The distributed developmental network - d2n: a social configuration to support design pattern generation

Niall Winters, Yishay Mor, Dave Pratt

This is a pre-print draft of a book chapter, to appear in:
Technology-enhanced learning: Design Patterns and Pattern Languages (eds Peter Goodyear & Simos Retalis). Sense Publishers, Rotterdam.

The distributed developmental network - d²n: a social configuration to support design pattern generation

Niall Winters, Yishay Mor and Dave Pratt

Abstract

DiSessa et al. (2004) conducted a comparative study of how research teams design, develop and evaluate TEL software, in the context of component-based educational programming. They identified the issue of the social configuration of the production team as "a critical family of issues that are easily marginalized" (p.117). These social configurations are loosely equivalent to what Activity Theorists refer to as the rules and division of labour (Engeström, 1987) in the activity system of TEL production. DiSessa et al. (2004) studied four such configurations in detail and noted their relationship with the evolution of the technology and its use. These models suggest different ways of bringing the various participants involved in TEL development together. Based on the definition of interdisciplinarity (van den Besselaar and Heimeriks, 2001; Gibbons, 1994), in this chapter we detail how to support participants from different disciplines to work together in small, product-oriented groups, using design patterns.

Our patterns were developed in the context of the *Learning patterns for the design and deployment of mathematical games* project, funded under the Kaleidoscope Network of Excellence of the European Union. Our primary aim was to develop patterns that worked at the interface between disciplines. They were focused on pragmatic ways to have teachers and technologists productively engage with each other. Furthermore, many patterns were developed from the use of particular tools in educational contexts, where the tools were developed from scratch as outputs of research projects. There was a reflection in the patterns of the need for participants to understand each others' practices in order to achieve integrated development. DiSessa et al. (ibid) reflect on the fact that teachers can find it "difficult and sometimes intimidating to participate as equal contributors in a technology-based development process" and suggest that effective management of collaboration can address this problem.

As distinct from DiSessa's four models, we identified a somewhat more complex emerging structure, that of a *development network*, where distributed groups with local expertise use a pattern language to share their expertise, sometimes in collaborative long-term projects, sometimes in ad-hoc exchanges. A detailed analysis of this model is presented in this chapter. What is clear at this stage is that a successful model needs to empower all partners in the design process, avoiding 'producer-consumer' and 'sage-

laymen' relationships.

Biographies

Niall Winters

Niall Winters (PhD in Computer Science, University of Dublin, Trinity College) is a RCUK Academic Research Fellow at the London Knowledge Lab, where he researches the ways in which *interdisciplinary* design and deployment can be supported in technology enhanced learning. Currently, his is involved in four research projects funded by the EPSRC/ESRC and the European Union. Niall has held visiting research positions with the Everyday Learning Group at Media Lab Europe in Dublin, and the Computer Vision Lab at Instituto Superior Tecnico in Lisbon.

Yishay Mor

Yishay Mor is currently a PhD student with the School of Mathematics, Science and Technology at the Institue of Education. Yishay holds an MSc in computer science from the Hebrew University, Jerusalem. He has previously worked as a researcher with the project Learning Patterns for the design and deployment of mathematical games and with the WebLabs project. Before that he designed and developed web-based network management software for Cisco Systems.

Dave Pratt

Dave Pratt (PhD in Mathematics Education, University of London, Institute of Education) is Professor of Mathematics Education at the Institute of Education, where he researches the relationship between mathematical thinking and the design of digital tools, often building learning environments as part of that research. Recently he has led Kaleidoscope projects on the design and deployment of mathematical games and on the construction of on-line materials for the study of design research, a methodology commonly adopted in his own research and in that of his doctoral students. He has published in many international journals and was, before that, a teacher of secondary level mathematics for 15 years.

