A Novel Approach to User-steering in Scientific Workflows

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

***Abstract*— From the scientist's perspective the workflow execution is like black boxes [25]. The scientist submits the workflow and at the end, the result or a notification about failed completion is returned. Concerning long running experiments or when workflows are in experimental phase it may not be acceptable. Scientist may need to fine-tune and monitor their experiments. To support the scientist with special user interaction tool we introduced intervention points (iPoints) where the user takes over the control for a while and has the possibility to interfere, namely to change some parameters or data, or to stop, to restart the workflow or even to deviate from the original workflow model during enactment. We plan to implement our solution in IWIR [9] language which was targeted to provide interoperability between four existing well-known SWfMS within the framework of the SHIWA project [1].**

I. Introduction

Supporting scientists in complex computational processes with dynamic workflow execution on distributed and parallel computing infrastructures imposes a challenge, not only because of the wide range of failures that could arise during enactment, but because of the complexity of the scientific workflows and the dynamically changing environment. There are numerous failures that can result in failed execution, because the administrator cannot interfere in time.

Furthermore due to the exploratory nature of scientific workflows it may be necessary to change the workflow model or to adapt the execution to intermediary results. Fine-tuning the parameters during the execution is a critical issue concerning long running, data and compute intensive scientific processes. All these problems can be accomplished by giving the user or the administrator the possibility to interfere with workflow execution (user-steering).

During the past decade there have been emerged a variety of Scientific Workflow Management Systems (Moteur [4], Triana [2], Taverna [5], Askalon [3], P-Grade [6], etc.). All of them have their own scientific community and thus their own scientific disciplines. Therefore they all have their own enactment system, workflow language and provenance manager. The SHIWA project [1] (2010-2012) was targeted to promote interoperability between different workflow systems by applying both coarse- and fine-grained strategies. The coarse-grained strategy treats each workflow engine as distributed black boxes, where data being sent to preexisting enactment engines and results are returned. One workflow system is able to invoke another workflow engine through the use of the SHIWA interface, and the Shiwa Portal facilitates the publishing and sharing of reusable workflows [7]. The fine-grained approach [8] deals with language interoperability by defining and Interoperable Workflow Intermediate Representation (IWIR) [9] language for translating workflows (ASKALON, P-Grade, MOTEUR and Triana) from one DCI to another, thus creating a cross-compiler for workflows.

In our work we would like to give an intermediary solution for interoperability between different workflow systems that should handle the dynamic behavior of workflow execution. This dynamism does not have to be implemented in each workflow management system. The implementation would be agnostic of any workflow management system, and could be attached to any IWIR compatible system.

Our paper is organized as follows. Chapter 2 gives a brief summary about the state of the art. Chapter 3 represents our new proposal and finally we conclude our work with a brief previsioning of possible future research directions.

II. Related Works

In the literature there exist several solutions to support dynamism at different granularity [12]. In one of our earlier work [10] we have classified the dynamic tools of scientific workflow management systems (SWfMS) at different phases of the workflow lifecycle and at different levels, namely system level, task level and workflow level. These features include some kind of language support [11], advance and late modelling techniques [13], [14], incremental compilation techniques [17], [18], [19], [20], dynamic resource allocation [15], workflow partitioning [21] and flexible data management [16]. Obviously most of them relates on monitoring the workflow execution, or the state of the computing resources.

From the scientist perspective monitoring is also very important, but data analyses and dynamic interference is also an emerging need concerning nowadays scientific workflows [26]. Due to their exploratory nature they need control and intervention from the scientist to conserve energy and time. There are several systems that support dynamic intervention such as stopping, or re-executing jobs or even the whole workflow but there is an increasing need to have more sophisticated manipulation possibility.

Vahi et al. [22] introduced Stampede, a monitoring infrastructure that was integrated in the Pegasus and Triana SWfMS. The main target was to provide generic real-time monitoring across multiple SWfMS. The results proved that Stampede was able to monitor workflow executions across heterogeneous SWfMS but it required the scientists to follow the execution from a workstation. This solution may be tiring in long-term executions. To tackle this, it is possible to pre-program triggers, such as proposed by Missier et al. [27], to check for problems in the workflow and to alert the scientist. In their other paper they worked out sciLightning [24], a system that is able to notify the scientist upon completion of certain, predefined events.

