A Specification and Execution Approach of Flexible Cloud Service Workflow based on a Meta Model Transformation

Imen Ben Fraj, Yousra BenDaly Hlaoui and Leila Jemni BenAyed
Research Laboratory in Technologies of Information and Communication and Electrical Engineering (LaTICE), Institute of Sciences and Techniques of Tunis, Tunis, Tunisia

Keywords: BPMN Model, MDA Approach, Meta-model Transformation, Flexible Workflow, Cloud Service.

Abstract: Cloud environments are being increasingly used for deploying and executing workflow composed from cloud services. In this paper, we propose a meta-model transformation for the specification and the execution of cloud service flexible workflows. To built workflow abstract models, the proposed approach uses a BPMN model for the specification of the cloud service workflow structure and the state-chart diagram for the specification of the cloud service workflow behavior. In addition, workflow models should be translated into BPEL4WS language which will be executed by the BPEL4WS engine, the latter is driven by the behavior described by the state-chart diagram. To fulfill, we define a set of meta-model transformations from the platform independent model (BPMN) to the platform specific model (BPEL4WS).

1 INTRODUCTION

1.1 Motivation

Cloud computing (Dillon and Chang, 2010) has emerged in the distributed computing community, it is mainly driven by the industry and has been largely adopted by the research domain. The basic goal of the cloud computing is to make a better use of distributed resources and be able to solve large scale computation problems. A cloud service is a web service that provides a set of well defined cloud interfaces and follows cloud specific conventions. These services constitute a powerful basis for modern application development. In order to enable users to compose their applications without taking care of the lower level details, the concept of cloud workflow has emerged as a method for modeling service applications (Yubin, Zeye, Zewei and Ximing, 2013). The problem of building such applications requires finding and orchestrating appropriate services that is frequently a non trivial task for a developer. This is due to the very large number of available services and the different possibilities for constructing a flexible workflow from matching services. Flexible workflow is a solution to fasten information system development in distributed and dynamical environment like the Cloud. Therefore, we propose in this paper a model driven approach for automatic execution of flexible workflow applications of cloud services using BPMN model (Allweyer, 2010). Recently, several solutions have been proposed to model applications from cloud services such as works presented in (Amziani, Melliti and Tata, 2013; Fengyu, Ying, Zheng, Wei and Xilong, 2015).

However, the proposed solutions need interaction with user and guidelines or rules in the design of the composed applications. In consequence, the resulting source code is neither re-usable nor it promotes dynamic adaptation facilities as it should. For applications composed of cloud services, we need an abstract view not only of the offered services but also of the resulting application. This abstraction allows in one hand the reuse of the elaborated application and on the other hand reduces the complexity of the composed applications. There are several architectural approaches for distributed computing applications (Hu, Zhang and Yu, 2002) which make easy the development process. However, these approaches need rigorous development methods to promote the reuse of components in future cloud development applications. It has been proven from past experiences that using structured engineering methods makes easy the development process of any computing system and reduce the complexity when

Fraj, I., Hlaoui, Y. and BenAyed, L.
A Specification and Execution Approach of Flexible Cloud Service Workflow based on a Meta Model Transformation.
DOI: 10.5220/0006338504670473
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 2*, pages 467-473
ISBN: 978-989-758-248-6
Copyright © 2017 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

building large cloud applications. To reduce this complexity and allow the reuse of cloud service applications, we adopt a MDA approach. The MDA (Gronmo and Jaeger, 2005) approach developing starts with defining high-level models in BPMN (Allweyer, 2010), defines conversion rules from BPMN to target platform, and then use code generation to derive much of the implementation code for desired platform.

