# Omnispective Analysis and Reasoning

## A framework for managing intellectual concerns in scientific workflows

Srinivas Chemboli
Research School of Computer Science
The Australian National University
Canberra, Australia
srinivas.chemboli@anu.edu.au

Clive Boughton
Research School of Computer Science
The Australian National University
Canberra, Australia
clive.boughton@anu.edu.au

## ABSTRACT

Scientific workflows are extensively used to support the management of experimental and computational research by connecting together different data sources, components and processes. However, certain issues such as the ability to check the appropriateness of the processes orchestrated, management of the context of workflow components and specification, and provision for robust management of intellectual concerns are not addressed adequately. Hence, it is highly desirable to add features to uplift focus from low level details to help clarify the rationale and intent behind the choices and decisions in the workflow specifications and provide a suitable level of abstraction to capture and organize intellectual concerns and map them to the workflow specification and execution semantics. In this paper, we present Omnispective Analysis and Reasoning (OAR), a novel framework for providing the above features and enhancements in scientific workflow management systems and processes. The OAR framework is aimed at supporting effective capture and reuse of intellectual concerns in workflow management.

## Categories and Subject Descriptors

D.2.m [**Software Engineering**]: Miscellaneous—*scientific workflows*

## General Terms

Theory

## Keywords

Scientific workflows, Omnispective Analysis and Reasoning, Intellectual concerns, Context

## 1. INTRODUCTION

Various scientific workflow management systems like Kepler [2], Taverna [15], VisTrails [4] and Triana [7] adhere to the definition of a *scientific workflow* as [12]: "the description of a process for accomplishing a scientific objective, usually expressed in terms of *tasks* and their *dependencies*." Though these systems provide features to design and orchestrate experimental and computational steps in scientific data collection, organization and analysis [18, 14], intellectual effort is inadequately managed due to *focus on low level implementation details*, *limited support for context*, and *inadequate handling of intellectual concerns*. The emphasis on low-level implementation details hinders the *understanding* and *verification* of the *rationale*, *pertinence* and *appropriateness* of workflow orchestration and instrumentation. The scientific workflow specification focuses more on details of data variables, memory allocation and optimization, and system-level tasks of process control and management. Coupled with the *open nature* of science and *data deluge* this takes focus away from the main objective of the scientific activity and obscures the interpretation of unexpected new results [16]. *Context* has been considered only in the limited sense of *conveying a relation within an environment*. Like in business workflow management [13], it is limited to details of execution environments (machines used etc.), users and the computation steps in the workflow. It seems no serious formal attempt to define context in software systems has been done until recently [1]. Context as a formal parameter will enable defining and disseminating the intent and purpose of the workflow specification and execution. Management of provenance is limited to data and processes [8, 3], and little information is available on context in provenance management. Adding context support will improve traceability of workflows to the underlying models and theories. Intellectual concerns (exploratory domain concepts, scientific models, representation of underlying theories and process specifications) form the backbone for the scientific experiment and workflow and are essential for *de novo* examination of the problem. They are not handled well, if at all, in current scientific workflow management systems. Though a scientific workflow may be *verified* (workflow execution adheres to specification), little support exists to *validate* its scientific soundness.

We have developed the Omnispective Analysis and Reasoning (OAR) framework to address these issues by managing all identified workflow concerns in domain-specific prototypes and archetypes at the conceptual, model and execution levels.

## 2. OMNISPECTIVE ANALYSIS AND REASONING (OAR)

We apply the philosophy of the *Domain of Science Model* in the design of the OAR framework in accordance with the

