December 1, 2008

RT0829
Computer Science 8 pages

Research Report

A Multi-layered Architecture for Workflow Management of IT Systems

Megumi Nakamura, Takayuki Kushida, Anuradha Bhamidipaty, Malolan Chetlur

IBM Research, Tokyo Research Laboratory
IBM Japan, Ltd.
1623-14 Shimotsuruma, Yamato
Kanagawa 242-8502, Japan

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

A Multi-layered Architecture for Process Variation Management

Megumi Nakamura
Shanon Inc.
Tokyo, Japan
megumi.n@shanon.co.jp

Takayuki Kushida
IBM Research,
Tokyo Research Laboratory
Kanagawa, Japan
kushida@acm.org

Anuradha Bhamidipaty, Malolan Chetlur
IBM Research,
India Research Laboratory
Bangalore, India
abhamidi@in.ibm.com, mchetlur@in.ibm.com

***Abstract*—Organizations and enterprises have to constantly re-engineer and customize business processes to stay competitive in their market. These modifications and customizations result in a large number of similar processes that lead to high costs of deployment and maintenance. This paper presents a multi-layer architecture for efficient management of process variations. The approach consists of organizing related processes into a *Base* layer and multiple *Variation* layers that separate the common fragment of the process from its customized parts. It avoids explosions of duplicate similar processes and allows business users flexible and easy change management of their processes. The implementation with the Eclipse modeling framework is also done to validate the proposed architecture.**

***Keywords*-business process; systems management; variation management; IT process;**

I. Introduction

A standard business and IT management procedures with a workflow management technology are followed to efficiently manage the information technology (IT) infrastructure of enterprises. These IT systems becomes much complex and pervasive. The standardization of business processes allows an organization to improve the service quality and reduce the operational cost. However, variations and customizations to those business processes are deemed necessary to increase a competitiveness and keep an efficiency. The business processe should be constantly re-engineered to adapt to the changing environments tools, expertise or for customer requirements and for other business related improvements. It is always required that the workflow management system should be flexible and easy to executing those business processes. The system should provide an ability to abstract the commonalities of similar related processes and create a variation of a process with specifying just the customized parts of the process. (the terms workflow and business process are used interchangeably in this paper).

To allow an efficient customization of business processes, a template of the workflow is usually prepared for the entire enterprise. The template is customized with many requirements of the individual organizations within the enterprise. An abstract workflow template prescribing to the corporate policy is typically prepared and strict rules and guidelines are provided to customize this template workflow to tailor to a specific environment. The governance and compliance around changes to a business process is necessary for the efficient maintenance and to minimize risks.

These variations and customizations of business processes result in workflow explosions leading to higher cost of deployment and maintenance. Managing variations of related business processes is not trivial and complexity increases with the number of variations. Similar business processes are typically treated as independent processes and any common changes are performed on all business processes. For example, a change for the corporate policy could result in process changes in the template workflow and all its variants. Current existing techniques to manage similar processes are usually manual and hence are error prone to accommodate enterprise wide process changes. Therefore, there is a strong requirement to manage those similar processes in a consistent and systematic manner.

A systematic approach to manage those variations among similar IT and business processes can reduce drastically both the deployment and the maintenance cost. An update to similar processes can be managed efficiently to avoid costly human errors and mitigate risks involved in such a change and update. Additionally, those tools to utilize the variation management among similar processes can also support authoring new variations that should be consistent with the guidelines provided to introduce customizations to the standard processes. In the results, the process variation management could simplify a business process governance in the enterprise.

The IT system should cater to the client requirements for IT service delivery. The requirements vary based on the designated locations and their organizations, and therefore it is usually introduced customizations within the standard processes of IT service delivery. IT service delivery environments should both be flexible to allow multiple customizations and be efficient without introducing additional work and delays. Because the sustained service quality and the same operational cost are required for IT service delivery. In addition, configuration and deployment of new and additional processes need to be simple, easy and efficient and align within the existing framework. Because they are maintained by IT service delivery organization. These form the core requirements towards an architecture for both efficient and flexible workflow systems in large enterprises.