1.2 Our Contribution

We propose a real time approach to built cloud services applications by following OMG(s) principals of the MDA in the development process (OMG, 2005). In this approach, we are interested to model and execute flexible workflows from existing cloud services. The workflow modeling identifies the resource from one depicted cloud service's operation to the next to built and compose the whole application. To model and express the composed flexible workflow of cloud services, we define and present in this paper a set of steps which begin from the specification of the cloud service workflow structure with BPMN and finish by processing the flexible properties of the workflow model using the real time meta-model transformation. In addition, we propose a control system model which controls the behavior of the BPEL engine based on the properties of flexibility defined in a state-chart diagram using the ECA rules (Hu, Zhang and Yu, 2002). The provided model forms the Platform Independent Model (PIM) of the proposed MDA approach. Figure 1 presents the architectural view of the proposed approach (Fraj, DalyHlaoui, Younes, and JemniBenAyed, 2015). At the first step of the approach, the user specifies its problem, i.e. the result that it wishes to get from the workflow composition, by modeling a composition request using BPMN model (functional view). Another specification of the cloud service workflow behavior using a state-chart diagram (behavioral view) should be defined. This request will be refined by the composition system to built the composed workflow from available cloud services. Before being executed, two transformations should be produced automatically, the first one is the transformation of the BPMN model into a running platform model like BPEL, the second is the transformation of the state-chart diagram into a running platform using a control system. Thus, we process the properties of flexibility of the workflow model using the real time meta-model transformation.

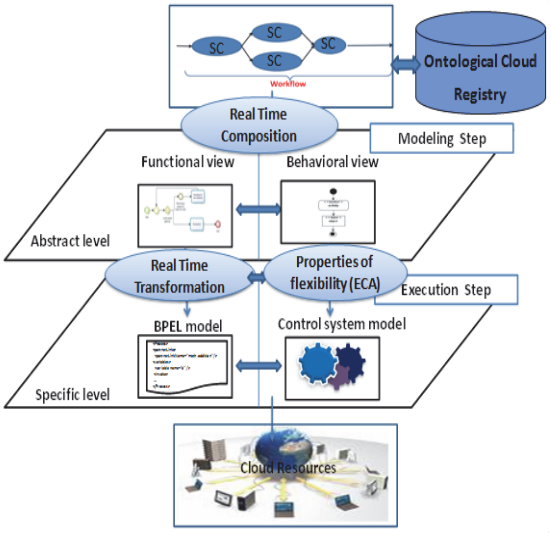

Figure 1: Architectural view of the approach.

1.3 Paper Reminder

This paper is organized as follows. Section 2 presents the related works. Section 3 details the proposed approach for systematic cloud services execution and section 4 presents the flexible workflow and the properties of flexibility for cloud services. Section 5 illustrates the specification process based on the proposed BPMN model and flexibility operator. Finally, section 6 concludes the paper and proposes areas for further research.

2 RELATED WORKS

Several works and researches were carried out in the field of modeling of flexible workflow of cloud services like works presented in (Amziani, Melliti and Tata, 2013; Fengyu, Ying, Zheng, Wei and Xilong, 2015). In (Fengyu, Ying, Zheng, Wei and Xilong, 2015) the authors propose a flexible UML-based workflow model that is able to exhibit the architecture of workflow engine and to adapt to the changes of business process. They were based on UML diagrams to describe the flexible workflow. This approach would have been better if the composition were automatically elaborated since the number of available services is in increase with the existence of several forms and manners to compose such services. In (Amziani, Melliti and Tata, 2013), the authors, propose a formal model elasticity for service-based business processes (SBP). In this model, processes are defined as Petri nets. Then, elasticity operations (duplication and consolidation)

are defined and heir correctness is proven. Each service is represented by at least one place (the set of places of each service are related with an equivalence relation). The transitions represent calls transfers between services according to the behavioral specification of the SBP. The originality of our contribution, relatively to this work, is that first we save the user from the dynamic refinement and execution as we propose an MDA approach which separates the specific model from the independent model. Second, we facilitate the composition of flexible workflow as we guide the user to compose her/his workflow using flexibility patterns. Third, we consider the properties of flexibility in the specification of the workflow in a more natural way for the human user as we define the functional and the behavioural views.

