.

ascilite 2011
CHANGING DEMANDS,
changing directions.
Wrest Point | Hobart Tasmania Australia | 4–7 December 2011

Contextual Course Design with Omnispective Analysis and Reasoning

Srinivas Chemboli
Research School of Computer Science
The Australian National University, Australia

Clive Boughton
Research School of Computer Science
The Australian National University, Australia

In this paper, we present a novel approach to contextualize course design by the application of the Omnispective Analysis and Reasoning (OAR) framework to map the goals and intent of a course to its design and delivery. Effective design and delivery of courses requires alignment between planned learning activities and learning outcomes. However, it is generally not trivial to translate learning outcomes into course design, particularly so when using a Learning Management System (LMS). This is further compounded by the differences between the language of teaching theory and that used by the LMS. Thus, there is a need to effectively capture the rationale for design decisions in a course and map them to desired outcomes. We illustrate the application of the OAR framework with a process for translating a learning objective into course design using the Moodle LMS.

Keywords: Context, Course Design, Moodle, Omnispective Analysis and Reasoning

Introduction

Teaching a course requires planning learning outcomes and goals. These are then mapped to existing educational technologies to deliver the course. However, current learning management technologies often require significant effort to design and organize courses. As a result, teachers are forced to concentrate their efforts on the quirks and methods of the underlying technological platform and this takes their focus away from the goals and outcomes of the course. The activities of online assessment and evaluation tend to become more of a chore, rather than means to support the goals and objectives of a course. This also makes it difficult to share ideas and educational objects across courses and adapt them in a reusable manner.

These difficulties can be overcome by adopting a process that will enable us to:

1. Formalize the implicit rationale in learning outcomes for a course and program of study.
2. Better manage the *intellectual effort* (underlying concepts, ideas and theories) in the design and implementation of a course.
3. Explicitly capture and manage the learning *context* associated with the effort.

To this end, we utilize Omnispective Analysis and Reasoning (Chemboli, 2010a) to develop a reusable specification (recipe) in order to enable flexible organization and management of courses using a learning management system such as Moodle. First, learning objectives and outcomes of a course are captured using an enhanced model of curriculum design. These concept level concerns are then expressed as *model level* concerns in terms of existing educational models in the form of a specification, which conforms to and is in terms of the concept level concerns. Finally, the design decisions of a course are implemented using a learning management system and associated educational technologies, enabling a two way mapping between the educational goals and resources in the underlying technological platform. This will not only enable teachers to map their course outcomes to the underlying educational technology, but will also facilitate evaluation of the effectiveness and suitability of various teaching goals and outcomes.

The remainder of this paper is organized as follows. In the following section we discuss the issues encountered in translating learning outcomes to course design, especially when using a Learning Management System (LMS) for course delivery. Next, we present an overview of the Omnispective Analysis and Reasoning framework, highlighting its main features and approach to concern and context refinement. This is followed by a discussion of the OAR process for contextualizing course design. We illustrate the application of the OAR framework with a process for translating learning outcomes into course design using the Moodle LMS, with particular focus on one learning objective to provide further detail. In the end, we give summary and conclusions and indicate direction of future work.

The Problem of Translating Learning Outcomes to Course Design

The application of Biggs' theory of constructive alignment (Biggs, 2003) to design course content results in an effective approach to course delivery. Constructive alignment requires that the teacher align the *planned learning activities* with the *learning outcomes* (Houghton, 2004). However, one often encounters mismatch and difficulty in mapping the *language of the learning outcomes* to the *resources* and *processes* of the Learning Management System (LMS). This mismatch is mainly because of the different natures of the domains of course design and learning outcomes and the LMS. For instance, while learning objectives for a course may be formulated in terms of the desired goals and means for assessment and evaluation of effective learning, the implementation of the course in an LMS is undertaken in terms of LMS resources such as lessons and exercise blocks, and widgets like buttons and checkboxes, accompanied by participatory student activities such as discussion forums, quizzes and chats. Therefore there is a need to provide a link between the pedagogical concepts and their implementation in an LMS such as Moodle (Rice, IV, 2007). Online instruments will further help this linking of Moodle activities and resources to learning outcomes.

Developing Outcomes for a Course

Courses are driven by outcomes and objectives. Developing an effective course requires answering the following questions:

- What are the learning outcomes for the course?
- What is rationale of the learning outcomes?
- What objective measures can be formed to assess student learning?