This paper proposes a new multi-layer architecture for an efficient management of those process(also workflow) variations. The architecture consists of a *Base* layer and multiple *Variation* layers that can be separated with a large common part of the workflow from its customized small fragments. In the multi-layered architecture, those variation workflows specified to an environment where they are executed on are usually captured, created and managed independently with the workflow on the common base layer. In this scenario, when the workflow on the base layer is revised, the new updated workflow on the base layer can be combined with the customized portion resulting in the required workflow for the customized environment. This hierarchical workflow management is proposed to manage variation workflows among similar workflows within enterprises. In order to realize this architecture, a workflow should be comprised of workflow fragments, and composition and verification of its fragments are performed when it is required. In this architecture, to manage the relation and dependency between workflow fragments, these fragments can be easily exchanged for the similar requirements. This architecture also supports composition of a business workflow to combine their necessary workflow fragments and the composed workflow can be used within a workflow engine.

This paper is organized as follows: (i)Section II presents the architecture of hierarchical management of workflows, (ii)Section III presents related works in the management of business processes and workflow management, (iii) Section IV presents in detail the multi-layered variation management approach, (iv) Section V presents applications of multi-layered variation management architecture, (v)Section VI presents illustration of the approach within Eclipse Modeling framework (EMF), and (vi) Section VII presents concluding remarks.

II. Hierarchical Architecture

Hierarchical Management: This section proposes a layered architecture to manage both of customized part and template part separately. This layered workflow management aims at the adoption of a new structured workflow with an incremental customization for various kinds of local organizations.

Figure 1 shows the hierarchical workflow management which defines two layers which are **Base Layer** and **Variation Layers**. The base layer treats a workflow in common for the whole workflow. The access to the base layer should be restricted and it should be maintained by the central authority of the workflow management. It is usually the same as the standard set of workflows or the commonly used workflows among many groups. Referring to the base layer, all the changeable parts with the customization are described to the variation layers in Figure 1, which can be defined as an unique workflow. It is usually related to the specific requirements for the location, the organization and other environmental reasons. The responsibility and the coverage of thisworkflow management can be naturally divided with those two layers.

All customizations in the variation layer can be composed with those workflow fragments extracted from existing workflows for an efficiency of the workflow deployment. The workflows in the base layer are referred and executed from the variation layer, and then the workflows in both of the base layer and the variation layer are linked mutually, and can be treated as one continuous workflow from a user. When this composed workflow is constructed, a workflow validation with both the base layer and variation layer should be always evaluated.

A system administrator can work on the centralized management of IT system by using the base layer in order to meet the compliance of IT system with a layered architecture.

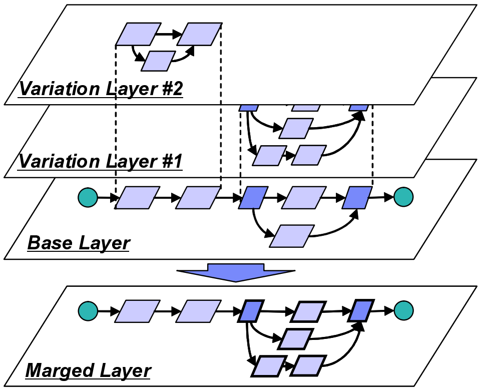


Figure 1. Concept of multi-layered architecture

Top-down Approach : In order to standardize a series of processes in IT system management, the administrator defines a workflow and applies it to a system (Figure 2). In system management process, they can use a template workflow on the base layer, such as a workflow described in Information Technology Infrastructure Library (ITIL). The ITIL is a set of best practice framework to manage IT services such as infrastructure, development and operations. Control Objectives for Information and related Technology (COBIT) is also a set of best practices of IT management which targets to measure the maturity of IT governance. When the enterprise introduces such a standard framework, it can be customized for the application. In the proposed architecture, the standard framework is introduced as the base layer and is separately managed for intelligible management.

Bottom-up Approach : In most cases, a workflow must be usually customized before the workflow is applied even if it is the standard one. Some workflows at many locations might be useful among locations and organizations. When

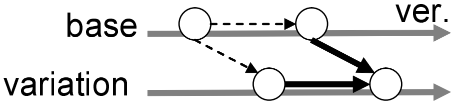


Figure 2. Top-down approach

there are valuable workflows that can be reused in the variation layers, they can be captured in a reusable format as in Figure 3. Even if it is not a standard workflow, the workflow that is frequently used can be easily introduced.