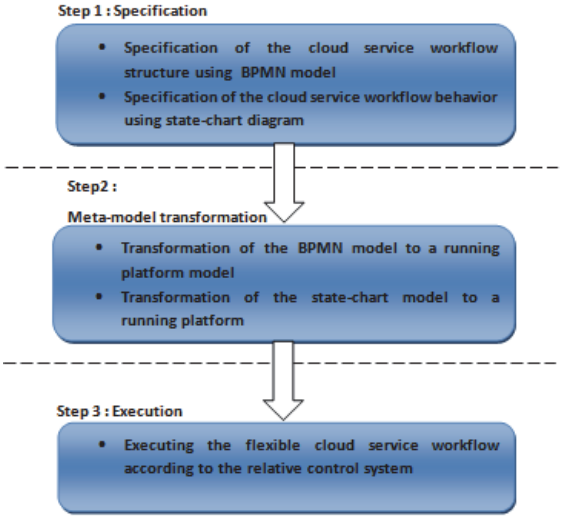

Figure 2: Steps of the approach.

3 THE APPROACH OF DEVELOPING COMPOSED CLOUD SERVICES FLEXIBLE WORKFLOWS

We propose a real time approach allowing the specification and the execution of flexible workflow applications composed from cloud services. The specification is based on semantic knowledge of cloud services and on the problem specification which the user provides to the composer system. There are three objectives to achieve by proposing an architectural approach for developing distributing computing applications:

1. ease the development process of such application,
2. enable rigorous development method,
3. promote the reuse of components.

To reach the first objective, our approach follows the model driven architecture by separating the platform development model from the platform specific model. Thus, we propose to specify the workflows model by a functional view using BPMN model (Allweyer, 2010). The model provides an abstract and understandable description of the cloud application which is obtained from the integration of cloud services in a workflow. The workflow model will be validated and verified to ensure the reliability of the future application before it is implemented and that to save material and human costs. Also the obtained models could be used as documentation for further application extension. Figure 2 illustrates the different steps and actions of the proposed approach.

Specification Step. The flexible workflow is modelled by a BPMN model based on different patterns of composition of workflows. Each activity represents the cloud service's operation that should provide the result. At the same time, the specification of the cloud service workflow behaviour is also defined using a state-chart model. The composition is described in an abstract level using BPMN model (Allweyer, 2010) for functional view and state-chart for behavioural view. Once the two views are built, the workflow should be validated and verified to ensure its reliability before being executed and reused as sub-workflow. In this paper we emphasize upon the modelling of composed flexible workflows only from Cloud services and not from sub-workflows.

Transformation Tool. This activity consists of two actions: export execution code and export sub-workflow description. The former transforms the BPMN workflow model into its executable workflow description by BPEL model. While the latter translates the state-chart model describing the behaviour of each properties of flexibility into its matching control system like ADA (Woodruff and Van Arsdall, 1998).

Execute the Flexible Workflow Application. The workflow description document is sent to a workflow engine that produces implementation code for handling control-flow and data-flow. An execution engine, such as BPEL4WS (Shen, Grossmann and Yang, 2007) engine, executes different workflow activities which are specified in a workflow execution document in the correct order and with their required input and output data. Before execution, the engine consider the preliminaries and requirements mentioned by the control system model, the latter control the behaviour described by

the state-chart model. Thus, the real time meta-model transformation is achieved.

4 FLEXIBILITY

4.1 Definition

The flexibility is the capability to implement changes of the requirements in the business process model and instances by changing only those parts of the business process model and instances that reflect the change.
Flexibility (Nurcan, 2008) has been the focus of many researches (Regev and Wegmann, 2005; Rosemann and Recker, 2006; Saidani and Nurcan, 2006; Schmidt, 2005). There are many definitions of the flexibility in literature (Shi and Danies, 2003). It is defined in (Regev and Wegmann, 2005) as "the ability to yield to change without disappearing". The flexibility is the capacity of making a compromise between, first, satisfying, rapidly and easily, the business requirements in terms of adaptability when organizational, functional and/or operational changes occur; and, second, keeping effectiveness. Processes
flexibility means fast reactivity to internal and external changes. It reflects the easiness to make evolve business process schemes (when required). Flexibility is also reflected by the ability that the support systems have to take into account business changes.