The above mentioned monitoring systems do not intend to prevent or solve unnecessary failed termination of workflow running, or user intervention. They mainly focus on the better optimization or scheduling of the jobs. One of the biggest challenge of the dynamic behavior is runtime workflow manipulation and workflow control. Mattoso et al. summarized the state-of-the-art and possible future directions and challenges [12]. They found that lack of support in user steering is one of the most critical issues that the scientific community has to face with. In their other work [23] they managed to implement dynamic parameter sweep workflow execution where the user has the possibility to interfere with the execution and change the parameter of some filtering criteria without stopping the workflow. However most of the existing solutions the opportunity of changes are limited and they do not solve on-the-fly modification of parameter sets, data sets or the model itself.

III. Workflow Manipulation

Due to the exploratory nature of scientific workflows and during „fine-tuning" phase it may be necessary to change the workflow model or to adapt the execution to intermediary or even to historical results. It can be accomplished by giving the user the possibility to interfere with workflow execution (user-steering). On the other hand it can be facilitated by providing proper language tools (query partial results, dynamic programming structures, time management, etc.). In a dynamic system a user already at composition phase should be given the opportunity to design more execution possibilities depending on some conditions or to schedule time management functions. Existing workflow management systems more or less provide some kind of dynamic language support, for example the if-then-else, while, for, foreach structures while other systems with basic language constructs enable the use of embedded workflows (for example: gUSE).

In our paper we propose a new, dynamic workflow control mechanism based on Intervention Points (iPoints) to enable dynamic and user-steered workflow execution, which is able to modify the execution according to intermediary results and to adapt it to environmental changes. With the use of iPoints during enactment the user can take over the control for a while and has the opportunity to restart and stop the workflow execution, to insert time management functions, or based on runtime intermediary data the user can change certain parameters, filtering criteria or the input dataset. Furthermore with the insertion of a checkpoint the user can also change the execution model of the running workflow.

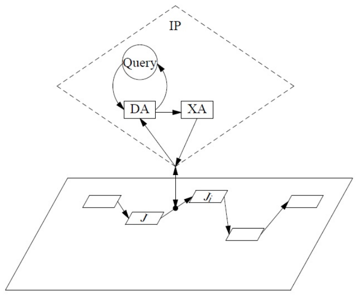

Figure 1. Structure of an iPoint

A. *Structure of an iPoint*

An iPoint is somewhat similar to a meta-workflow or sub-workflow. It is located out of the plane of the workflow. In fig.1 the big rectangle with solid line represents the workflow plane, where the small rectangles on it represent the jobs and the sequential or parallel mesh of the user defined jobs form the scientific workflow. The iPoint which can be imagined or handled as a special job jumps out from this plane. It contains series of steps that are not part of the computational tasks (bordered with dashed line in fig.1) defined by the scientist rather which can affect the real execution (analyses of data or controlling functions, etc.). During the execution of an iPoint first a designator action (DA) is performed. The DA designates or determines what changes are necessary during the execution. It can be one of the following actions: intermediary data query, starting a timer, to stop a timer, resetting the timer, checking the timer and an alarm request.

If DA is a query an input reply is returned and based on this reply an eXecutable Action (XA) is performed. This XA can be one of the following possibilities: modifying the workflow model with checkpoint request, restarting or stopping the workflow execution, changing certain parameters, filtering criteria or input dataset or requesting a checkpoint. Of course, the scientist has the possibility to perform more queries.

B. *Types of iPoints*

We have differentiated three types of ipoints:

Closed iPoint: When the conditions and the changes are also known before execution, then the user can design the location and the function of the iPoints during the composition phase. He can also define the proper queries for intermediary results, or the time management functions and depending on the results the action (XA) that should be carried out. So in this case the user do not have to interfere, all the actions are clearly defined, that is why we call it closed iPoint.

Open iPoint: When the conditions or the changes are not known before execution, because they can only be specified depending on the query results, then the user should only determine the places where the intervention should take place. With these iPoints the user can interrupt the workflow execution for a while. During this predefined time interval the user can decide what DA to take, and depending on the results what XA to perform next.

Ad-hoc iPoint: When the demand for interfering arises only during execution upon some kind of external effect (for example computational or other failure, or because from an input sensor unwanted data arrives or simply because the user need some intervention) which cannot be foreseen before execution. In this case by using a special flag the system could stop the workflow execution at the nearest possible place, it should perform a checkpoint (if necessary) and give the control to the user for a while. At this point the user (like in the previous cases) may perform some query for intermediary results and depending on the outcome of the query certain XA can be carried out.

C. The Placement of iPoints

The iPoints can be scheduled upon four various events: after data arrival at input port, after data arriving at output port (fig. 2a), before job starting, after job completion (fig. 2b). However a trigger event of an iPoint also can be: timer expiration, external effect, failure message arrival, timed alarm. In this case the placement happens in an ad-hoc manner at the nearest possible moment during the execution.

