

# Workflow considerations in the emerging CI-ML virtual organization

Chris Boyle<sup>1</sup>, Artur Abdullin<sup>1</sup>, Rammohan Ragade<sup>1</sup>, Maciej A. Mazurowski<sup>1</sup>, Janusz Wojtusiak<sup>2</sup>,  
Jacek M. Zurada<sup>1</sup>

**Abstract**—In a virtual organization, the interaction of its members for any purpose generates a sequence of activities referred to as a workflow. This paper seeks to identify the workflows needed for the Computational Intelligence and Machine Learning Virtual Organization. The underlying architecture of the repository should support these workflows in a smooth and efficient manner.

**Index Terms**—virtual organization, computational intelligence, machine learning

## I. INTRODUCTION

IN a companion paper [1], a vision for a collaborative virtual organization (VO) for the Computational Intelligence and Machine Learning (CI-ML) community was introduced. The purpose of this VO is to provide a portal for peer reviewed software, algorithms, tools, data, and models. Members of the VO are allowed to review submitted content from outside developers and determine if it is acceptable for redistribution in the organization's repositories. Outside learners are allowed to access this content for educational purposes. In a VO, the interactions of the different types of users that utilize computer resources generate a sequence of pre-defined activities known as workflows.

Workflows have been widely discussed in the development of virtual organizations. The work of Khoshafian and Buckiewicz [2], describes their consideration of workflows in collaborative computing. In the area of machine learning and distributed AI, Singh and Huhns [3] laid out challenges for cooperative computing. With agents they handled issues of workflows in the computing task. More recently, we see examples of workflows in cooperative, collaborative computing in bioinformatics, computer security applications, text mining and other large scale applications. Java based agent technologies have enabled smoother software engineering tasks for the infrastructure of virtual organizations [4]. Even assessment processes, which has many factors in common with reviewing, has seen the need for workflows in computer supported collaborative tasks [5]. Dennis Gannon and his associates identify workflows for e-science [6] which is clearly an aspect in CI-ML. At the January 2008 workshop titled "Building Effective Virtual Organizations" hosted by the National Science Foundation, Marru, Gannon and Plale [7] gave a presentation to identify the role of workflows in e-science.

The workflows for the CI-ML VO are categorized as: sharing data, sharing software resources, sharing computational resources, education and networking. These workflow categories, while common for many other VO's, gain some unique qualities when applied to CI-ML. These qualities were considered at the NSF workshop to identify many established and emerging VOs and their characteristics. A common characteristic is that each VO has a domain of application, in which most of the members of the VO are familiar. The infrastructure being established by the VO should nurture and sustain the regular and evolving activities of the VO. Therefore, the lifecycle of the many workflows in a VO should be considered in detail.

The life cycle of a workflow identifies the resources and interactions between a sequence of logical activities. The computing resources set up for each interaction within the VO is dependent on how these interactions are identified. These interactions are easily defined as a use case to describe how a type of system user is able to interface with the rest of the system. A use case is usually defined with an actor, which can be considered as a user or subsystem, and a definition of acceptable behaviors across system boundaries. The formal definition of all possible use cases for a particular actor is important due to implications on system security and information assurance. The collection of use cases define the functional requirements for the VO and define the services it provides. Given the functional requirements and its accompanied workflows a software architecture can be created that is modular and scalable.

The importance of creating workflows has major implications for the future growth of a VO. Considerations for the CI-ML VO were made to allow its expansion given the addition of new functional requirements and use cases. This forced the software architecture to be well defined to accommodate these changes while having minimal impact on the system as a whole. This paper will discuss these workflows, their impact on the software architecture, and security issues to accommodate the use case requirements.

## II. WORKFLOWS ENVISAGED

The goal of the CI-ML VO is to create a community that will eventually become a place where researchers, students, and the general public come to seek information about computational intelligence, machine learning, and related topics. The section below discusses the scenarios in which workflows are needed. These scenarios are then considered for the workflow

<sup>1</sup>University of Louisville, Louisville, KY

<sup>2</sup>George Mason University, Fairfax, VA

categories: sharing data, sharing software resources, sharing computational resources, education and networking.

#### A. Scenarios

There are a wide variety of evolving scenarios for the VO. The main focus will be on the sharing and collaborative efforts among users of the system.

Most researchers in CI-ML, as in any other research community will review scholarly publications in monographs, e-journals, journals and conferences. If the content for review is not familiar, researchers seek other work that is closely related. This activity results in a variety of tasks associated with reviewing and referring. If it is a new method or algorithm, the researcher may even try out the method or tool on a familiar problem, before taking further action. If there is an associated software package, depending on the complexities of the package further workflows would ensue.

A researcher may seek to collaborate with other reviewers to assess a contribution. In this case, the researcher must establish the compatibility of the datasets with the proposed tools and domains. If needed, datasets will be transformed to enable appropriate tools to be used. Subsequently, comparative results require normalization for fairness in evaluation. Benchmarks often provide frameworks for this task.