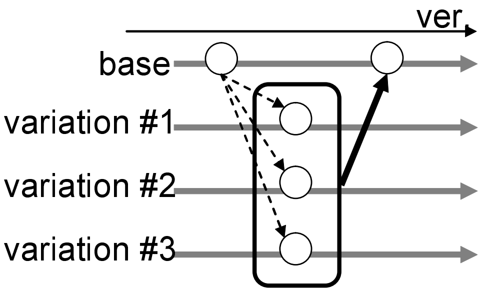


Figure 3. Bottom-up approach

Multiple-Layer Architecture: Following listed technologies are required to realize the proposed approach of the hierarchical workflow management and they are shown in Figure 4.

(A) Transformation: Converting the workflow description to canonical form for analysis and execution.

(B) Extension: Composing the variation layer, selecting fragments of workflow, and creating a repository for storing the fragments.

(C) Validation: Composition and validation for the combination of the base layer and the variation layers.

(D) Version control: Maintaining a set of the base layer and the variation layers, and change management for the variation layer.

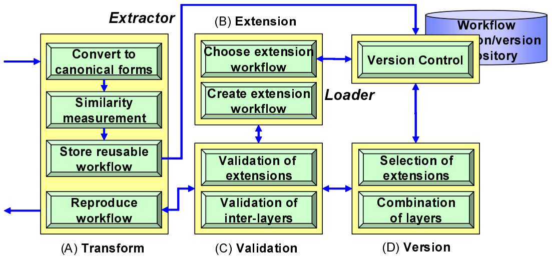


Figure 4. Management architecture

In the item 1), the input workflow written by various workflow descriptions is converted to a canonical form to be treated in the same logic, and the output workflow is transformed into the format which is used in the target environments. Workflow fragments are extracted by measuring their similarities, their structures, and their data flow. They are stored into the workflow repository. In the item 2), a workflow designer chooses reusable fragments from the repository, and composes the variation layers. In the item 3), the workflow created in the variation layer is checked so that it has the soundness structure, and the combination of the variation layer and the base layer is verified so that they work well on the workflow engine. In the item 4), the creation process of a workflow is managed and it helps for creating a better workflow. The relation of the base layer and the variation layer, the family line of the workflow fragments used in the variation layer are recorded, and then, enables it to reuse fragments and the tracking of change is made possible.

This architecture realizes the hierarchical workflow management and the life cycle of a workflow. Workflows used in actual business scenes are stored into the repository, and they are analyzed to extract reusable workflows for the variation layer. The portion of the workflow used in actual environment might be better to be described as a template workflow. Creation and application of a partial workflow can be considered as the same as the distribution and application of a patch of software development, and you do not need to write a workflow from scratch with this extensible workflow structure.

Two-Layer Architecture: Multiple-layer architecture combines the variation layers. However, there are many situations to write a workflow from scratch based on the standard workflow. In this case, the created workflow is put on the variation layer, and the two layers of the base layer and the variation layer are connected together. The workflow on the variation layer is created referring to the base layer, and the new workflow is produced from the two layers. The two-layer architecture requires the same technologies as what was referred in Figure 4.

III. Related Work

As it is explained in Section II, the layered workflow management requires various technologies toward incrementally customizing business processes to satisfy varying requirements. This section provides the survey of related studies that have been proposed in the literature which can be used in the four modules of the hierarchical architecture. Those four moudles are Transformation, Extension, Validation and Version Control.

The workflow management across organizations discussed in [1], [2] support the need for managing processes and variations at an enterprise level. [1] surveys required design and analysis techniques for cross-organizational business processes (CBPs). [2] defines model description of the information when exchanging the fragmentation of a process model among organizations. Inter-organizational processes have to accommodate variations due to geographical differences, organizational differences and workflow engine and tools to be used.

Transformation: [3] presents a technique to enhance control-flow analysis of a business process model. Their process model is decomposed into single-entry-single-exit (SESE) fragments in linear time, and a workflow graph is represented as a process structure tree (PST); and each fragment is analyzed and tagged with its categories. [4] presents a modular technique of workflow graph parsing to create a fine PST [3]. They have made some simplifying assumptions about workflow graphs to parse various patterns of workflow. [5] proposes 17 change patterns and 6 change support features to complement existing workflow patterns. These patterns are useful in the evaluation and analysis of workflows.

