Problem Oriented Engineering in action: experience from the frontline of postgraduate education

Lucia Rapanotti
J.G.Hall

1st October 2008

Department of Computing
Faculty of Mathematics and Computing
The Open University
Walton Hall,
Milton Keynes
MK7 6AA
United Kingdom

http://computing.open.ac.uk
Abstract

The paper reports on the early phases of a development project aimed at a highly innovative post-graduate research programme for the Open University, UK, a world leader in supported distance higher education. The new programme is a part-time MPhil to be delivered at a distance, supported by a blend of synchronous and asynchronous technologies. After a brief description of the project and the complexity of the task at hand, the paper discusses how Problem Oriented Engineering, an emergent engineering design framework, was adopted on the project, and its performance in this complex real-world setting. The outcome of the study will be of relevance to practitioners who wish to adopt the approach in tackling real-world development problems, as well as researchers who are striving to improve the framework for practical adoption.

2. The problem, its context and its requirements

The eMPhil is required to meet a number of research objectives, set by the Computing Department as part of their overall research strategy, to:

- enhance and develop the department’s provision to their graduate community;
- promote wider use of technology to support research students at a distance, especially part-timers;
- create cohorts of research students on specific research themes and projects;
- attract more research students, particularly those who feel that they cannot commit the time to a PhD;
- increase the overall amount of research supervision that takes place within the department, particularly involving staff who are not currently supervising research students.

Already the Computing Department has a lively community of residential full-time doctoral students who, besides having access to their supervisor(s), are given an opportunity to take part in a structured programme of induction and training, facilitated by senior academics and highly experienced doctoral supervisors. The current programme aims at developing all the required research skills [5], and also
functions as a forum for the student community and as a peers’ aid and study group, for which sessions are run on a weekly basis throughout most of the academic year. The programme also fits within the overall university approach to research student induction and training and is compliant with national standards [6] the university has subscribed to. Therefore as well as meeting the Department’s stated objectives, the eMPhil was required to comply with external standards, and to provide a comparable induction and training programme for part-timers at a distance.

There are important social aspects to doing research, which the eMPhil was required to support, such as students be given opportunities to interact with peers and to work collaboratively, as well as to develop friendships and a sense of belonging. Unfortunately, part-time distance students cannot always take advantage of the residential induction and training programme, due to the cost and time that travelling to a campus would require.

A part-time distance MPhil programme is already in place at the Open University (OU), which is highly regulated and monitored, and which details the required practices for both supervisors and students. This programme include codified practices which stipulate a certain amount of face-to-face supervision and training for the students, therefore requiring them to travel to our campus on a number of occasions throughout the year. Whilst the eMPhil was required to adhere to all necessary current regulation, the intention was, through it, to lift the burden of travel for the students, while enhancing their online interaction with both supervisors and their wider research community.

The OU, as a world leader in supported distance higher education, has a long tradition of technological innovation in support of students, a recent example being the adoption of Moodle [7], an Open Source eLearning environment, as a platform for our on-line provision. The OU has invested heavily in making the technology robust as well enhancing it (please see the OU VLE project at [8] for further details) through a series of early adopters projects and deployment on selected courses as test cases. The state-of-play is that the platform is now deemed ready for wider adoption by the organisation, so this factor was taken into consideration in the development of the eMPhil infrastructure.

Similarly, the OU has been experimenting for some time with Second Life [9], a commercial multi-user virtual environment (MUVE) by Linden Labs. Although not initially conceived for education, Second Life has become popular with many global organisations and educational institutions around the world, many of which have developed a Second Life presence of some kind or another. The embodiment of users as avatars in the virtual space holds some promise for eLearning and has attracted the attention of researchers in the field [10, 11, 12, 13]. MUVEs remain, however, a relatively new, hence immature, technology and little empirical evidence exists as to their effectiveness in education. Nevertheless, they are becoming popular and have already shown some potential, and this factor too was taken into consideration in the project.