When developing a new method, use of known software toolsets in the composition of proposed methods brings in additional considerations of software versions, licences, permissions, languages used and operating system environments. Through dialog in the CI-ML community, the researcher can obtain appropriate components for the proposed new method. This dialog process generates yet another appropriate workflow that should be facilitated in the emerging VO for the CI-ML community.

Established VO's such as those in genomics have already standardized processes for access to shared community resources. However, due to serious considerations of intellectual property (IP) and concern for prior disclosure, even queries have to be conducted in a secure non-disclosure proof environment. These concerns will also carry over to the emerging CI-ML VO, atleast where software tools are concerned.

Based on these scenarios, we have identified basic top level categorization of broad use cases for a review process. These are from a member submitter's perspective and from an administrative perspective.

#### B. Categories

The five workflow categories of the VO are: sharing data, sharing software resources, sharing computational resources, education, and networking. Listed below are the various activities and processes for each category. Each element in a category defines a workflow along with the interactions between elements. The permutations and combinations of the interactions are too numerous to be listed. This highlights the importance of defining an element as independently as possible. For example, proposing a new algorithm may require the use of many other processes but these do not need to be considered.

1) *Sharing Data*: The sharing data category encompasses the requirement that the VO handle algorithms, data, and models [1]. An outside developer has the ability to make a proposal for one of these types for review with a VO member. For example, when a new algorithm is proposed, a set of testing procedures must be defined for verification.

These are the activities and processes for the category:

- Proposing a new algorithm, data, or model. The proposal of a new elements initiates the review and testing for the user's submission.
- Reviewing/Testing
  - Benchmarking
  - Comparison with other methods/data
  - Scaling for real problems
  - Storing associated data
  - Accessing relevant data
  - Provide collaborative review of results
  - Data format transfer issues

2) *Sharing Software*: The sharing software category encompasses the requirement that the VO handle software and tools [1]. An outside developer can propose software in either its source code or executable form for review.

These are the activities and processes for the category:

- Finding quality software
- Making modules work together
- Check for language syntax/formatting/documentation issues
- Evaluate appropriateness for the CI-ML domain

3) *Sharing Computational Resources*: Computational resources for the system must be protected and managed correctly. Given that computational resources can be released, workflows must be defined for control.

- Working towards an open Framework development for large scale application
- Allowing diverse distributed resources/grid approach, etc.

4) *Education*: A future goal of the VO is to allow the collaborative sharing of educational content.

- Activities for generating, posting, updating Tutorials
- Creation of Frequently Asked Question (FAQ) sections
- Utilizing newer modes, such as access of multimedia content

5) *Networking*: Another future goal is to allow members, developers, and learners to collaborate and network together.

- Identifying focused CIML interests among members
- Providing channels of communication among members

The above workflows are substantially complex and the architecture should accommodate the workflows productively. As we move towards identifying appropriate architecture, issues of security and priorities for work flows take center stage. The next two sections briefly address our approaches towards these tasks.

### III. CURRENT VO ARCHITECTURE

The basis for the current architecture for the CIMLVO is from Martin Folwer's book for Patterns of Enterprise Architectures [8]. Three principal layers are identified for presentation,

services, and data access. These are created in order to divide responsibilities into smaller more manageable entities. For example, the required workflows for the presentation layer have a differing focus than the data access layer. This separation of concerns promotes modularity throughout the system.

The presentation layer has the responsibility for providing a user interface for the application. Given that this is a virtual organization, the presentation layer will be accessible through the World Wide Web and the services that the VO provides are defined through this gateway. The responsibilities for its workflows are focused with interacting directly with the user. For example, when the user types in an invalid input, a process is defined to display error messages. The presentation layer has a direct dependency to the service layer.

The service layer provides all of the application logic or business logic. The previous section has defined the major workflows that are implemented in this layer. If the presentation layer needs to access a table in a database, the service layer will provide the functionality through its dependency with the data access layer. Most workflows originate for this layer in the architecture because it provides all of the foundational logic for the entire system. This layer is divided into smaller managers that further modularize functionalities. For example, the submission manager allows submission of new content.

The data access layer gives access to any persistent storage that is needed. For example, if direct access is needed for a database. The VO uses a database for persistent storage and has an object-relational mapping (ORM) for interacting with the database. This provides a nice abstraction layer so that specific knowledge of the storage method is not needed.

The VO is coded in Java® and hosted inside a Java Application Server. The application server is responsible for the handling of requests and instantiation of workflows. The server generates a finite number of threads for new requests and responds accordingly. The number of concurrent threads is variable and can be changed to fit the needs of the VO. It is important to maintain this property by monitoring the needs of the system and providing the necessary system improvements.

#### IV. SECURITY ISSUES

The workflow architecture for the VO should contain explicit requirements for the management and enforcement of security and privacy of sensitive information. The workflows must be implemented in a way to guarantee that the functionality of the network does not override any security concerns. More specifically, workflows should enforce three principals for the VO: integrity, authorization, and availability [9]. These properties become exceedingly more important because the VO typically handles sensitive data and protected functionalities.