1. Introduction

One of the reasons the development of technology enhanced learning (TEL) design patterns is a complex process is because it is dependent upon expertise from a number of disciplines. Each discipline brings to the field their own practices and experience; software developers rely on well-trialled engineering principles when building software; teachers are well versed in course and activity design and so on. If the challenges raised by TEL are to be adequately addressed, interdisciplinarity, defined as follows (Committee on Science, Engineering and Public Policy, 2004) needs to play a central role:

> Interdisciplinary research is a mode of research by teams or individuals that integrates information, data, techniques, tools, perspectives, concepts, and/or theories from two or more disciplines or bodies of specialized knowledge to advance fundamental understanding or to solve problems whose solutions are beyond the scope of a single discipline or field of research practice.

The aspect of interdisciplinarity that we focus on is the sharing of *design knowledge* across domains (including, but not limited to computer science, educational technology, teaching and pedagogical design). This is of fundamental importance as without it TEL artefacts risk being biased towards one of the dimensions of development (e.g. technically rich but pedagogical poor). Design patterns can directly address the challenge of interdisciplinarity if they are developed in a manner whereby they encapsulate the various types of design knowledge. The first question is how to support this process.
In this chapter, we take a two-pronged approach to this problem, describing a *model of development* – a characterisation of how development can occur and its associated *social configuration* – a description of the ways in which participants collaborate. More specifically, we are required to define the model of pattern development and the structure of the pattern development team. We established an *interdisciplinary* model of development interrelated with a *distributed development network (d^2n)* social configuration. The key point of this social configuration is that distributed groups with local expertise use a pattern language to share their expertise, sometimes in collaborative long-term projects, sometimes in ad-hoc exchanges.

The d^2n configuration was chosen so as to directly link the ways in which patterns are developed to interdisciplinary practice. The aim was to have participants work together on pattern development in an interleaved, iterative and integrated manner, while avoiding the potential pitfalls of multidisciplinary approaches. (In multidisciplinary approaches, each participant maintains their own disciplinary approach, effectively creating silos within the team that can lead to little or no integration). This was a challenge for two reasons. First, the process of developing TEL design patterns is complicated by the fact that within the field, both knowledge of design patterns and the process by which they are developed are less established than in other fields, such as software engineering. Thus, the potential of design patterns needs to be made aware to all participants. Second, interdisciplinarity, while often promoted as a laudable aim, is difficult to support in a pragmatic manner. One of the reasons for this is because barriers exist to developing a common understanding of a topic, making the establishment of common ground difficult (Caruso and Rhoten, 2001). Furthermore, working in an interdisciplinary manner implies

that opportunities for participants to collaborate on the development of patterns exist. However, managing collaborative practice to afford such opportunities can be complex for a number of reasons (diSessa, Azevedoa & Parnafes, 2004):

- *Divergent views*: TEL relies on input from many communities. Each has their own perspective on how to approach and solve problems. Therefore there is a danger that the design patterns developed may be biased towards one particular domain. Participants may also have different priorities in relation to the role of technology in education which need to be negotiated. In particular, in relation to design, participants may view the role of teachers and learners within the team very differently.
- *Social hierarchy*: diSessa, Azevedoa & Parnafes (ibid) point out that "[t]echnologists tend to have high status or, in a self-fulfilling manner, assume they have high status compared to educators, especially teachers". Related to the above point, the teachers' role can be supported by educational researchers, particularly in supporting them in abstracting their practice into design patterns (see Section 5.1)
- *Community-specific practices*: In any collaboration there is always the danger that participants are strongly affiliated with their own domain. Part of the practice of working in an interdisciplinary manner is overcoming this hurdle. The process of developing design patterns is one way of scaffolding practice, as the patterns can become a means through which collaborative discussion occurs. Additional support may be found at an institutional level, if interdisciplinary practice is valued.

In the Learning Patterns project, we have attempted to deal with these issues using the d^2n. Configured in this way, over the course of a year, the team produced in excess of 120 patterns (http://lp.noe-kaleidoscope.org/outcomes/patterns/). As the d^2n configuration is distributed in nature a means of supporting collaborative development at a distance was central to the process. Thus, we designed and built a web toolkit, which facilitates participants to undertake pattern development in a flexible manner. Furthermore, the toolkit is designed to support a "flowing engagement" between participants, mediating their practice but critically not getting in the way of it (Gross and Do, 2007).