4.2 Flexible Workflow

The flexibility refers to the executable ability to flexible process definition by workflow management system (WFMS). Flexibility of workflow means the dynamic generation and modification on definition of process instances during execution. Workflow have to provide means to suit the flexibility and adaptability requirements at any given time. Flexible workflow management technology is one of the core technologies to fasten information system development in the cloud environments.

4.3 Flexibility Types

There are two types of workflow flexibility; flexibility by modelling (a priori) and flexibility by execution (a posteriori). The first type supports dynamic selection for functional variability or user interface variability of the cloud services. However, the second one supports dynamic modifications in different situations of the cloud service workflows. It, also, adapts the workflow to the current situation of the cloud during its execution.
The flexibility by modelling (a priori) (Nurcan, 2008) is based on modelling formalisms which can offer the capacity to deal with the environmental change without any evolution of process definitions. This means that this capacity should be incorporated in process definitions during build-time. It is based on the dynamic construction of instances by selection of components in a library. The process instances are al-ways in conformity with the process definition. This type of flexibility focus on the specification of a flexible workflow execution behaviour to express an accurate and less restrictive behaviour in advance; flexible and adaptable control and data flow mechanisms have to be taken into account in order to support ad hoc and co-operative work at the workflow level.
The flexibility by execution (a posteriori) (Nurcan, 2008) allows adapting the process definition or its instances during their execution. This is the most usual case in the literature. The principal challenge is to know (i) when and according to which criteria the adaptation should be done? (ii) in the type or the instance level? (iii) ? how to deal with the instances which are currently running? Approaches which
offer only this kind of flexibility are based on prescriptive modeling formalisms. this type of flexibility is provided by the change and evolution of workflow models in order to modify workflow specifications on the schema and instance level due to dynamically changing situations of a real process.

4.4 Flexibility Properties

In this section, we present the properties of flexibility, it can be achieved by three ways in which the routing of cases along workflow tasks or cloud services can be changed (Van der Aalst, 2001):

- *Extend*. Each cloud service has a maximal capacity of treatment over that the QoS of the service decrease and we risk to have a breakdown, thus the solution is adding new tasks which (1) are executed in parallel, (2) offer new alternatives, or (3) are executed in-between existing tasks.
- *Replace.* An instance of cloud service could be unavailable, so a task is re-placed by another task or a subprocess (i.e., refinement), or a complete region is replaced by another region.
- *Re-order.* A cloud service instance cannot follow the order defined in the work-flow

to reduce the execution time. Changing the order in which tasks are executed without adding new tasks, e.g., swapping tasks or making a process more or less parallel.

Thus, flexibility in cloud service workflows remains a solution for optimizing the cloud application performance and running costs of cloud services.

5 ILLUSTRATION OF THE EXECUTION OF FLEXIBLE WORKFLOWS FROM CLOUD SERVICES

In the following, we illustrate the execution process through the example of an online computer shopping service (Amziani, Melliti and Tata, 2013) composed of four services :

- Service requests (*S*1): receives requests to purchase a computer (processing time = 1s).
- Service components assembly (*S*2): performs the assembly of computer components according to the desired characteristics (processing time = 60s).
- Service invoice (*S*3): creates the invoice related to the purchased computer (processing time = 1s).
- Service delivery (*S*4): delivers the computer with its invoice to the customer (processing time = 1s).

Figure. 3 models the normal workflow functional behaviour, without considering the flexibility. However, we can meet some problems during this behaviour. These problems should be considered in the functional model to predict any abnormal behaviour.

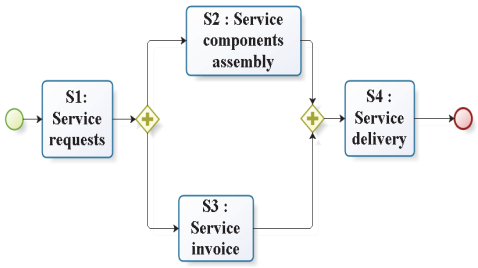

Figure 3: Initial workflow.