A study into the feasibility of the undertaking, including the potential market for the product and its main competitors, was conducted at the beginning of 2008. The project started in March ’08, with the project team appointed and some initial planning carried out, coupled with seeking initial approval within the organisation. In June ’08, development started on one of the main software components, which was concluded in September ’08. It is the development process and activities up to this point which are the subject of this paper. The project is still ongoing, with the eMPhil due to be rolled out in October 2009.

The team decided to adopt the Problem Oriented Engineering (POE) framework as a tool to investigate the problem space and guide product development. POE is an emergent framework for engineering design under development at the Open University. From the eMPhil project’s perspective, the framework was attractive in offering a blend of conceptual tools to tackle problem complexity, with the added bonus that POE expertise existed within the project team and in the Department. Of particular attraction to the project was POE’s problem solving view of engineering design, which extends beyond software to processes and organisational issues, as well as POE’s ability to accommodate different practices within a unified conceptual framework. POE had already shown some promise when applied to a variety of development problems, particularly within the safety-critical software industry (see, e.g., [14, 15, 16]). Those studies focused on particularly aspects of development. Instead, the eMPhil project offered a unique opportunity to test the framework on a live mission critical project in its entirety, hence gain useful feedback on its performance in sustained practice.

The next sections will briefly recall the main elements of POE and describe its application to the eMPhil project. It is not our intention to provide a complete presentation of POE, which can be found, instead, by visiting the POE home at [17], where a comprehensive bibliography is also available.

3. POE in theory

POE is a framework for engineering design, intended as a creative, iterative and often open-ended undertaking of conceiving and developing products, systems and processes. In POE’s view, engineering design includes:

- the identification and clarification of the recognised need;
- the understanding and structuring of the real-world
context into which the engineered system will be deployed

- the specification of a design for a solution that can ensure satisfaction of the recognised need in context;
- and the construction of arguments, convincing for all validating stake-holders, that the engineered solution will provide the functionality and qualities that are needed;

POE strives to make provision to support each of them.

The basic building blocks of POE are problems, transformations and justifications, jointly called the **POE triad**. A **POE problem** is a collection of named descriptions, one each for the requirement (the identified need), the solution (the artefact subject of design), and the context (that part if the world where the solution will be located to meet the requirement). Such descriptions hold knowledge about the problem and its parts. As they capture knowledge about the real world, they can be of various nature: from precise and formal to vague and inconsistent, from textual to pictorial, etc. **POE transformations** are design steps, changing problems into other problems and giving rise to a stepwise problem solving process. **POE justifications** allow for stake-holders validation, hence are a tool for managing risks, those of addressing the wrong problem and/or designing the wrong solution.

POE gives equal relevance to both artefact design and its validation. It also advocates early validation as a tool for managing risk, hence to be applied throughout development. These observations, coupled with the transformational nature of problem solving in POE, result in particular design structures, as we will see, that of forests of design and validation trees.

The POE notion of problem is inspired by the seminal work of Jackson, Zave and others [18, 19] on Requirements Engineering (RE), which was concerned with establishing criteria for the completeness of RE processes within software development. POE ads to that work in several ways: it extends beyond requirements engineering to software engineering, and more generally to engineering design; it includes iterations between problem and solution spaces, so that solution artefacts, such as the notion of architecture, are accounted for; it is a constructive framework, in which transformations are used as design steps; it relaxes the notion of proof, too strong for may real-world engineering endeavours, to that of ‘adequacy’ with respect to stake-holders.

### 4. POE in practice

The eMPhil core project team is composed by four academics, with the first author as project leader. The team drives the whole development and has responsibility for identifying and co-ordinating the activities of other teams, both within and outside the Open University, who make specific contributions to the project. For instance, during the first phase of the project, a commercial development team was contracted to carry out some development on the Second Life component of the project, while a development team from one of the production centres from within the OU was employed to work on the Moodle component. In both cases, the core eMPhil team had to draw appropriate specifications and co-ordinate development activities. Similarly, the eMPhil team made use of a team of librarians to identify appropriate services to be provided online to the eMPhil users: this too required negotiation and co-ordination.