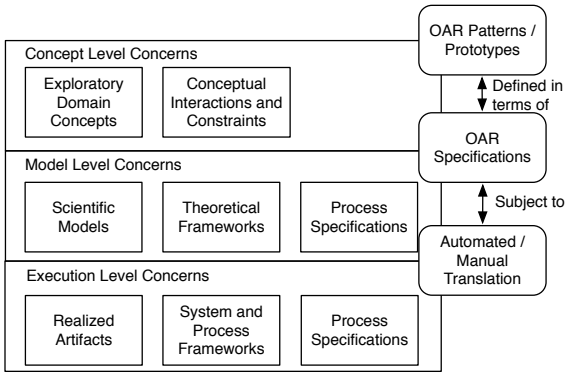


**Figure 1: Concern hierarchy in OAR [6].**

science of Generic Design [17]. Currently, the term "scientific workflow" has been established by ad-hoc usage and lacks precision. It should encompass any process from the simplest to the most involved and should reflect the basic nature of science. Utilizing the *method of science for fixing belief* to include the four universal priors of science [17], we define *a scientific workflow as any logical, systematic and repeatable inquiry, investigation and corresponding set of actions.*

## 2.1 Managing concerns in OAR

In the OAR framework, the problem under study is closely analyzed and the concerns that are relevant to the different disciplinary domains involved are extracted as recipes and managed at three levels (Figure 1). Exploratory domain concerns and their interactions are considered at the *concept level.* Identified knowledge for the concerns of the domain is encapsulated in these recipes to different degrees of firmness. Theories and paradigms which describe the physical and logical systems are abstracted as recipes in terms of mathematical and analytical models, vocabularies, data sets, natural language representations, ontologies and process guidelines. These concerns are abstracted at the *model level.* These abstractions constitute OAR specifications and are defined *exclusively and explicitly in terms of and conforming to* OAR patterns and recipes that have been identified at the concept level. This in turn makes it easy to verify and validate these specifications for conformance and well-formedness. Recipes at the *execution level* constitute the implementation details of OAR specifications in terms of available process specifications, system and process frameworks and known implementation platforms. End-to-end coordination between individual concept level concerns, their model representations and the corresponding implementation and execution in terms of the available platforms and technologies and frameworks, is ensured by the hierarchical nature of the OAR framework.

## 2.2 Concern refinement

All the recipes that are collected in a given domain are *prototypes* and are yet to be analyzed and assessed for their applicability, degree of formalism and robustness for any fitness or purpose. The prototypes may be atomic, or may exist in overlapping groups and the distinction between individual prototypes depends on the context of the problem

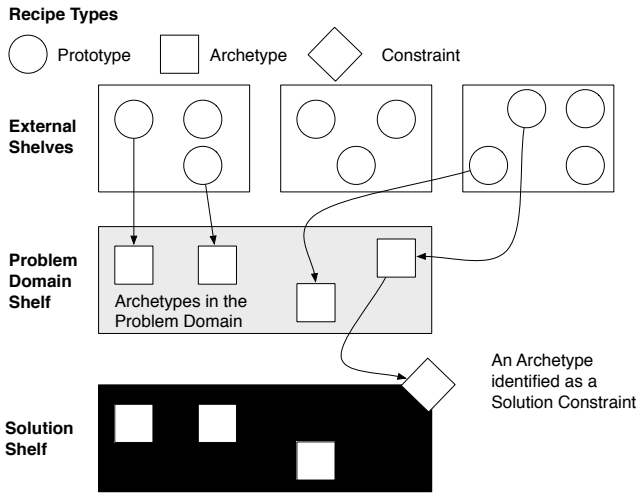


**Figure 2: Managing concerns with recipes and shelves in the OAR framework.**

and the granularity at which we conduct the study. Thus, a prototype can encapsulate either nascent or well-formed domain concerns that may be available to support the analysis of a problem situation with the OAR framework. Depending on the discipline and the area of study, the prototypes can range from rudimentary outlines and sketch-ups to formal blueprints. Analysis of the problem may show that some prototypes can be considered to be *exemplar* or *best practice* recipes. Such recipes are *archetypes* and influence our net understanding of the problem domain. An archetype that is found to impose strict criteria on an OAR specification becomes a *constraint.* A solution to the problem that *sufficiently satisfies* all the requirements of constraints without exception is considered to be a *valid* solution, and is often subject to rigid conformance.

We manage and organize all the recipes by arranging them into unordered collections categorized by domains and their relevance to the problem (Figure 2). These collections, which may contain any number of recipes, are termed *shelves.* Three categories of shelves are used as shown in Figure 2.