We resume these problems as follows:

Problem 1: Each service has a maximal capacity of treatment over what the QoS of the service decrease and we risk to have a breakdown (Amziani, Melliti and Tata, 2013).

Problem 2: In case of quality satisfaction , we are with another copy of service, already create and useless (Amziani, Melliti and Tata, 2013)..

Problem 3: In case of unavailable resource, the process of the execution risk to fail.

The use of the flexibility properties is the solution for these problems. We should update the initial workflow model (e.g. the model of Figure 3) to consider each of these properties based on the behaviour parameter of the cloud service. We define three modelling solutions:

Solution 1 (add): This solution consists in duplicating (Amziani, Melliti and Tata, 2013) the service to increase for treatment of requests. This copy is created to treat the excess of requests.

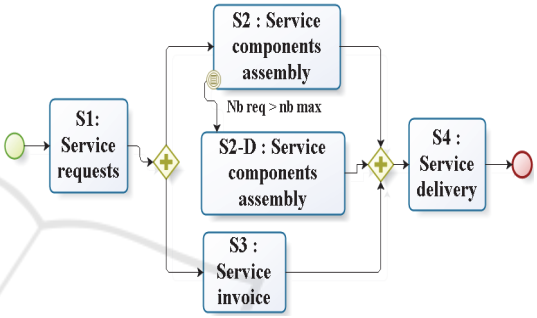

Figure 4: Duplication of service S2.

Back to the initial workflow, the Service S2 has the maximum time processing thus we can have an overload on this service when it has a lot of calls, to avoid that, we create a flexible workflow (Figure 4) which contains a duplication of the service S2 into S2-D when the condition is verified.

Our approach provide a real time transformation to BPEL model by this code:

```
<event name="Overload" />
  <case condition= "nb_req>nb_max">
   <invoke>
     <partner="S2D.assembly"/>
     <portType="GS2DPortype2D"/>
     <operation="assembly"/>
     <qos ="nb_req"/>
     <state =" true"/>
   </invoke>
    <reply partner ="reply2" porType =
"GS2DPortype2d" operation ="assemby"
variable = "R2"/>
  </case>
 </event>
```

Solution 2 (delete): If the number of the requests decreases, then it is necessary to delete the service already created (Amziani, Melliti and Tata, 2013). We create a new service which is responsible for deleting the copy (Figure 5)

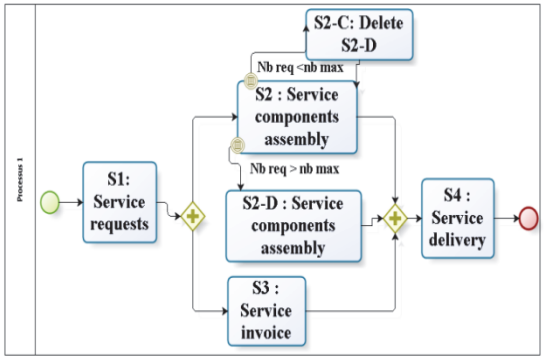

Figure 5: Delete of the service S2.

The transformation into BPEL is :

```
<event name="Unload" />
  <case condition= "nb_req<nb_max">
   <invoke>
     <partner="S5.delete"/>
     <portType="GS5Portype5"/>
     <operation="delete"/>
     <qos ="nb_req"/>
     <state =" true"/>
   </invoke>
     <reply partner ="reply5" porType =
"GS5Portype5" operation ="delete"/>
  </case>
```

Solution 3 (resource): This solution consists in adding new resource when the previous resource is unavailable, after a delay of waiting , we should add the new resource, to avoid the suspension of the execution. (Figure 6)

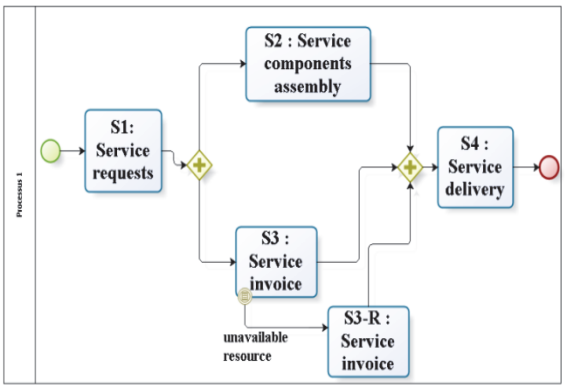

Figure 6: Add resource for the service S3.