In adopting POE on the eMPhil project, we had to acknowledge that a gap existed between POE as a theoretical framework and its practical application. POE is defined as a formal framework, both in its definition and notation. Previous work (see, e.g., [15]) had gone some way towards filling that gap, for instance by adopting a graphical representation of problems to complement their mathematical form, as well as providing a generic template to express justifications in a textual manner. However, no bespoke tool support exists as yet, which raised the issue of how to scale the approach for the eMPhil project, considered its length, the number of stake-holders involved and the amount of documentation likely to be generated.

Despite the many Requirements and Software Engineering tools available, we could not identify an appropriate subset which could in concert support the framework in its entirety—most focus on specific aspects of software development or specific modelling techniques. Also, the idea of switching between a disparate set of tools manually was deemed too cumbersome and time consuming for the project. Therefore, the team resolved to a low cost, low tech approach consisting of the practices explained below. The project leader also kept a project diary, to log any difficulty or other issue raised by the adoption of POE.

#### 4.1. Graphical representation

A convention was adopted for the pictorial representation of problems, their transformations and related justifications, which extended and enhanced the graphical representation adopted previously. Specifically, while POE problems can be represented by using a variant of the Problem Frames notation [20], transformation steps and their justification had no graphical notation defined. Therefore, we designed some templates for this purpose using a commercial graphic tool, which was then used to generate all graphics for the project. An illustration is given in Figure 1, which refers to the design space from the development team’s viewpoint. The ‘design problems’ relate to the
design of the product itself, i.e., the eMPhil; the ‘validation problems’ refers to design validation by relevant stakeholders. As already mentioned, POE sees validation and design as equal partners, so both design and validation spaces have to be explored as part of the problem solving process. From a practical viewpoint, however, we found it more convenient to define a notation where these two representations are brought together, while still acknowledging their theoretical distinction. Each problem includes a representation of context (rectangles), requirement (oval), and solution under design (decorated rectangles).

In the figure, the design problem at the bottom of the horizontal line is transformed into the design problem at the top (this particular POE transformation is called an solution interpretation in that is it used to provide a structure for the to-be-designed solution). To the side of the transformation line are validation problems whose solution provides a justification of adequacy of the step. Validation is the process by which relevant stake-holders give their approval as to the adequacy of a design step, for instance signing off some stated assumptions on the problem context, or the choice of a particular solution architecture. This sign-off process is a way of managing development risks, as lack of approval may signify that, say, stated domain assumptions are incorrect or a proposed solution is unfeasible. Moreover, different stake-holders will be interested in different aspects of the design, and will require that relevant information about the design is made available to them.

Taking the above into consideration, we conceived the following way of representing a validation problem. Its context is the stake-holder (or stake-holder group) with an interest in the related design step. The requirement is a particular aspect of the design the stake-holder has competence for and sign-off authority: we call such a requirement a concern. The solution is the artefact that the stake-holder will inspect to express their opinion as to the adequacy of the design step: we call such a solution the validation artefact, typically some documentation capturing a projection of the design problem in a form accessible to the stake-holder. As is the case in Figure 1, the justification of a design step may require many validation problems to be solved. In the example, the design step aims at producing a cost model for the eMPhil and validation is sought from two stake-holders, the Head of the Computing Department and the Senior Administrator of the Faculty in which the Department is located. Both stake-holders are budget holders and were asked to sign-off the proposed cost model as viable from a financial viewpoint. The validation artefact produced for each of them was the portion of the overall project cost model to be attributed to their respective budgets.

Transformation steps are linked together as problems are transformed to other problems. Figure 2 gives a representation of the eMPhil development at the end of July ’08. The root of the development tree is at the bottom of the figure; particularly noticeable is the high number of validation problems that were addressed at the beginning of the project: these relate to the initial approval sought from within the organisation, and reflect the nature of its administrative structure.