Extension: Efficient workflow creation is performed by reusing a workflow partially [6], [7], [8], [9], [10]. [7] proposes a life cycle model for reusing business process modeling. This life cycle begins with the planning phase, goes through the identification, (re-)design, annotation, storing, retrieving, tailoring, and integration phases and iterates back to the planning phase. [6] presents an approach to collaborative process model and a modeling tool for extracting distributed process model. This approach supports the management of CBPs that require collaboration between multiple sub-processes within or across organizations. [8] shows a process family architecture to model e-business in BPMN. This template based approach realizes rapid and cost-effective development and deployment of customer tailored business processes. [9] presents a micro-workflow architecture to introduce the workflow functionality required in object oriented applications. The core components provide basic workflow function, and additional components implement advanced workflow features. [10] discusses the emergence of the best business process practices in a fractal enterprise; and also discusses the life cycle of multilayer business process development consisting of (i) operation and evaluation, and (ii) business process design.

Validation: [11] discusses enterprise-wide and cross-enterprise business processes that must be performed without causing inconsistencies between the various models of the process-oriented application. [3] verifies the business model from the viewpoint of soundness of a control flow.

Version control: Hierarchical workflow management across organizations require technologies in version control such as extracting differences and merging workflows. Toward a version control of workflow, [12] proposes a process version management which detects the change in the XML process definition. The version graph includes attributes namely: object, a version number, and change operation type; and the version management uses check-in/check-out algorithms. In [13], the version management tracks both temporary changes and permanent improvements, and proposes updates to a workflow process definition on the fly. [14] analyzes the effect of the management of variant constructions and presents an approach for the version management of reference models as a solution. [15] analyzes the XML format used in the office documents and establishes XML based version control. They measure XML diff performance of XyDiff, Diffxml and Xmldiff, and implemented an XML versioning API that can be integrated into general version control systems.

[16] presents an approach for the detection and resolution of differences in the absence of a change log that is based on correspondences between process models, and also makes use of the concept of a SESE fragment decomposition of process models. All changed elements are expressed by primitive change operations instead of using a change log. [17] proposes X-Diff, an effective algorithm that integrates key XML structure characteristics with standard tree-to-tree correction techniques. Although this work does not involve business processes, many business processes are described by XML and hence this technique is useful and related to the approach presented in this paper.

[18] presents a conceptual object-oriented framework for role-based business process modeling. This proposes role based modeling as a mechanism to achieve separation of concerns to increase the understandability and re-usability of business process models. The approach divides a business process model into a business object model and a role model. [19] proposes a framework to show the relevance of control and resource patterns for context-aware business process re-engineering.

The techniques described in this section complement our work and can be incorporated as plugins to the various modules of the proposed architecture. To illustrate the concept, we have implemented one set of these techniques as described in section VI.

IV. Management of Layers

This section describes the management of layers in addition to the technologies referred in Section III. The management of the relationship between the base layer and the variation layers is required for the hierarchical workflow management. This management includes how the variation layers are composed and how the reusable fragments are maintained.

In the proposed architecture, the workflows used in several organizations in common are analyzed. Then, the reusable fragments are retrieved and are stored into the repository. We cannot expect that there is a common identification which is used in organizations because multiple-/two-layered workflow management is aimed at using a workflow with the customization little by little among many organizations. These issues are addressed to realize the workflow management.

Version Control of Base and Variation Layer: To efficiently manage a customized workflow, a version control should manage the function of the variation layer and the linkage of the corresponding base layer.

In this paper, version control to time-line is used for the management in the variation layer. In order to use a version control system in hierarchical workflow management, it has to be managed with reusable workflow fragments and the relationship of the base layer and the variation layer.

To bind the base layer and the variation layer, a workflow fragment of the variation layer refers the base layer by a location indicator. When the variation layer needs to change some elements in the base layer, only edit operation (ADD/DELETE/MODIFY and its object) is recorded to the variation layer. The ADD operation doesn't need to the variation layer if it has no location indicator to the base layer. The layers are merged to the workflow for the actual usage by applying the edit operation to the base layer.

For instance, if the variation layer refers Action X (name="Request Approval to Manager", param1="3days") of the base layer, the variation layer has the data for modification like Action X (base="Location_A"). If the data of the variation layer needs to be modified, the edit operation is recorded to the variation layer like Action X (base="Location_A", operation="MODIFY", name="Request Approval to Financial Team"). The actual workflow becomes to have Action X (name="Request Approval to Financial Team", param1="3days") after applying the variation layers to the base layer.

Composition Management for Variation Layer: When the variation layer is composed by choosing a workflow fragment, it needs to manage which workflow fragment should be chosen. The selected workflow fragments is referred as a location indicator like the case of referring the base layer. The feature of a workflow fragment on the created variation layer is saved as metadata. The process on the variation layer should be used for the verification when two processes of the base layer and the variation layer are united.