*External shelves* hold all the known recipes – concepts, data, data collection procedures, experimental processes, constraints, models, etc. – from different domains of interaction in a reasonably usable form. The *problem domain shelf* holds selections from the external shelves. These selections satisfy given criteria in the problem and correspond to the *best practice* recipes and constitute the understanding of the problem domain. The *solution shelf* contains the archetypes, constraints and the meta-recipes (recipes of recipes) of interconnected specifications of the archetypes relevant to the solution of the problem. Depending on the context, the solution shelf may either be an executable domain or may require further translation.

Based on the approach given by Flint [9], we formulated the following process for concern refinement (Figure 2):

*CNR-1:* Initialize the external shelves with prototypes. This is a bootstrap step and may not be required if we start with

pre-existing external shelves containing domain knowledge.

*CNR-2:* Collect in the problem domain shelf all the prototypes, archetypes and constraints that are relevant to the problem. These are identified from the various external shelves. All of the recipes identified in this step may not be needed in the solution specification.

*CNR-3:* Analyze the archetypes in the problem domain using context refinement to obtain a solution specification in the solution shelf. This step identifies the relationships between the problem domain archetypes and constraints and consolidates the solution specification.

External shelves need not represent full understanding and representation of all domains. Problem domain shelves facilitate localized ontologies which will be good enough for the particular problem scenario even if they may be inadequate for universal use. The solution shelf removes any ambiguity since it captures all identified recipes and constraints.

## 2.3 Context refinement

Extending foundations proposed in earlier work [1], context is managed in OAR as a function of two dimensions: *firmness* and *influence*.

*Firmness* is a measure of the degree of well-formedness of a recipe. If the recipe is ambiguous or vague, the knowledge encapsulated therein is *pliable*. An explicitly defined and well-formed recipe can be considered *firm*.

*Influence* is a measure of the effect exerted by the prototype in the analysis of the problem domain.

If a prototype in an external shelf exerts a *strong* influence on the analysis, then it is identified to encapsulate *exemplar criteria* for the problem situation, and can be considered as an *archetype* in the problem domain. If a prototype, though considered relevant, does not affect the problem domain, then its influence is considered to be *weak*.

OAR recipe context ($C$) is defined as a continuous function of *influence* ($I$) and *firmness* ($F$) (Figure 3) as $C = f(I, F)$. No *a priori* assumption is made regarding the influence and firmness of the recipes. If analysis suggests the use of a particular recipe, then it is identified as exerting a non-zero influence. If the recipe is a *best practice* in the discipline, then it is identified as firm. Consequently, the specification composed from the selected recipes will become increasingly *firm* as situational and imposed constraints are satisfied. Strict adherence to constraints and archetypes will ensure *uniqueness* of the solution.

Though $I$ and $F$ may take any value in the range $[0, 1]$, we find it convenient for purposes of prototype selection to specify context by the following four discrete labels:

$C(I = 0, F = 0)$ : weak influence and low firmness.

$C(I = 0, F = 1)$ : weak influence and high firmness.

$C(I = 1, F = 0)$ : strong influence and low firmness.