Note that some of the steps introduce sub-problems, which lead to branches in the design tree. This happens when a number of solution components have been identified, each to be designed, together with their architectural relations and mutual constraints. The POE transformation which generates them, called solution expansion, generates appropriate sub-problems for each to-be-designed component, based on such the architectural knowledge. Each sub-
problem then because the root of a (sub-) design tree.

As validation is a problem in itself (see also the discussion on ‘cross-problems’ in [21]), its solution will be arrived at through a problem solving process, which will typically require time. From a purist POE perspective, a validation problem could be treated as any other problem, hence solved through transformational design steps, with an equivalent graphical representation as discussed above. On this project, however, we took a more pragmatic approach and introduced a risk indicator (see Figure 1) for each validation problem, whose associated description provides the necessary information as to the progress made in the solution of the validation problem. Pictorially, risk indicators are colour-coded (this may not be not visible in the figure due to black-and-white rendering): green when sign-off has been obtained; red at the start of the validation process; yellow somewhere in between. Given the lack of tool support, this was deemed a simple, but useful indicator from a project management perspective; of course, a more accurate estimation of risk would have required more sophisticated tools. Figure 3 gives an intuition of the meaning of the risk indicator: to the right is the equivalent fully expanded validation problem.

4.2. Repository

A very large amount of material was produced as descriptions associated with the various design and validation problems. A naming convention was adopted for problems, so that all related artefacts could be classed and clustered accordingly. A repository for the project was then created on a designated machine, simply consisting of a directory structure mirroring the adopted naming convention. All relevant artefacts were kept in digital format within the repository, following the adopted naming convention. For artefacts resulting from conversation or hand-sketches, appropriate digital summary or images were generated and stored. For digital assets kept on remote servers (e.g., the Second Life island, which is hosted by Linden Labs), appropriate pointers (e.g., URLs or SLURs) were kept in the repository. A back up system was used to keep hourly snapshots of the repository, as a permanent record of the development.

A project board was used to maintain a paper copy of the (graphical representation of the) development forest, with
snapshots taken after each transformation step, so that the whole development could be replayed (a fragment is given in Figure 4). Coupled with the naming convention for problems, this provided an easily accessible record of how the development was progressing.

Figure 4. Snapshot of the eMPhil development at the end of July ’08

4.3. Stake-holders

The project involved a large number of stake-holders the team had to interact with: 52 were identified within the first 6 months of development alone, both from within the OU’s various academic, administrative and production departments, and from outside contractors.

As stake-holders play an essential validation role and are named in transformation justifications. As the project progressed and the stake-holders community grew, it became more and more difficult to keep track of names and relations just by looking at the POE tree, therefore the team resorted to maintaining a stake-holders map, so that all stake-holders could be easily identified and their relationship understood. Mind mapping software was used to generate it. A snapshot of the map at the end of July ’08 is given in Figure 4.

4.4. Inclusive practices

POE is a broad church in that it provides a coherent framework in which a great variety of practices and notations can be accommodated. In this respect, it is very different from, say, a development process, such as RUP [22], or a description language, such as UML [23]: different design problems and steps in POE will make use of any combination of descriptions and practices which are deemed appropriate by the problem solver. As an example, the development of the Second Life component of the project required off-campus third-party developers to be contracted by the eMPhil project team, which, in this case, became the customer of such a development. In this relationship, a formal contract was drawn and a range of agile practices [24] were adopted, from small incremental iterations, to user stories as a tool for requirements elaboration, and ‘inworld’ validation sessions to check development progress on a weekly basis. Another example is the interaction between the eMPhil project team and the team of librarians consulted to specify appropriate library services for the eMPhil community. In this case, the two teams were co-located on campus and worked collaboratively: after an initial verbal agreement, the two teams often came together for face-to-face meetings, and made use of a wiki to produce a joint specification.