These learning outcomes are usually expressed as a *course description statement*.

Difficulty and Disconnect in Mapping Outcomes of Activities / Resources in a Learning Management System

We encounter the following difficulties while translating the intent and learning outcomes of a course into activities and resources in an LMS implementation:

- Resolving the differences between language structure and terminology used in defining learning outcomes and the Moodle implementation.

- There is an apparent disconnect between stating goals of a course and developing a course in Moodle.
- Lack of an effective means to capture the *rationale* for design decisions in a Moodle implementation (which activity / resource is chosen and why, what are the tradeoffs in using different resources to achieve the same outcome).
- Communication and collaboration issues that emerge in the interaction between course designers, lecturers and LMS developers.

Learning Outcome Rationale

The preliminary process of understanding the rationale for learning outcomes begins with identifying relevant *concerns* of interest in the course under consideration. We apply the Omnispective Analysis and Reasoning (OAR) framework to analyze the desired learning outcomes against accepted exemplars (*external prototypes*) presented by Ramsden (1992) and Biggs (2003) as applicable to the context of research-based education (*constraint factor*) at the ANU (Strazdins, 2007). We elaborate the process of linking this analysis in the example discussed later in this paper. This results in capturing the basis for course design in a repeatable manner which can be mapped to the outcomes of the course.

The Omnispective Analysis and Reasoning (OAR) Framework

The OAR framework has been developed to address the problem of capturing and reusing the intellectual effort in scientific workflows (Chemboli, 2010a). In this framework, a scientific workflow is defined as any logical and repeatable inquiry, investigation and set of actions.

The main features of OAR are:

1. Omnispective: The framework provides an outward looking rich context perspective for analyzing and reasoning about the underlying concerns in a problem situation.
2. Epistemic: Provides explicit support for evidential validation in a repeatable manner.
3. Managing intellectual effort: Capturing and reusing theories, concepts, interactions and constraints in recipes and prototypes. Exploratory domain concepts, scientific models, representation of theories and process specifications are represented in recipes (prototypes, archetypes and constraints).

Managing Concerns in OAR

In OAR, all the concerns that are identified in the problem domain are collected and managed at three levels – the concept, model and execution levels. Figure 1 illustrates the idea of hierarchical concern management in the OAR framework.

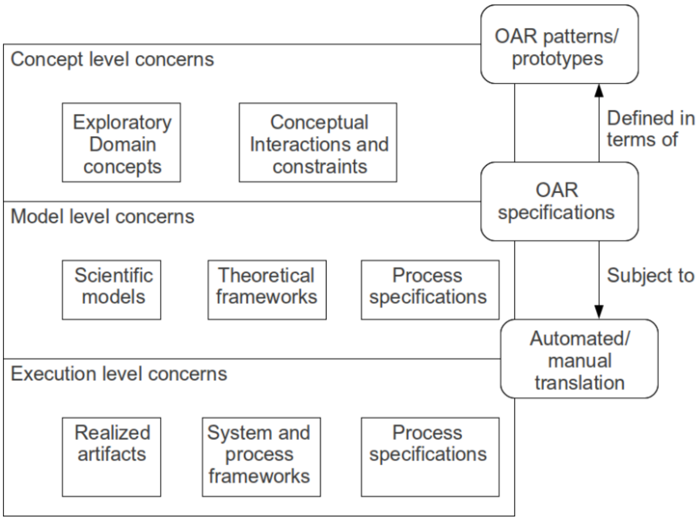

Figure 1: Hierarchical concern management in Omnispective Analysis and Reasoning

All individual concerns that emerge from an analysis of the domain of the present investigation (Scientific Activity) are extracted as recipes. At the *concept level*, these include exploratory domain concepts and the interactions and constraints between them. These aspects of the domain are represented in recipes (OAR patterns and prototypes). Each OAR prototype encapsulates the identified knowledge for a concern in the domain to varying degrees of firmness. At the *model level*, we describe physical and logical systems in terms of formalisms incorporating physical theories and paradigms. These are abstracted in mathematical and analytical models, vocabularies, data sets, natural language representations, ontologies and process guidelines. These abstractions (OAR specifications) are *defined explicitly in terms of and conforming to* the OAR patterns and recipes which have been identified at the model level. Since the OAR specifications are always defined exclusively in terms of OAR patterns, they will be easy to verify and validate for conformance and well-formedness. Finally, at the *execution level* the OAR specifications are implemented by automated/manual translation utilizing different available process specifications, system and process frameworks and known implementation platforms while *satisfying the requirements* of the OAR patterns/recipes which constitute the specification detail. Thus, the hierarchical nature of the OAR framework will ensure an end-to-end coordination between individual concept level concerns, their model representations and the corresponding implementation and execution in terms of available platforms, technologies and frameworks.