Cross-Organizational Workflow Management: This paper aims at managing efficiently, using a workflow effectively in a life cycle. The workflow used among several organizations is analyzed, and it is taken out as a workflow fragment for reuse. When judging whether it is reusable in this life cycle, a common portion and difference are distinguished. Since it is not the workflow used in the same environment, when analyzing the customized portion, common ID and a common name cannot be assumed.

Therefore, the similarity degree is calculated from attributes and structure of a workflow by using the policy which defines threshold, calculation method of similarity, and weight. Elements which have similarity beyond the threshold level are regarded as the same elements.

Visualization for Workflow Designer: When designing a hierarchical workflow, it is necessary for a workflow designer to understand which portion was customized parts on the base layer. The designer wants to look for the candidate of workflow fragments simply when he/she creates a variation layer. Dynamic verification of the combination of layers and dynamic notification of error or a correction candidate are useful for a designer.

Thus, it is critical to have data for each layer in workflow design to show the role of layer.However, when designing a workflow, it is necessary to show only one workflow for the designer in order to create a consistent workflow. Therefore, the method of dividing layers needs to be taken when the edited workflow is stored to file.

The base layer and the variation layer are completely separated with the workflow. The workflow can be edited by dividing workflow into two layers. A template and the edited part can be divided and managed by introducing this architecture to the editor. This realizes suitable workflow management even if the workflow is customized freely.

V. Applications

A. Consolidation

When we edit and use a template workflow, there are many workflows used by this template, and they have only a little difference. It becomes easy to grasp a parent workflow by dividing a workflow into the base workflow and the variation workflow. Moreover, when the same processes appear frequently in the variation layer, it may be better that the processes are included to a common template. Arranging the present condition and dividing workflows into a part with frequently change and a stable part makes managing of the workflow easier.

B. Analyzer

When the duration of each step is recorded, you can analyze the performance of each layer because the layers are separated. The sum total time shows the time which a workflow takes, and it turns into response time after receiving a request until service is offered. When estimating the quality of service quantitatively, the time which this service took in IT system management is investigated in many cases. The response time of the service to offer can be compared using the parameter of duration. When the candidate for comparison uses common workflow on the base layer, the process in the variation layer brings a difference. We can use the information acquired here for the analysis of bottlenecks in the workflow of comparison contrast.

VI. Illustration

This section presents an illustration of multi-layered workflow management of similar processes. The following illustration depicts the advantage of maintaining those different workflows across organizations without maintenance of individual workflows for similar but different processes. Let us consider a global delivery service company *ABC* which manages IT infrastructures for its customers. ABC offers similar service to its customers *Bank A* and *Bank B*. Let us assume due to contractual obligations, all Bank B related to IT services are to be performed in the same geography of where Bank B is situated. On the other hand, there are no geographic restriction on performing the services of Bank A. ABC decides to distribute the individual tasks of Bank A among its other locations to efficiently use the expertise and resources.

We could consider a specific IT service and its associated processes impacted based on the service delivery policies adopted between its customers. Let us consider a specific service where ABC need to manage to apply urgent updates to OS and databases installed on its customer infrastructures. The series of steps to fulfill this service are: (i) Receive request from customer, (ii) Get approval for change, (iii)Perform Change, and (iv) Notify service completion.