D. iPoint Language Support

A dynamic system that supports user intervention must provide the user with language tools to define intervention points along with XA actions. In this case the iPoint is a if Query = X then „Action1 else Action2" statement or time management function.

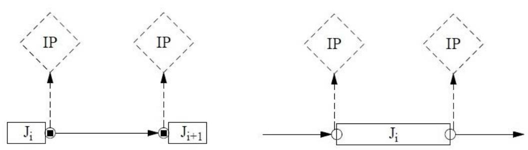

Figure 2. placement of iPoints concerning : a) data-flow oriented workflows b) control-flow driven workflows.

At abstract workflow level the iPoint is a special job, which can be visualized with a pentagon or hexagon (fig. 3 and 4) The figures show the course of the execution steps involving an iPoint. In fig. 3 the iPoint performs some kind of partial data analyses, and depending on the result the workflow can be stopped, restarted the execution model can deviate from the original model or even a checkpoint can be performed. In fig. 4 time management functions are inserted into the model.

E. Implementation

The above introduced iPoints are planned to be an extension of the IWIR language (Interoperable Workflow Intermediate Representation) [9], which was developed within the frame of the SHIWA project. It will gain the advantage of being implemented in an intermediate language for IWIR compatible systems. The implementation of the iPoint can be realized with a Scientific Workflow Manager independent module that handles the actions taking place during the interventions.
This module takes over the control of the workflow while the actions defined in the given iPoints are executed. This module can be an extension of an existing workflow management system, or a completely new system. In this latter case there is no need to change the existing SWfMS, only an interface should be specified and implemented.

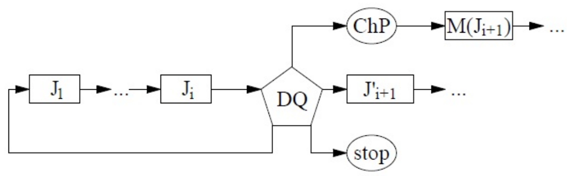

Figure 3. Courses of execution steps involving partial data analyses

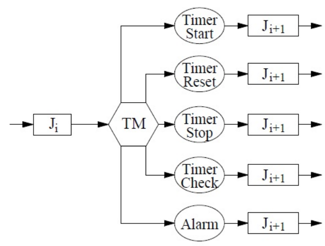

Figure 4. Courses of execution steps involving time management functions

IV. Conclusions and Future Work

We have proposed a new dynamic workflow control mechanism based on Intervention Points (iPoints). With the help of the introduced intervention points and system monitoring adaptive and user steered execution can be realized at different level. Furthermore when the system supports (runtime) provenance analyses, with the help of these iPoints provenance based, adaptive execution can be realized. Originally the iPoints were planned to solve the problem of user-steering, but also the administrator can benefit from the use of iPoints. The administrator can also insert them to realize provenance based adaptive fault recovery or even system optimization tasks. The analysis of provenance data can be carried out during and after execution. Our future work is to work out the mentioned usage fields in detail.

References

[1] www.shiwa-workflow.eu/project

[2] The Triana Project, http://www.trianacode.org

[3] The Askalon Project, http://www.askalon.org

[4] The moteur Project, http://www.shiwa-workflow.eu

[5] The Taverna Project, http://www.taverna.org.uk , 2009.

[6] Kacsuk, P. (2011), P-GRADE portal family for grid infrastructures. Concurrency Computat.: Pract. Exper., 23: 235–245. doi: 10.1002/cpe.1654

[7] Terstyánszky G., Kukla T., Kiss T., Kacsuk P., Balaskó Á., Farkas Z.: Enabling scientific workflow sharing through coarse-grained interoperability. Future Generation Computer Systems-The International Journal Of Grid Comput, vol. 37. pp. 46-59, 2014. ISSN 0167-739X, 2014 10.1016/j.future.2014.02.016

[8] Plankensteiner K., Prodan R., Janetschek M., Fahringer T., Montagnat J., Rogers D., Harvey I., Taylor I., Balaskó Á., Kacsuk P.: Fine-Grain Interoperability of Scientific Workflows in Distributed Computing Infrastructures. Journal of Grid Computing, vol. 3 pp. 429–55, 2013. doi:10.1007/s10723-013-9261-8.