Concern Refinement

Depending on the degree of *firmness* and relevance, we categorize OAR recipes in three types – *prototypes*, *archetypes* and *constraints*. A *prototype* is any OAR recipe that is available in a given domain without particular consideration of its applicability, degree of formalism or robustness for any fitness or purpose. Thus, a prototype can encapsulate either nascent or well-formed domain concerns that may be available to support the analysis of any problem situation with the OAR framework. An *archetype* is a prototype that can be considered an *exemplar* or a *best practice* recipe for a concern in a particular domain. The choice of an archetype is dependent upon the analysis of the problem under consideration and influences our net understanding of the problem domain. A *constraint* is a special instance of an archetype which is identified to impose strict criteria on an OAR specification. A *valid* problem solution is required to *sufficiently satisfy* all requirements of the constraint without exception, and is often subject to rigid conformance.

Individual concerns (recipes and patterns) in OAR are managed in collections known as a *shelf*. A shelf is simply an unordered collection of recipes categorized by domains and their relevance to any problem under consideration using the OAR framework. As illustrated in Figure 2, OAR recipes (prototypes, archetypes and concerns) are organized in three categories of shelves – *external*, *problem domain* and *solution*.

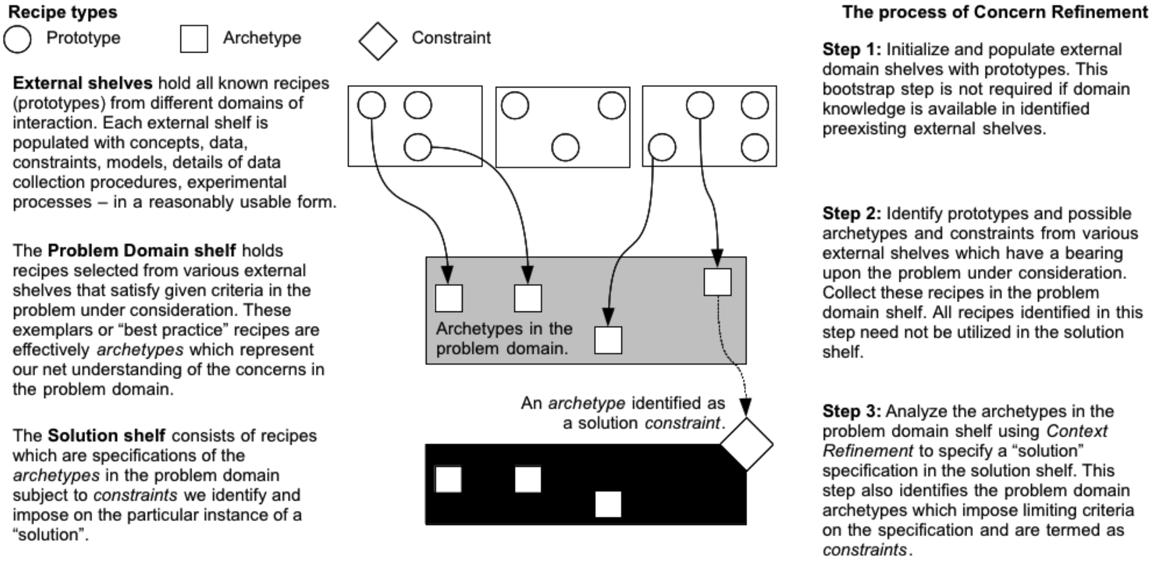
Recipe types
Prototype
Archetype
Constraint
External shelves hold all known recipes (prototypes) from different domains of interaction. Each external shelf is populated with concepts, data, constraints, models, details of data collection procedures, experimental processes – in a reasonably usable form.
The Problem Domain shelf holds recipes selected from various external shelves that satisfy given criteria in the problem under consideration. These exemplars or "best practice" recipes are effectively archetypes which represent our net understanding of the concerns in the problem domain.
The Solution shelf consists of recipes which are specifications of the archetypes in the problem domain subject to constraints we identify and impose on the particular instance of a "solution".
Archetypes in the problem domain.
An archetype identified as a solution constraint.
The process of Concern Refinement
Step 1: Initialize and populate external domain shelves with prototypes. This bootstrap step is not required if domain knowledge is available in identified preexisting external shelves.
Step 2: Identify prototypes and possible archetypes and constraints from various external shelves which have a bearing upon the problem under consideration. Collect these recipes in the problem domain shelf. All recipes identified in this step need not be utilized in the solution shelf.
Step 3: Analyze the archetypes in the problem domain shelf using Context Refinement to specify a "solution" specification in the solution shelf. This step also identifies the problem domain archetypes which impose limiting criteria on the specification and are termed as constraints.