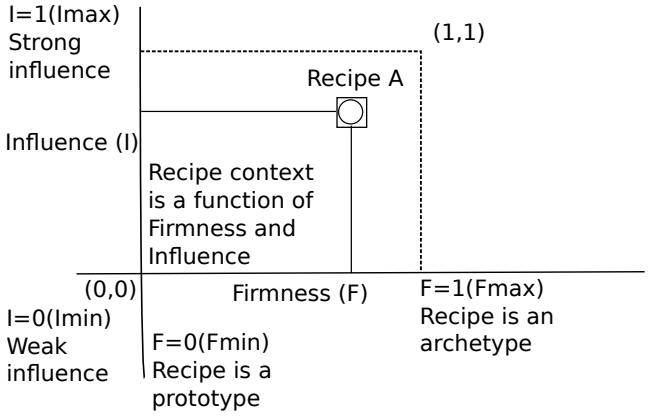


**Figure 3: OAR recipe context as a function of Firmness and Influence.**

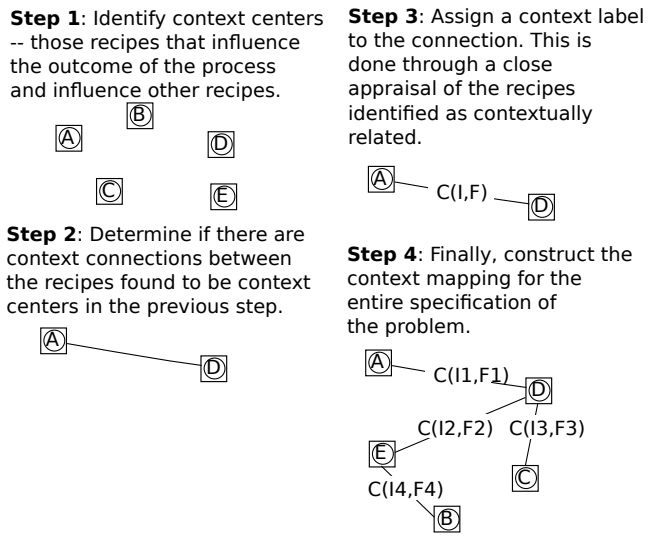


**Figure 4: Context refinement in OAR [6].**

$C(I = 1, F = 1)$ : strong influence and high firmness.

Context refinement (Figure 4) determines recipe relevance. The first two steps of context refinement may be carried out recursively to obtain a solution specification.

# 3. OAR ORIGAMI SPECIFICATION

Origami folding demonstrates many of the characteristics of scientific workflows [11], making it suitable to illustrate the OAR framework. The *folds*, *bases* and the *sequence* of steps are all well-defined and they constitute recipes in the workflow. Ordering in the folding process displays the feature of contextual relation between the steps and highlights the interactions and constraints at play.

## 3.1 Iris flower workflow

The iris flower is a traditional origami construct [10], built either from a *preliminary* or a *frog base*. We first identify prototypes satisfying the folding vocabulary and procedures. The *Fold* external shelf presents us recipes for instructions for modifying the shape of the paper. Paper type affects

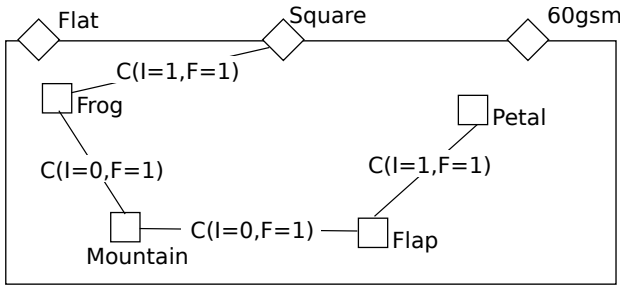


**Figure 5: Origami iris flower specification.**

the ease of folding, and is selected from prototypes in the *Paper* external shelf. Recipes for the *Preliminary* and *Frog* base are selected from the *Base external shelf*. We select the *Flat* technique of folding to implement the workflow on a tabletop surface. It is easier to fold the iris construct with lighter (*60gsm*) paper. A solution specification (Figure 5) is defined using context refinement:

*SS-1:* The Frog base archetype is implemented using the Flat technique with a Square paper. Therefore, this is a constraint archetype that exerts a high degree of influence on the workflow: Flat $C(I = 1, F = 1)$ Square ;
Square $C(I = 0, F = 1)$ Frog

*SS-2:* Although the Preliminary base can also be used as a starting point for the iris flower, it is not as convenient as starting with the Frog base: Preliminary $C(I = 0, F = 1)$ Iris ; Frog $C(I = 1, F = 1)$ Iris

*SS-3:* The iris petals can be formed by folding the Frog base further along the flaps: Frog $C(I = 1, F = 1)$ Petal

*SS-4:* We form four symmetric petals in order to construct the iris flower: Petal $C(I = 1, F = 1)$ Iris

This is translated into execution level by concern and context refinement, and implemented in accordance with a translation archetype.