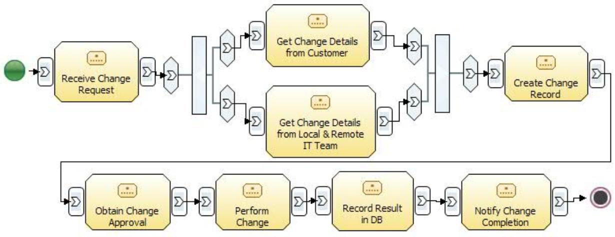


Figure 5. Process to fulfill Bank A's request

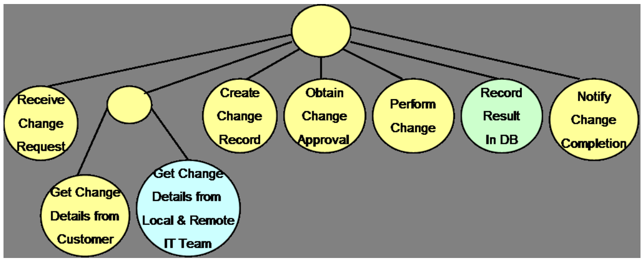


Figure 6. Process Structure Tree of Bank A's request

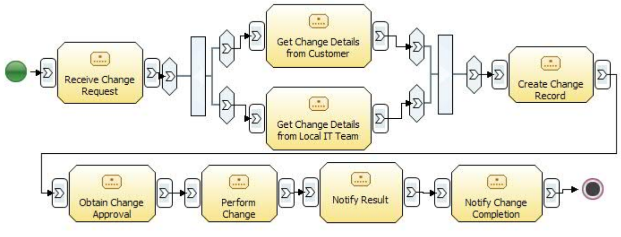


Figure 7. Process to fulfill Bank B's request

Figures 5 and 7 present the processes to satisfy the service request of Bank A and Bank B respectively. The variation management system automatically generates the process structure tree (PST) of the processes (as described in [3], [4]) and analyzes their differences. Figures 6 and 8 depict the process structure trees of Bank A process and Bank B process respectively. It extracts the common (base) workflow that is followed in both the processes. It also extracts the difference between the common workflow and the original workflow referred as *diff* workflow. Based on the common and *diff* workflows, the original workflows can be restored. The workflow management system also enables authoring a new workflow by adding variations to the existing workflow. The authored workflow keeps track of the base layer and recording the changes performed with the base. This allows single copy of base workflow and its variations arrived based on the patches (workflow fragments) applied. This results in lower cost due to the maintenance of single base layer for all instances of similar workflows. This also ensures similar handling of common process steps across enterprises, thus ensuring quality.

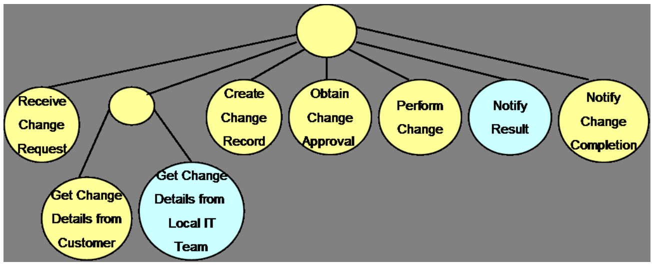
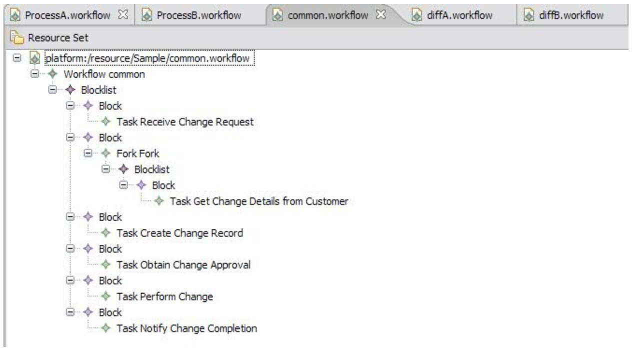


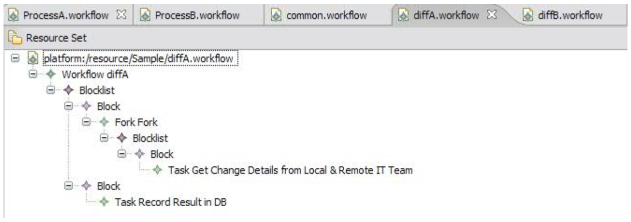
Figure 8. Process Structure Tree of Bank B's request

As a proof of concept we have implemented a prototype with the Eclipse plug-in, including (i) the functionality for creating a base workflow given two related workflows, and (ii) creating a variation layer from a base layer by inserting workflow fragments. Figures 5 and 7 show two processes (for Bank A and Bank B) modeled in IBM Websphere Business Modeler (WBM) [20]. These processes are exported from WBM as XML files and imported in Eclipse as workflow models. Figure 9 shows the PST representation of the base workflow derived from processes A and B. Figures 10 and 11 depict the workflow fragments obtained by the diff operator as introduced earlier. The base workflow as derived from this process can be used by business users to create new process models by using the ADD/DELETE/MODIFY operations. Figure 12 shows a new process, VariationA, created from the base process by inserting a new node. The bottom half of Fig. 12 shows that for the new node inserted, the 'Base Layer' property is empty whereas the rest of the nodes are derived from the base layer. Using this prototype software, users could easily identify and create a base process capturing the commonalities of several variants for a process thereby enabling them to introduce a process change to the base layer which could propagate to all variants of that process.