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

Context Refinement

The determination of which recipes are relevant to a problem domain is carried through the process of context refinement. Context specifies the degree of relevance of individual recipes in the problem domain. We manage context in the OAR framework by utilizing an enhanced model of context which extends the foundations proposed by Alshaikh and Boughton (2009). In the OAR framework, we consider context as a function of two dimensions: *firmness* and *influence*.

Firmness is a measure of the degree of wellformedness of the recipe (prototype) under consideration in a shelf. If the prototype is ambiguously defined, vague, or peppered with implicit characteristics, the knowledge encapsulated in the prototype can be considered to be quite *pliable*. On the other hand, an explicitly defined and well-formed prototype can be considered to be *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 is identified as relevant to the problem domain, then the prototype exerts a non-trivial influence on the analysis. If it exerts a *strong* influence on the analysis, then the prototype is identified to encapsulate "exemplar criteria" or "best practice" for the problem situation, and can be considered as an *archetype* in the problem domain. If the prototype does not affect the problem domain, but may still be considered relevant, it is identified as exerting *weak* influence.

Considering these two dimensions, OAR recipe context (C) is defined as a function of *influence* (I) and *firmness* (F) (illustrated in Figure 3) as C = f(I, F).

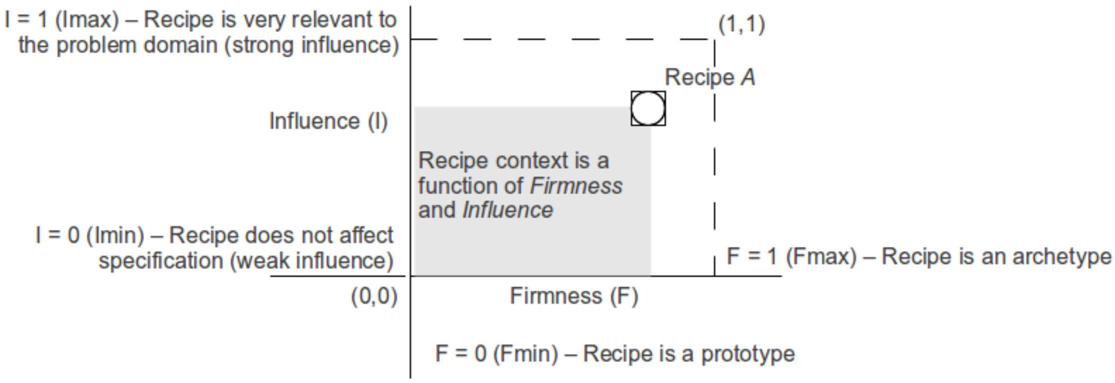
I = 1 (Imax) – Recipe is very relevant to the problem domain (strong influence)
(1,1)
Recipe *A*
Influence (I)
Recipe context is a function of *Firmness* and *Influence*
I = 0 (Imin) – Recipe does not affect specification (weak influence)
F = 1 (Fmax) – Recipe is an archetype
(0,0)
Firmness (F)
F = 0 (Fmin) – Recipe is a prototype

