006)*, IEEE Computer Society Press, Washington, DC, USA: Guilin, IEEE digital library, China, Oct./Nov. 2006.
47. Bussler, C. Workflow Instance Scheduling with Project Management Tools. *Proceedings of Workshop on Database and Expert Systems Applications (DEXA'98) (Lecture Notes in Computer Science*, vol. 1460), Springer-Verlag: Berlin, Berlin, Germany, Aug. 1998; 753-758.
48. Cardoso, J., Sheth, A., Miller, J. 2002. Quality of Service and Semantic Composition of Workflows. *Ph.D. Thesis*, University of Georgia, Athens, Georgia.
49. Pozewaunig, H., Eder, J., Liebhart, W. ePERT: Extending PERT for Workflow Systems. *Proceedings of EastEuropean Symposium on Advances in Database and Information Systems (ADBIS'97)*, St. Petersburg, Russia, Sept. 1997; 217-224.

50. Brandic I, Benkner S, Engelbrecht G, Schmidt R. Towards Quality of Service Support for Grid Workflows. *Proceedings of the European Grid Conference 2005 (EGC2005) (Lecture Notes in Computer Science*, vol: 3470), Springer-Verlag: Berlin, Amsterdam, The Netherlands, Feb. 2005; 661-670.
51. Brandic, I., Benkner, S., Engelbrecht, G., Schmidt, R. QoS Support for Time-Critical Grid Workflow Applications. *Proceedings of 1st IEEE International Conference on e-Science and Grid Computing (e-Science2005)*, IEEE Computer Society Press, Washington, DC, USA: Melbourne, Australia, Dec. 2005; 108-115.
52. Hennessy, M. 1998. Algebraic theory of processes. Series in the Foundations of Computing, MA, USA, Mit Press, Cambridge.
53. Sangiorgi, D., Walker, D. 2004. The Pi-Calculus: A Theory of Mobile Processes, England, Cambridge Uni. Press.
54. Amit, K.C., Munindar, P.S. Commitments for Flexible Business Processes. *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS2004*), IEEE Computer Society Press, New York, USA, Aug. 2004; 1362-1363.
55. Davulcu, H., Kifer, M., Ramakrishnan, C.R., Ramakrishnan, I.V. Logic based modelling and analysis of workflows. *Proceedings of ACM Symposium on Principles of Database Systems*, ACM Press New York, June 1998; 25-33.
56. Al-Ali, R., Amin, K., Laszewski, G.V., Rana, O., Walker, D., Hategan, M., and Zaluzec, N. Analysis and Provision of QoS for Distributed Grid Applications. *Journal of Grid Computing* 2004; 2(2): 163-182.
57. DELIGENT. http://www.diligentproject.org/.
58. CROWN Team. CROWN portal, http://www.crown.org.cn/en/.
59. Discovery Net project. http://www.discovery-on-the.net/.
60. SWINDEW-G Team. System Architecture of SwinDeW-G. http://www.ict.swin.edu.au/personal/jchen/SwinDeW-G/System_Architecture.pdf.
61. Kim, J., Spraragen, M., Gil, Y. An Intelligent Assistant for Interactive Workflow Composition. *Proceedings of 9th International Conference on Intelligent User Interface*, ACM Press New York, Jan. 2004; 125 – 131.
62. Chen, J., Yang, Y. Key Research Issues in Grid Workflow Verification and Validation. *Proceedings of 4th Australasian Symposium on Grid Computing and e-Research (AusGrid 2006)*, ACSW Frontiers 2006, Australian Computer Science Communications, Hobart, Australia, Jan. 2006; 28(7): 97-104.