# 4. SUMMARY AND FUTURE DIRECTIONS

We have introduced the Omnispective Analysis and Reasoning (OAR) framework for capturing and managing intellectual concerns in scientific workflows. All domains and concerns that are likely to influence analysis are identified at the concept, model and execution levels and managed in external, problem domain and solution shelves. Initially all concerns are in the external shelves. Only those recipes which have the desired influence and firmness are placed in the problem domain shelf. The solution shelf consists of recipes which are specifications in terms of and conforming to archetypes in the problem domain shelf. Depending on the context, a solution shelf may either be an executable domain or may require further translation. An example workflow from origami is presented.

The generic nature of OAR formulation makes it applicable to diverse domains. We have applied the framework to contextualizing the design and implementation of a software engineering course using the Moodle Learning Management System [5, 6]. We are also developing tool support for concern and shelf management in the OAR framework.

# 5. ACKNOWLEDGMENTS

This work is based on initial research supported by the Australian National University, and the Commonwealth of Australia, through the Cooperative Research Centre for Advanced Automotive Technology. Srinivas would also like to thank V.Ganesh for his many helpful comments.

# 6. REFERENCES

[1] Z. Alshaikh and C. Boughton. The context dynamics matrix (CDM): an approach to modeling context. In *Proceedings of the 16th Asia-Pacific Software Engineering Conference*, pages 101–108, 2009.

[2] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludäscher, and S. Mock. Kepler: an extensible system for design and execution of scientific workflows. In *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, pages 423–424, June 2004.

[3] R. Barga, Y. Simmhan, E. C. Withana, S. Sahoo, J. Jackson, and N. Araujo. Provenance for scientific workflows towards reproducible research. 2010.

[4] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva, and T. V. Huy. Managing the evolution of dataflows with VisTrails. In *Proceedings of the 22nd International Conference on Data Engineering*, 2006.

[5] S. Chemboli. Contextualising learning outcomes and course design in moodle. In *Moving Up. Moodleposium AU 2010*, Canberra, Australia, Oct. 2010.

[6] S. Chemboli and C. Boughton. Contextual course design with omnispective analysis and reasoning. In G. Williams, N. Brown, M. Pittard, and B. Cleland, editors, *Changing Demands, Changing Directions. Proceedings ascilite (to appear)*, Hobart, 2011.

[7] D. Churches, G. Gombas, A. Harrison, J. Maassen, C. Robinson, M. Shields, I. Taylor, and I. Wang. Programming scientific and distributed workflow with triana services. *Concurrency and Computation: Practice and Experience*, 18(10):1021–1037, Aug. 2006.

[8] S. Davidson, S. Cohen-Boulakia, A. Eyal, B. Ludäscher, T. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 32(4):44–50, 2007.

[9] S. Flint. A model-driven approach to systems-of-systems engineering. In *Proceedings of the Systems Engineering Test and Evaluation Conference (SETE 2008)*, Sept. 2008.

[10] E. Kenneway. *Complete Origami*. Ebury Press, London, 1987.

[11] R. J. Lang. A computational algorithm for origami design. In *Proceedings of the twelfth annual symposium on Computational geometry SCG'96*, pages 98–105. ACM Press, 1996.

[12] B. Ludäscher, M. Weske, T. McPhillips, and S. Bowers. Scientific workflows: Business as usual? *Business Process Management*, page 31–47, 2009.

[13] H. Maus. Workflow context as a means for intelligent information support. volume 2116, pages 261–274.

Springer Berlin / Heidelberg, 2001.

[14] T. McPhillips, S. Bowers, D. Zinn, and B. Ludäscher. Scientific workflow design for mere mortals. *Future Generation Computer Systems*, 25(5):541–551, 2009.

[15] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics (Oxford, England)*, 20(17):3045–3054, Nov. 2004. PMID: 15201187.

[16] W. Steffen. Surviving the anthropocene: Changing our interaction with the fragile planet : Sensible ways forward in response to global change, Oct. 2008.

[17] J. N. Warfield. *Science of Generic Design: Managing Complexity Through Systems Design*. Iowa State University Press, 2 edition, 1994.

[18] M. Weske, G. Vossen, and C. B. Medeiros. Scientific workflow management: WASA architecture and applications. Technical report, 1996.