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

At the outset, all recipes are assumed non-contextual and not relevant to the problem under study. If analysis suggests the use of a particular recipe, then it becomes contextually relevant for the problem. In addition, if the recipe falls under the category of an "exemplar" or "best practice" in the discipline, then it becomes both relevant and contextually explicit for the problem under consideration. Consequently, the specification composed from the selected recipes will become increasingly *firm* as situational and imposed constraints are satisfied. Though OAR recipe context is a continuous function of two dimensions, in most circumstances, recipe context can be conveniently tagged by discrete labels of the function f(I, F) affecting prototype selection:

1. C(I=0,F=0): a context state corresponding to weak influence and low firmness.
2. C(I=0,F=1): a context state corresponding to weak influence and high firmness.
3. C(I=1,F=0): a context state corresponding to strong influence and low firmness.
4. C(I=1,F=1): a context state corresponding to strong influence and high firmness.

The process of context refinement (Figure 4) is used to determine the relevance and influence of the recipes to a problem domain.

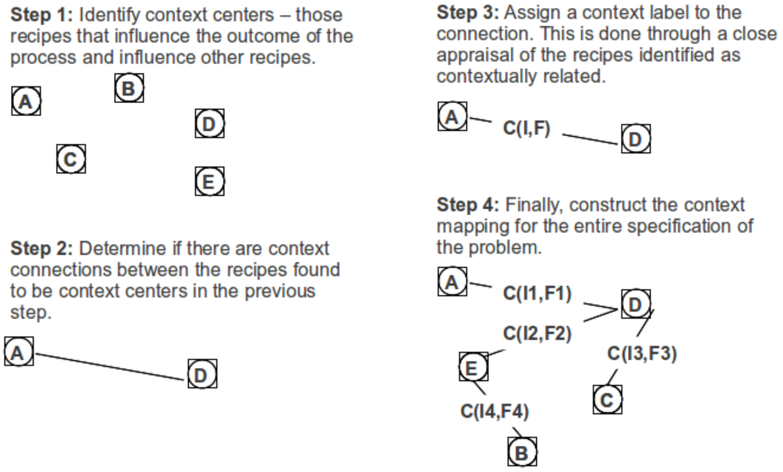
Step 1: Identify context centers – those recipes that influence the outcome of the process and influence other recipes.
A
B
D
C
E
Step 2: Determine if there are context connections between the recipes found to be context centers in the previous step.
A
D
Step 3: Assign a context label to the connection. This is done through a close appraisal of the recipes identified as contextually related.
A
C(I,F)
D
Step 4: Finally, construct the context mapping for the entire specification of the problem.
A
C(I1,F1)
D
C(I2,F2)
C(I3,F3)
E
C
C(I4,F4)
B

Figure 4: Context refinement in the OAR framework

The first two steps of context refinement in Figure 4 may be carried out recursively to obtain a complete mapping of the context relationship in the final specification.

The OAR Process for Contextualizing Course Design

In this section, we demonstrate the application of Omnispective Analysis and Reasoning for developing a contextual specification of course design. First, learning objectives and outcomes of a course are captured using an enhanced model of curriculum design. These *concept level* concerns are then expressed as *model level* concerns in terms of existing educational models in the form of a specification. Finally, the design decisions of a course are implemented using a learning management system and associated educational technologies, enabling a two way mapping between the educational goals and resources in the underlying technological platform.

After identifying the learning outcomes for the course, we execute the following steps:

1. Choose an outcome for the course.
2. Identify the characteristics of the outcome.
3. Using an online instrument (Chemboli, Kane, & Johns-Boast, 2010) to identify the Moodle resource/activity or associated technology which supports the development of these characteristics.
4. Develop the Moodle activity/resource to satisfy the outcome.

Translating Learning Outcomes for COMP8120

COMP8120 was a mandatory course in the Master of Software Engineering Program at The Australian National University. The Master of Software Engineering was aimed at updating and equipping software and systems engineering professionals with a repertoire of best practices and paradigms, empowering them to make informed decisions and evaluate their impact in a measured and systematic manner.

Learning Objectives for COMP8120

1. **[LO-1]** The students will develop an appreciation of past, contemporary and emerging software/systems analysis and design techniques.
2. **[LO-2]** The students will be able to evaluate and understand the scope, applicability, limitations and extensibility of software/systems analysis and design techniques.
3. **[LO-3]** They will be able to apply and expand the methodologies to accommodate appropriate multidisciplinary knowledge, and inform decisions affecting problem situations.
4. **[LO-4]** Students will use the approaches learnt to identify necessary improvements and inform recommendations about their realization using appropriate and well-informed inputs from multiple sources of knowledge and expertise.
5. **[LO-5]** The students will reflect, inform and communicate the issues in the integration approach identified

for improving a problem situation to the satisfaction of all stakeholders identified.