5. Early evaluation

The experience so far has been very encouraging, and has clearly indicated that the conceptual tools offered by POE, coupled with the practices outlined in the previous section, were able to cater for all relevant aspects of the project. Some of the practices were introduced initially and then refined during this first phase of the project, particularly the graphical representation of validation problems and the stake-holders map. This added some overheads to the management of the project.

The graphical notation adopted was considered very effective in terms of project management. Design forests provide a powerful summary of the development, with all critical decision points clearly exposed, and all sub-problems

1 ‘Inworld’ is the term used in MUVE to indicate users interacting with the 3D environment via their avatars.
and streamlined to be of practically use. The risk indicators, despite their lack of sophistication, were considered very useful in sign-posting critical parts of the development. The notation was also considered an effective communication tool: its relative simplicity and abstraction allowed even non technical stake-holders, like senior managers in academic and academic-related units, to grasp the essence of the project with very little explanation required. The inclusion of validation problems within the development tree, with an explicit acknowledge of all relevant stake-holders, was also considered a valuable tool to gauge the criticality of each design step, as well as to focus attention on the aspects of the problem of significance to each stake-holder.

The separation between the graphical representation and the associated multi-formed documentation was also of great utility: the abstract representation allowed the team to index and give an homogenous structure to the development space; the associated multi-formed documentation allowed the team to add the required detail through appropriate descriptions, whether static (e.g., textual or pictorial) or dynamic (e.g., audio, video or a working system). The documentation being clustered and indexed through the graphic notation meant the team was able to locate information efficiently, and dip in and out looking at information at various levels of abstraction. The stake-holders map was also considered a useful complement to POE.

The granularity of the design steps recorded in the POE tree was greater than that of the basic transformations defined in the theoretical framework, where the design steps are very small. Instead, in practice large steps, combining a number of basic transformations, were deemed more meaningful. This phenomena was not unexpected and confirmed what had already been observed in previous practical applications of POE [2, 16]. Previous experience also highlighted how POE developments seem to proceed through repeated applications of a simple process pattern unit, which compose in sequence, parallel or recursively [21]: this too was also observed during the eMPhil project.

The lack of automated tools for the project was a real drawback of adopting POE on the project. In order to be true to the approach, a great degree of discipline and care was required of the project leader in manually recording artefacts and their transformation, something which could not be normally expected in a commercial development. On the other hand the need to combine the scientific investigation with a life project with concrete milestones and deliverables, meant that the whole approach had to be distilled and streamlined to be of practically use.

6. Future work and Conclusion

Encouraged by the experience so far, the team will continue to use POE on the next phase of the project, following the practices described above, and gathering further data on POE’s performance, and the extent the practices adopted may contribute to fill the gap between theory and practice. Some enhancement of the graphical notation is also planned, by improving the integration between stakeholders map and design forest through appropriate colour coding.

An unexpected, but welcome, outcome of the project is also a set of requirements for a prospective POE tool. The POE research team have been working on various prototypes for the framework for some time (see, for instance [25]). However, this project has provided some very concrete ideas as to which core functions such a tool should possess, as well as some insights into its possible user interface, which the research team will be able to take forward.

Acknowledgments

Many thanks go to the other members of the eMPhil production team at The Open University, namely Leonor Barroca, Marial Vargas-Vera and Shailey Minocha, as well as our brilliant OU librarians, Non Scantlebury and James McNulty. Many people have helped us make POE a better framework. We acknowledge SE Validation Limited for their financial support and, in particular Colin Brain for his many comments and insights. A particular thank you goes to Derek Mannering, at General Dynamics UK, whose work has helped us develop and validate POE for safety engineering. Thanks also go to Jens Jorgensen and Simon Tjell of Aarhus University for much fruitful discussion on how to integrate POElog and CPN Tools and our colleagues in the Computing Department at The Open University, particularly Michael Jackson.

References