1.1. The interdisciplinary model of development

As outlined in the introduction, a model of development is a characterisation of how development can occur. More specifically, it is a way of thinking about how to go about a particular development practice (diSessa, Azevedoa & Parnafes, 2004). Examples might include how to develop code or how to develop TEL resources. The important thing is that any model is not an idealised mode of practice. Instead it should be viewed as a guide to supporting implementation of a social configuration. In our case, when focusing on developing design patterns, we want an interdisciplinary model of development and therefore need to detail the characteristics that such a model will have:

- Design knowledge is captured in the form of design patterns: this emphasises that design patterns are the *construct* around which interdisciplinary practice is facilitated (see Section 3)
- The viewpoints of all participants have equal validity: Alexander (1979) promoted that idea that pattern languages have the explicit aim of externalizing knowledge to allow accumulation and generalization of solutions and to allow *all* members of a community or design group to participate in discussion relating to the design.
- Design patterns are co-constructed by participants: While patterns can be created by anyone, their *evolution* should be a community process
- Participants make every effort to work across domains: participants contextualise their perspective motivating their work for others to understand
- Patterns can emerge from the intervention of existing artefacts in particular settings: when developing patterns, it is not necessary to build new tools but it can also be a welcome source.

2. Related work: social configurations for development

Design pattern development in TEL is interdisciplinary. The elicitation and iteration of design patterns are well known to be dependent upon team development and critique (Retalis, Georgiakakis and Dimitriadis, 2006; Baggetun, Rusman and Poggi, 2004). As such, understanding the social configuration of teams and how this supports collaborative practice (in our case the sharing of design knowledge) is of critical importance. In attempting to undertake interdisciplinary development, the social configuration of the development team plays a significant role. The social configuration is defined as the ways in which participants collaborate and how this collaboration is structured. Before going on to discuss the d^2n configuration in more detail, we first present related configurations from software engineering and TEL.

Within the field of software engineering there has been an increasing interest in social structure, in particular within the Open Source movement. The overarching aim is to improve the planning of code development and software releases from understanding the various social structures and configurations that occur in practice. Crowston and Howison (2005) elaborate an idealised model of open source development based on four case studies of open source projects in the domain of computer science (see Figure 1).

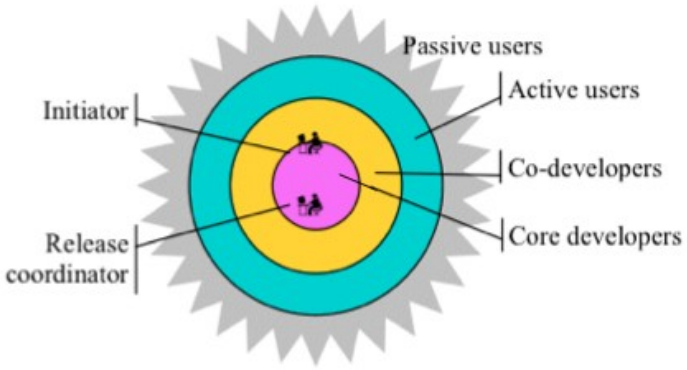


Figure 1 An idealised social structure of Open Source development (taken from

Crowston and Howison (2005))

The model consists of a small set of core developers responsible for major coding, supported by co-developers who primarily submit bug fixes. Active users provide use-cases and bug reports and test new releases. Finally, there are the passive users of the software who do not contribute directly to development. Scacchi's (2002) research supports this model. He notes that open source programmers take on particular roles in a development model where their contributions are shared on moderated community websites. Discussion and critique takes place within related forums or threaded emails.

In the software engineering community, in particular the object oriented programming community (see, for example http://www.industriallogic.com/training/dpw.html), pattern production is often facilitated at conference workshops. Indeed, we too have developed a workshop model (http://lp.noe-kaleidoscope.org/outcomes/workshops/). The focus of workshops is to aid participants to create patterns by themselves. However, this approach has limitations in that workshops are often (but not exclusively) focused on seed pattern development.