Analyzing Context for [LO-1] and [LO-2]

It is assumed by the designers of the course that students may not be conversant with system/software development methodologies (*ssdm*), or their application to specific problem situations (*ps*). We express this in the following context statement:

student C(I=0,F=0) *ssdm* C(I=0,F=0) *ps*

Students are introduced to subject matter pertaining to the detail and process of various methodologies, *without concern* for specific usage scenarios. Following this activity, students will now possess the requisite knowledge of system/software development methodologies even if they may still not be aware of areas of application of specific methodologies. Thus, students will acquire *explicit* knowledge of methodologies, although they may still be unable to contextually apply them to targeted problem situations. This can be expressed as:

student C(I=0,F=1) *ssdm*

Analyzing Context for [LO-3]

The students participate in guided learning exercises concerning the development of understanding and evaluation of problem situations while developing informed recommendations *without particular emphasis on solution strategies*. This phase of learning is governed by efforts to *credit local knowledge*.

student C(I=1,F=0) *ps* C(I=0,F=1) *ssdm*

Analyzing Context for [LO-4]

The students now undertake activities to iterate through *constraints* that bear upon a particular problem situation and formulate and develop *implementation processes* in accordance with all identified aspects of the problem domain. An early design decision by the course organizers required that the students employ the Aspect-Oriented Thinking (AOT) framework (Flint, 2006) for this purpose. Consequently, the choice of the AOT framework is a firm constraint (no alternatives are considered).

student C(I=1,F=1) *ssdm* C(I=1,F=1) *ps* C(I=1,F=1) *AOT*

Analyzing Context for [LO-5]

The students are to undertake activities to effectively *communicate* and *collaborate* in order to actively arrive at an agreement on the identified intervention in a problem situation.

student C(I=0,F=1) *communication and collaboration*

Solution Specification for [LO-5]

As a further illustration, we now discuss the OAR process for formulating a solution specification for implementing communication and collaboration for [LO-5] using the Moodle LMS (Chemboli, 2010b). Although the process for defining solution specifications for the other learning objectives is not presented here, they are defined in a similar manner.

Initialize External Shelves

We identify the following external shelves (Figure 5) with prototypes for designing the specification for realizing [LO-5].

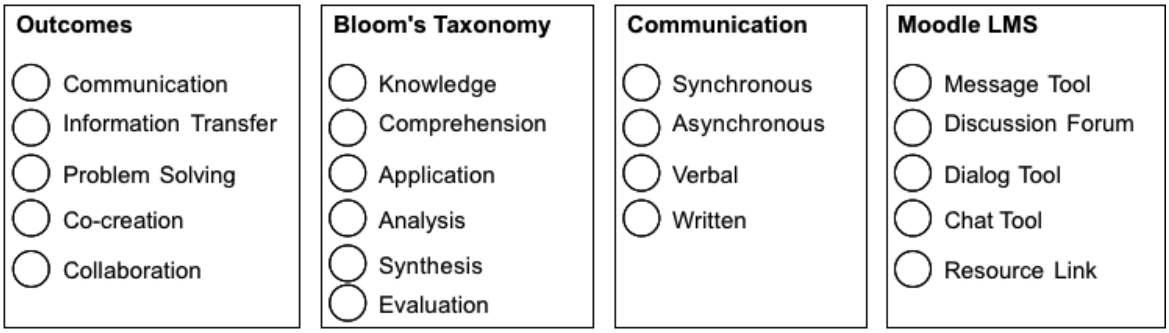

Figure 5: External shelves and prototypes

We utilize Bloom's taxonomy in order to align the learning outcomes of COMP8120 with assessment criteria. Hence, we identify the prototypes for assessment modes in Bloom's taxonomy as relevant recipes for consideration. The Outcomes external shelf presents us recipes for the activities pertinent to [LO-5]. Written or verbal communication between students can either be synchronous (in real-time) or asynchronous. These recipes are collected in the Communication external shelf. Since, we intend to implement [LO-5] using the Moodle LMS, an external shelf with resources and activities supporting communication and collaboration in Moodle is considered.