##### A. Integrity

The VO secures its workflow tasks by assigning each user a set of roles that classify the user's responsibilities. The layered structure of the software architecture helps by dividing the Service/Workflow layer from the Data Access layer. The

Service/Workflow layer maintains data integrity by correctly defining workflows and maintaining user responsibilities.

A role based security model provides many benefits because it reduces complexity and can easily be incorporated with existing technologies. Most importantly it allows the system to run by the principal of least privilege, where the minimal set of privileges is allowed to execute [10].

##### B. Authorization

Each role within the VO has permission to perform a set of defined responsibilities on the data. The originating paper describing the VO [1], discusses a layered model for access of developer's submission. The figure below shows the layered nature of the submission network.

The most basic VO roles are defined as: Learner/User, Developer, and Member. The responsibility of each type of basic VO user is a subset of the responsibilities for a higher layer. The basic roles can be defined as:

$$U \subseteq D \subseteq M$$

where

$U$  = Responsibilities of Users/Learners

$D$  = Responsibilities of Developers

$M$  = Responsibilities of Members

A more complex set of roles can be assigned to a VO user to handle administration duties. For example, an editor role exists that has the ability to assign a review to an appropriate member. There are many disjoint administration roles than contain smaller responsibilities. This is done to provide flexible management of the VO, which has the ability to grow to a very large size. It is vital to plan for future growth by allowing micromanagement of the VO. All administrative users have the same responsibilities as developers but are not required to be members.

Using this methodology, integrity is maintained by allowing data manipulation to occur by responsible VO users. Each role within the VO interacts with the Service/Workflow layer of the VO software architecture. The Service layer is responsible for maintaining the authorization and data integrity. It then interacts with the Data Access layer for data access and modification.

##### C. Availability

An important part in the development of a workflow is that required resources are available upon request. The workflow must be able to execute within a reasonable amount of time so that the VO can function reliably. A secure workflow that provides the availability property is defined as: For every task there must be at least one agent who is able to execute the task [9].

The availability property is also maintained by proper testing of workflow logic. The distributed and concurrent nature of the VO requires analysis of the workflow to avoid negative symptoms like dead-lock. Thorough unit, integration, and system testing is required for verification that all three principals are maintained.

## V. CONCLUSIONS

It is well known that in any networking activity, complexities of workflows and tasks increase as the number of interacting entities increase. For an emerging VO, it is essential to maintain coherence of the activities to its stated goals and purpose. The community must evolve a mechanism to make adjustments and adaptations. Portals are established for this purpose, which allow filtering of activities. The modular and layered approach will allow new workflows to be integrated into the architecture, with careful consideration. As long as the main focus is to nurture the VO community, these additions and adaptations can be accommodated.

## ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation Grant CBET 0742487. The findings and opinions expressed here are those of the authors, and do not necessarily reflect those of the sponsoring organization.

Chris Boyle's contribution supported by the University of Louisville Intramural Research Incentive Grant. Also, we would like to thank Jordan Malof on his editing suggestions for this paper.

## REFERENCES

- [1] J. M. Zurada, M. A. Mazurowski, J. Wojtusiak, R. Ragade, and J. Gentle, "Building virtual community in computational intelligence and machine learning," *Computational Intelligence Magazine*, 2008.
- [2] S. Khoshafian and M. Buckiewicz, *Introduction to Groupware, Workflow, and Workgroup Computing*. John Wiley and Sons, 1995.
- [3] M. P. Singh and M. N. Huhns, "Challenges for machine learning in cooperative information systems," in *Distributed artificial intelligence meets machine learning, Lecture Notes in Artificial Intelligence*, pp. 11–24, Springer-Verlag, 1997.
- [4] M. L. Griss, "Software engineering with java agent components," in *Borcon 2003*.
- [5] K. R. Lai and C. H. Lan, "Modeling peer assessment as agent negotiation in coputer supported collaborative learning," *ICMAS'96 Workshop LIOME*, 2006.
- [6] I. Taylor, E. Deelman, D. Gannon, and M. Shields, *Workflows for e-Science: Scientific Workflows for Grids*. Secaucus, NJ, USA: Springer-Verlag, 2006.
- [7] S. Marru, D. Gannon, and B. Plale, "Lead workflow use cases," [powerpoint presentation at the NSF workshop on Emerging Virtual Organizations, Washington DC, Jan 15 -17, 2008.].
- [8] M. Fowler, *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.
- [9] P. C. K. Hung and K. Karlapalem, "A secure workflow model," in *ACSW Frontiers '03: Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003*, (Darlinghurst, Australia, Australia), pp. 33–41, Australian Computer Society, Inc., 2003.
- [10] J. Whittaker, "Why secure applications are difficult to write," *IEEE Security and Privacy*, vol. 1, no. 2, pp. 81–83, 2003.