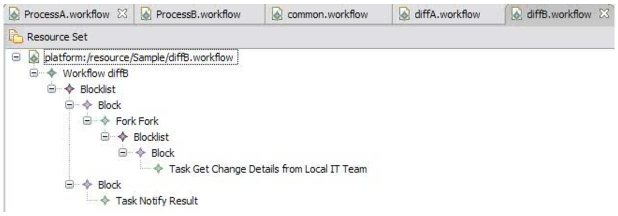
```
platform:/resource/Sample/common.workflow
  Workflow common
    Blocklist
      Block
        Task Receive Change Request
      Block
        Fork Fork
          Blocklist
            Block
              Task Get Change Details from Customer
      Block
        Task Create Change Record
      Block
        Task Obtain Change Approval
      Block
        Task Perform Change
      Block
        Task Notify Change Completion
```

Figure 9. Process Structure Tree Common to Process A and Process B



```
platform:/resource/Sample/diffA.workflow
  Workflow diffA
    Blocklist
      Block
        Fork Fork
          Blocklist
            Block
              Task Get Change Details from Local & Remote IT Team
      Block
        Task Record Result in DB
```

Figure 10. Difference between Process A and Common Process Steps

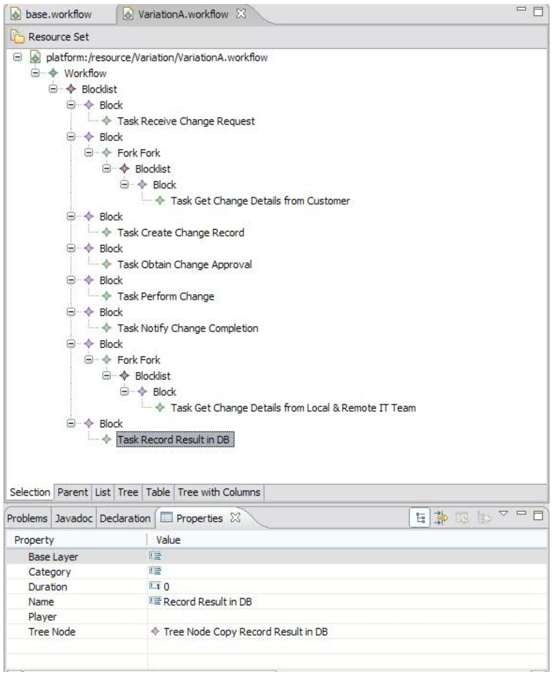


```
platform:/resource/Sample/diffB.workflow
  Workflow diffB
    Blocklist
      Block
        Fork Fork
          Blocklist
            Block
              Task Get Change Details from Local IT Team
      Block
        Task Notify Result
```

Figure 11. Difference between Process A and Common Process Steps

VII. Conclusion and Future Work

The IT systems management becomes complex because complex business processes are executed on those IT systems, therefore advanced workflows are usually required for those IT systems. The business procedures adopted to achieve the similar effect vary based on the customer needs, tools used, corporate policy, and laws followed in the situated location. These result in several workflows to be authored that are similar but customized in certain parts. The variations resulting in these workflows result in higher deployment and maintenance costs. This also affects the quality and increase the risk during changes to business processes. This paper presented a multi-layer architecture to manage different but similar processes. This architecture manages similar workflows together by identifying the base layer that is common to the related workflows and the variation layers containing customizations to the base layer. The restoration of original workflow essentially reduces to applying the variation layer patches to base layer. This results in consolidation of variations and systematic introduction of variants to any process. This multi-layered architecture also supports workflows maintained in distributed locations, with base layer in a central repository and location specific customizations maintained locally and linking them during workflow execution. This multi-layered architecture also supports composing new business processes by changing the existing business processes in its repository. A proof of concept of this architecture is demonstrated within Eclipse framework.



```
platform:/resource/Variation/VariationA.workflow
  Workflow
    Blocklist
      Block
        Task Receive Change Request
      Block
        Fork Fork
          Blocklist
            Block
              Task Get Change Details from Customer
      Block
        Task Create Change Record
      Block
        Task Obtain Change Approval
      Block
        Task Perform Change
      Block
        Task Notify Change Completion
      Block
        Fork Fork
          Blocklist
            Block
              Task Get Change Details from Local & Remote IT Team
      Block
        Task Record Result in DB
```

Property	Value
Base Layer	
Category	
Duration	0
Name	Record Result in DB
Player	
Tree Node	Tree Node Copy Record Result in DB

Figure 12. Process Variation Management

There are several directions for future work. One focus is on the application of the concept and the tool presented in the paper. We have applied it to a service delivery scenario and plan to extend the application to complex IT service delivery processes and further validate the approach. Then, this approach can be tested by incorporating multiple techniques to do transformation, validation and version control of processes. Future work will also include the elaboration of our approach to create Base and Variation layers of the processes in distributed environments.