At the time of designing this course, we used Bloom's taxonomy (Bloom, 1963). There is nothing inherent in the OAR framework which mandates the kind of taxonomy used. The revised Bloom's taxonomy (Churches, 2008) can be incorporated in a straightforward manner as another external shelf which provides additional prototypes for consideration in the solution specification.

Identify Relevant Archetypes and Constraints

Next, we perform concern refinement for [LO-5] in order to select the specific prototypes which can be imported into the problem domain shelf. [LO-5] states the requirement for *active* communication and collaboration. The recipes for Communication and Collaboration are selected from the Outcomes external shelf. We also identify two constraint recipes – *Synchronous* and *Written* from the Communication external shelf. The students are required to communicate and collaborate in a synchronous fashion using written tools (these constraints are imposed by the course designers). Utilizing Bloom's taxonomy, we can align [LO-5] with assessment of Application skill. The students are required to utilize the communication platform to execute stakeholder engagement. Finally, we also select the prototypes for the activities in Moodle which support communication in a written form. The resultant problem domain shelf is illustrated in Figure 6.

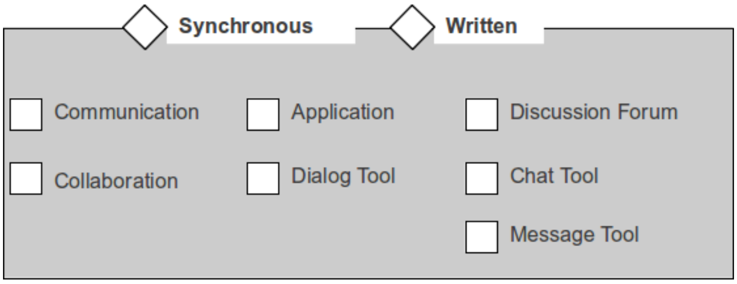

Figure 6: Archetype and constraint identification for [LO-5]

Solution Specification for [LO-5]

We now undertake the process of Context Refinement to define the solution specification for [LO-5] (Figure 7). The process is outlined in the following specification statements.

[SS-1] The *Communication* archetype needs to be implemented in a *Synchronous* and *Written* fashion. *Synchronous* C(I=1F,=1) *Communication* C(I=1,F=1) *Written*.

[SS-2] Although the Dialog Tool, Discussion Forum and Message Tool can be used for synchronous communication, they are not particularly suited for interactive text exchange between participants. Therefore, only the Chat Tool satisfies the requirement of synchronous communication, and is imported in the solution shelf.
Synchronous C(I=1,F=0) *Chat Tool.*

[SS-3] The Chat Tool was selected because it is a writing tool.
Written C(I=0,F=1) *Chat Tool.*

[SS-4] The use of the Chat Tool will satisfy the requirement for assessing how the students apply the skills learnt via [LO-5] because the course coordinators can assess the chat logs for significant contributions. Thus, we can state this as:
Chat Tool C(I=0,F=1) *Application.*

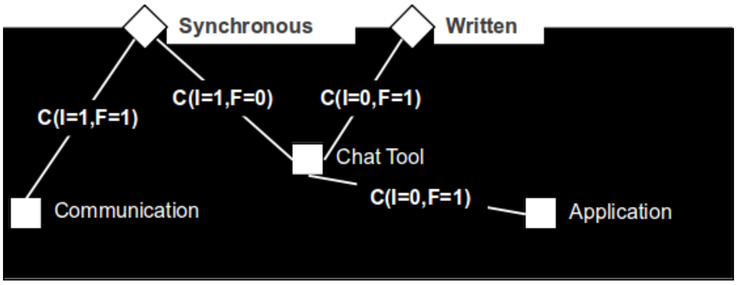

Figure 7: Solution specification for [LO-5]

Implement Solution Specification

Finally, we implement the solution specification for setting up a chat tool for satisfying [LO-5]. This is achieved through either manual or automated translation using the information provided by the archetype for the chat tool. In this case, we utilize a Moodle workflow for setting up the chat activity in the LMS (Chemboli, 2010c), which is specified in the chat tool recipe that was imported from the Moodle external shelf during the process of concern refinement. The use of the chat tool satisfies the design requirement for [LO-5] (synchronous and written communication) which is specified in Figure 7.