[9] Plankensteiner K., Prodan R., Fahringer T., Montagnat J., Harrison A., Glatard T., Hermann G., Kozlovszky M.: IWIR Specification v1. 1, 2011. https://www.shiwa-workflow.eu/documents/10753/55350/IWIR+v1.1+Specification

[10] E. Kail, A. Banati, K. Karoczkai, P. Kacsuk, és M. Kozlovszky. „Dynamic workflow support in gUSE". In Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on, 354–59. IEEE, 2014. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6859590.

[11] E. M. Bahsi, Dynamic Workflow Management For Large Scale Scientific Applications, PhD Thesis, B.S., Fatih University, 2006.

[12] Mattoso, Marta, Kary Ocaña, Felipe Horta, Jonas Dias, Eduardo Ogasawara, Vitor Silva, Daniel de Oliveira, Flavio Costa, és Igor Araújo. „User-Steering of HPC Workflows: State-of-the-Art and Future Directions", 1–6. ACM Press, 2013. doi:10.1145/2499896.2499900.

[13] P. Heinl, S. Horn, S. Jablonski, J. Neeb, K. Stein, and M. Teschke. A Comprehensive Approach to Flexibility in Workflow Management Systems. In Proceedings of the International Joint Conference on Work Activities Coordination and Collaboration (WACC '99), pages 79–88. ACM Press, New York, NY, USA, 1999.

[14] M. Pesic: Constraint-Based Workflow Management Systems: Shifting Control to Users, PhD thesis, Eindhoven: Technische Universiteit Eindhoven, 2008

[15] K. Lee, R. Sakellariou, N. W. Paton, and A. A. A. Fernandes. Workflow adaptation as an autonomic computing problem. In 2nd Workshop on Workflows in Support of Large-Scale Science (Works 07) in Proceedings of HPDC, pages 29–34, 2007

[16] K. Vahi, M. Rynge, G. Juve, R. Mayani, és E. Deelman, „Rethinking data management for big data scientific workflows", in Big Data, 2013 IEEE International Conference on, 2013, o 27–35.

[17] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, és J. Good, „Pegasus: A framework for mapping complex scientific workflows onto distributed systems", Scientific Programming, vol 13, sz 3, o 219–237, 2005.

[18] T. Heinis, C. Pautasso, and G. Alonso. Design and evaluation of an autonomic workflow engine. In 2nd International Conference on Autonomic Computing, pages 27–38. IEEE Computer Society, 2005

[19] R. Duan, R. Prodan, and T. Fahringer. Run-time optimisation of grid workflow applications. In Proc. Intl. Conference on Grid Computing, pages 33–40. IEEE Press, 2006

[20] J.-H. Lee, S.-H. Chin, H.-M. Lee, T. Yoon, K.-S. Chung, and H.-C. Yu. Adaptive workflow scheduling strategy in service-based grids. In GPC, pages 298–309. Springer, 2007.

[21] B. Ludäscher, I. Altintas, S. Bowers, J. Cummings, T. Critchlow, E. Deelman, D. D. Roure, J. Freire, C. Goble, és M. Jones, „Scientific process automation and workflow management", Scientific Data Management: Challenges, Existing Technology, and Deployment, Computational Science Series, o 476–508, 2009

[22] Vahi, K.; Harvey, I.; Samak, T.; Gunter, D.; Evans, K.; Rogers, D.; Taylor, I.; Goode, M.; Silva, F.; Al-Shakarchi, E.; Mehta, G.; Jones, A.; Deelman, E., "A General Approach to Real-Time Workflow Monitoring," High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion: , vol., no., pp.108,118, 10-16 Nov. 2012

[23] Dias, J., Ogasawara, E., de Oliveira, D., Porto, F., Coutinho, A.L., Mattoso, M., "Supporting dynamic parameter sweep in adaptive and user-steered workflow", 2011 in: Proceedings of the 6th Workshop on Workflows in Support of Large-Scale Science. ACM, pp. 31–36.

[24] Julliano Trindade Pintas, Daniel de Oliveira, Kary A. C. S. Ocaña, Eduardo Ogasawara, Marta Mattoso, "SciLightning: A Cloud Provenance-Based Event Notification for Parallel Workflows", 2013.

[25] Ailamaki, V. Kantere, and D. Dash, 2010, Managing scientific data,Communications of the ACM, v. 53, n. 6 (Jun.), p. 68–78.

[26] Ailamaki, 2011, Managing scientific data: lessons, challenges, and opportunities, In:Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, p. 1045–1046

[27] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dun-lop, A. Williams, T. Oinn, and C. Goble, 2010, Taverna, reloaded, In:Proceedings of the 22nd international conference on Scientific and statistical database management, p. 471-481