References

[1] A. B. Sonia Lippe, Ulrike Greiner, "A survey on state of the art to facilitate modelling of cross-organisational business processes," *Proceedings of the Second GI-Workshop*

XML4BPM - XML for Business Process Management (XML4BPM 2005), 2005.

[2] W. D. F. Lindert, "Modelling inter-organizational processes with process model fragments," *Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications*, 1999.

[3] F. L. Jussi Vanhatalo, Hagen Volzer, "Faster and more focused control-flow analysis for business process models through sese decomposition," *In Proceedings of 5th International Conference on Service-Oriented Computing (ICSOC 2007). LNCS 4749, pp. 43-55. Springer, September 2007*, 2007.

[4] H. V. Jussi Vanhatalo and J. Koehler, "The refined process structure tree," *6th International Conference on Business Process Management (BPM)*, 2008.

[5] S. R. Barbara Weber and M. Reichert, "Change patterns and change support features in process-aware information systems," *CAiSE 2007*, 2007.

[6] K. S. Wasim Sadiq, Shazia Sadiq, "Model driven distribution of collaborative business processes," *IEEE International Conference on Services Computing (SCC'06)*, 2006.

[7] F. L. Zhilei Ma, "A lifecycle model for using process fragment in business process modeling," *Business Process Life-Cycle: Design, Deployment, Operation & Evaluation (BPMDS'08)*, 2008.

[8] F. P. Arnd Schnieders, "Variability mechanisms in e-business process families," *9th International Conference on Business Information Systems (BIS 2006)*, 2006.

[9] D. A.Manolescu, "An extensible workflow architecture with objects and patterns," *Technology of Object-Oriented Languages, Systems, and Architectures*, 2003.

[10] M. K. Julija Stecjuka, Janis Makna, "Best practices oriented business process operation and design," *Business Process Life-Cycle: Design, Deployment, Operation & Evaluation (BPMDS'08)*, 2008.

[11] P. D. M. Reichert, T. Bauer, "Enterprise-wide and cross-enterprise workflow-management: Challenges and research issues for adaptive workflows," *Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications*, 1999.

[12] J. B. Hyerim Bae, Eunmi Cho, "A version management of business process models in bpms," *Advances in Web and Network Technologies, and Information Management, APWeb/WAIM 2007 International Workshops Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg 2007*, 2007.

[13] C. L. Xiaohui Zhao, "Version management in the business process change context," *Business Process Management, 5th International Conference, BPM 2007 Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg 2007*, 2007.

[14] O. Thomas, "Design and implementation of a version management system for reference modeling," *Proceedings of the 40th Annual Hawaii International Conference on System Sciences, Journal of Software, IEEE*, 2008.

[15] U. M. B. Sebastian Ronnau, Jan Scheffczyk, "Towards xml version control of office documents," *Proceedings of the 2005 ACM symposium on Document engineering*, 2005.

[16] A. F. Jochen M. Kuster, Christian Gerth and G. Engels, "Detecting and resolving process model differences in the absence of a change log," *6th International Conference on Business Process Management (BPM)*, 2008.

[17] J.-Y. C. Yuan Wang, David J.DeWitt, "X-diff: An effective change detection algorithm for xml documents," *Proceedings of the 19th International Conference on Data Engineering (ICDE'03)*, 2003.

[18] J. T. Artur Caetano, Antonio Rito Silva, "Using roles and business objects to model and understand business processes," *2005 ACM Symposium on Applied Computing*, 2005.

[19] O. S. S. N. Kahina Bessai, Bruno Claudepierre, "Context-aware business process evaluation and redesign," *Business Process Life-Cycle: Design, Deployment, Operation & Evaluation (BPMDS'08)*, 2008.

[20] "IBM Websphere Business Modeler, v6.2," http://www-01.ibm.com/software/integration/wbimodeler/.