The corresponding transformation into BPEL

```
<event name="Unavailable" />
  <case condition= "Temp_req>Temp_max">
   <invoke>
     <partner="S3R.invoice"/>
     <portType="GS3RPortype3R"/>
     <operation="invoice"/>
     <qos ="Temp_req"/>
     <state =" true"/>
   </invoke>
     <reply partner ="reply3" porType =
"GS3RPortype3R" operation ="invoice" variable
= "R3"/>
  </case>
 </event>
```

To be executed (See Figure 7), the composed Cloud service application models which are Platform Independent Models (PIMs) should be transformed to Platform Specific Models. The specific platform is the BPEL4WS (Andrews, Curbera, Dholakia, Goland, Klein, Leymann, Liu, Roller, Smith, Thatte, Trickovic, and Weerawarana, 2003) platform. To achieve the objective, we propose a meta-model transformation between different meta-model languages used to specify the PIMs and the PSMs of the composed Cloud service applications.

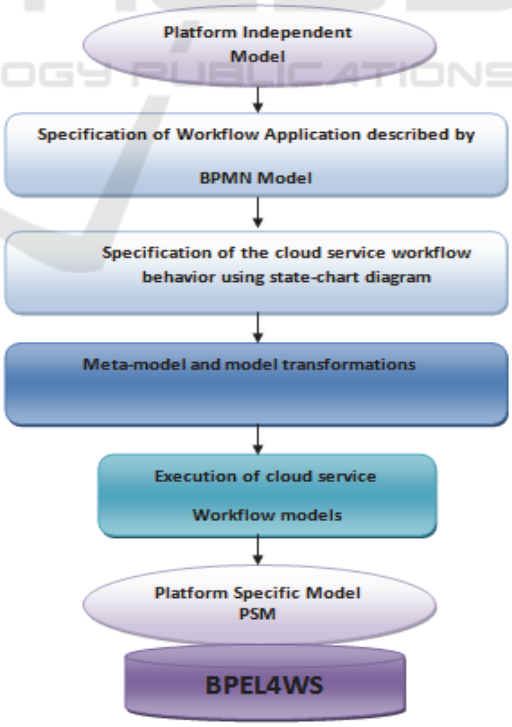

Figure 7: Meta-model transformation for executing composed Cloud service workflow models .

6 CONCLUSION

We have proposed an MDA approach for the specification and the execution of cloud workflow applications composed from cloud services. We have detailed a set of steps for integrating and matching, systematically, cloud services in a workflow. In addition, we have presented a set of properties of flexibility which are used in the execution process to depict the right cloud service to involve in the workflow. The approach shows also how the modelling proposed constructs are applied to model and represent a flexible workflow from cloud services specified by BPMN to its running platform by BPEL4WS. This process was illustrated under the example of an online computer shopping (Amziani, Melliti and Tata, 2013). As a proposal for further work and with the objective to realize the model-driven vision of OMG's MDA (OMG, 2005), we need to develop a control system that verifies the behaviour of the BPEL engine.

REFERENCES

Allweyer, T., 2010. *BPMN 2.0: Introduction to the Standard for Business Process Modeling*.

Amziani M., Melliti, T., Tata, S., 2013. Formal Modeling and Evaluation of Service-Based Business Process Elasticity in the Cloud. Chapter *On the Move to Meaningful Internet Systems*: OTM 2013 Conferences .Volume 8185 of the series Lecture Notes in Computer Science, pp 21-38.

Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S., 2003. Business Process Execution Language for Web Services, Version 1.1. BEA Systems, *International Business Machines Corporation, Microsoft Corporation*, SAP AG, Siebel Systems, Tech. Rep.

Dillon, T., Wu, C., Chang, E., 2010. Cloud Computing : Issues and Challenges. In *24th IEEE International Conference on Advanced Information Networking and Applications*, pp 27-33.

Ellis, C., Keddara, K., Rozenberg, G., 1995. Dynamic Change Within Work-flow Systems. In *Proc. Conference on Organizational Computing Systems* (COOCS), Milpitas, CA, pp. 10–22.

Fengyu, Y., Ying, C., Zheng, H., Wei, Z., Xilong, D., 2015. Design and Implement of a Flexible Workflow Model Based on UML Modeling Technology. *Applied Mechanics and Materials* Vols. 738-739, pp. 304-310.

Fraj, I. B., DalyHlaoui, Y. B., Younes, A. B., JemniBenAyed, L., 2015. Towards to Compose Cloud Service Flexible Workflow Applications. In *COMPSAC Workshops*, Taichung, Taiwan, pp 404-409.

Gronmo, R., Jaeger, M. C., 2005. Model-Driven Semantic Web Service Composition. In. *Proc. APSEC'05*, Dec, pp. 79-86.

Hu, J. M., Zhang, S. S., Yu, X.Y., 2002. A Workflow Model Based on ECA Rules and Activity Decomposition. *Journal of Software*, 13(4), pp 761-767.

Nurcan, S., 2008. A Survey on the Flexibility Requirements Related to Business Processes and Modeling Artifacts. In *Proceedings of the 41st Hawaii International Conference on System Sciences*.

O.M.G., 2005. *Uml 2.0 superstructure specification*. Technical report.

Regev, G., Wegmann, A., 2005. A Regulation-Based View on Business Process and Supporting System Flexibility, *Proc. of the CAiSE'05 Workshop*, pp. 91-98.

Reichert, M., Weber, B., 2012. Enabling Flexibility in Process- Aware Information Systems: *Challenges, Methods, Technologies*. Springer, Berlin-Heidelberg.

Rosemann, M., Recker, J., 2006. Context-aware Process Design Exploring the Extrinsic Drivers for Process Flexibility, BPMDS, Luxembourg.

Saidani, O., Nurcan, S., 2006. A Role-Based Approach for Modelling Flexible Business Processes, *The 7th Workshop on Business Process Modelling, Development, and Support* (BPMDS'06), (in association with CAISE'06), Springer Verlag (pub), Luxembourg, 2006.

Schmidt, R., 2005. Flexible Support of InterOrganizational Business Processes Using Web Services. *Proceedings of the CAiSE'05 Workshop*, pp. 51- 58.

Shen, J., Grossmann, G., Yang, Y., Stumptner, M., Schrefl, M., Reiter, T., 2007. Analysis of Business Process Integration in Web Service Context. *Future Generation Computing System*, vol. 23, no. 3, pp.283–294.

Shi, D., Danies, R.L., 2003. A survey of Manufacturing Flexibility: Implications For E-Business Flexibility. *IBM Systems Journal* 42(3), p.414-427.

Turab, N. M., Abu Taleb, A., Masadeh. S.R., 2013. Cloud Computing Challenges and Solutions. *International Journal of Computer Networks & Communications* (IJCNC) Vol.5, No.5.

Van der Aalst, W. M. P., 2001. How to handle dynamic change and capture management information? An approach based on generic workflow models. In *Proceedings of the Fourth IECIS International Conference on Cooperative Information Systems*.

Woodruff, J. P., Van Arsdall, P. J., 1998. A Large Distributed Control System Using Ada in Fusion Research. In *Proceedings of the 1998 annual* ACM SIGAda international conference on Ada, pp 121-131.

Yubin, G., Zeye, C., Zewei, L., Ximing, L., 2013. Design and Implementation of a Flexible Workflow Management System. *Journal of Software*, Vol 8, No 12, pp. 3060-3065.