Summary and Conclusions

In this paper, we have presented the application of the Omnispective Analysis and Reasoning (OAR) framework to contextualize course design by mapping the goals and intent of a course to its development and delivery. This is a novel approach and we expect this will support the development of courses which robustly satisfy their learning outcomes. We align the learning outcomes of a course with activities related to teaching and assessment by explicitly capturing the context of course design using an enhanced context paradigm. We have illustrated this process by the example of a specification to satisfy a learning outcome for synchronous and written collaboration between students in a software engineering course using the Moodle LMS. We expect that adding support for capturing and incorporating context and rationale using the OAR framework will greatly aid the process of course design. Though we have not explicitly addressed the idiosyncrasies of learners and instructors in the example presented, these can be incorporated by adding prototypes from relevant external shelves. As a further extension to the work presented in this paper, we are currently developing a “Context Plugin for Moodle”. This is an OAR workflow plugin to capture and represent the intellectual effort in course design and manage its context within the Moodle LMS itself.

References

Alshaikh, Z., & Boughton, C. (2009). The Context Dynamics Matrix (CDM): An Approach to Modeling Context (pp. 101-108). Presented at the 16th Asia-Pacific Software Engineering Conference, Penang, Malaysia. Retrieved from http://dx.doi.org/10.1109/APSEC.2009.74

Biggs, J. (2003). *Teaching for Quality Learning at the University: What the Student Does*. Open University Press.

Bloom, B. S. (1963). *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook 1: Cognitive Domain*. New York: D. McKay.

Chemboli, S. (2010a, February 25). *Omnispective Analysis and Reasoning: An epistemic approach to scientific workflows*. Presented at the CECS Seminar Series, School of Computer Science, The Australian National University.

Chemboli, S. (2010b). Contextualising learning outcomes and course design in Moodle. *Moodleposium AU 2010*. Presented at the Moodleposium AU, Canberra, Australia. Retrieved from http://moodleposium.netspot.com.au/

Chemboli, S. (2010c, July 19). Workflow for creating a chat activity in Moodle. Retrieved June 25, 2011, from http://csrins.wordpress.com/2010/07/19/workflow-for-creating-a-chat-activity-in-moodle/

Chemboli, S., Kane, L., & Johns-Boast, L. (2010). Translating Learning Outcomes in Moodle. Presented at the MoodleMoot AU, Melbourne, Victoria, Australia. Retrieved from http://www.usq.edu.au/~/media/USQ/learnteach/Events/JOHNS_BOAST_USQ_wattleWorkFlows_presentation.ashx

Churches, A. (2008). Bloom's Digital Taxonomy. Retrieved August 11, 2011, from http://edorigami.wikispaces.com/Bloom%27s+Digital+Taxonomy

Flint, S. (2006). *Aspect-Oriented Thinking -- an approach to bridging the interdisciplinary divides*. The Australian National University. Retrieved from http://hdl.handle.net/1885/49328

Houghton, W. (2004). *Engineering subject centre guide: Learning and teaching theory for engineering academics*. Loughborough: HEA Engineering Subject Centre. Retrieved from http://www.engsc.ac.uk/learning-and-teaching-theory-guide

Ramsden, P. (1992). *Learning to Teach in Higher Education*. Routledge.

Rice, IV, W. H. (2007). *Moodle Teaching Techniques: Creative Ways to Use Moodle for Constructing Online Learning Solutions*. Packt Publishing.

Strazdins, P. (2007). *Research-Based Education in Computer Science at the ANU: Challenges and Opportunities*. The Australian National University. Retrieved from http://cs.anu.edu.au/techreports/2007/TR-CS-07-05.pdf

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. Support from the Research Student Development Centre, the CECS Educational Development Group and the Division of Information at the ANU is gratefully acknowledged.

Author contact details:
Srinivas Chemboli srinivas.chemboli@anu.edu.au
Clive Boughton clive.boughton@anu.edu.au

Please cite as: Chemboli, S. & Boughton, C. (2011). Contextual Course Design with Omnispective Analysis and Reasoning. In G. Williams, P. Statham, N. Brown, B. Cleland (Eds.) *Changing Demands, Changing Directions*. *Proceedings ascilite Hobart 2011*. (pp.210-219). https://doi.org/10.14742/apubs.2011.1774