The social configuration of TEL teams can be considered to be less researched than their open source community equivalents. However, that is not to say that interesting research has not been done. A particularly illuminating study was undertaken by diSessa, Azevedoa & Parnafes (2004) who compared how research teams design, develop and evaluate TEL software, in the context of component-based educational programming. They identify the issue of the social configuration of the production team as "a critical family of issues that are easily marginalized" (p.117). They studied four such configurations in detail. We briefly summarise their structure as follows:

- *The Integrated Team Model*: Teams are structured into small, product-oriented groups
- *The Two-Legged Model*: Interaction is structured between two distinct teams (educationalists, technologists)
- *Member-Sustained Community Model*: "[An] Internet-based community of experts – teachers, researchers, developers, and others –that self-organizes to publish, share, find, critique, and improve software resources and associated materials'' (Roschelle, Pea, et al., 1999, p. 2, cited).
- *The LaDDER Model*: Four layers of participants. The focus is on empowering participants with less technological expertise, especially teachers and students, to solve as many of their own technology problems.

Each social configuration proposes a way to co-develop artefacts (in the case of diSessa, Azevedoa & Parnafes, software components). Critical to all four configurations is the relationship between educationalists and technologists. Each aims to structure this relationship – and the collaborative processes that go with it – in particular ways. As such, each can be viewed as operating on a 'spectrum of collaboration'. On the one hand, emphasis is placed on providing structures to support educators in working with technologists. On the other, there is a focused effort on maintaining an active and self-

organizing community of TEL participants.

We note here that whichever social configuration one chooses to investigate, complexities will arise. There is always a balance to be maintained between the pragmatics of technical development and the reality of classroom deployment on the ground. Indeed, diSessa, Azevedoa & Parnafes found that teachers (within the Integrated Team Model) found it "difficult and sometimes intimidating to participate as equal contributors in a technology-based development process". Furthermore, maintaining a workable coupling between domains of expertise is a significant challenge in order to avoid participants solely focusing on their own area of expertise.

3. A social configuration: the distributed development network (d^2n)

The distributed development network as a social configuration for developing design patterns evolved during the learning Patterns project. This was a 1-year project involving partner institutions across six European countries with expertise in computer science, educational technology, teaching, pedagogical design and games. The network further involved partner schools in three of the six countries, with 21 people making up the core of the team. The main aim of the project was to identify, elaborate and connect *design knowledge* from the various domains of expertise within and across the project and capture this knowledge in the form of an emerging set of design patterns.

By its very nature then, the development network is *distributed* in nature. Expertise is not co-located but rests with participants who are geographically dispersed. While some partners may have differing expertise available at their particular location, others may not. The distributed network structure of our development model thus requires *a hub* – a place where multiple perspectives on a problem could be shared and discussed. Furthermore, this hub must support the elicitation and construction of design patterns in a meaningful manner. Effectively this means that each of the main stages (Winters and Mor, 2008) of pattern development – identification, development, refinement – must be able to be undertaken in a distributed manner. In our case, building a hub necessitated the construction of a web toolkit (see Section 4) with distributed pattern development occurring primarily online, augmented by a small number of face-to-face meetings and workshops (see Section 4.3). Importantly, the development network and the web toolkit to support it are tightly coupled – one cannot exist without the other.

The next characteristic of the d^2n development model is that is has to *support pattern development* by those who are very familiar with the process but crucially must also provide "ways-in" for participants who are novices in the practice. Mechanisms need to be provided for participants to leverage their everyday experience and practices in order to support them in bootstrapping their development of patterns. We chose case studies as the mechanism. This choice was motivated by Yin (1994) who posits that case studies work well for describing interventions (in our case TEL artefacts) and the settings in which they occur. Moreover, the need to accommodate the concerns of diverse design

partners drives the author of a case study to identify the critical elements in their TEL design process, with respect to what design decisions worked and why, reflecting key choices that were made.

It is important that d²n supports a *cyclical link* between design and deployment. This needs to be maintained throughout the pattern development cycle. Participants with different expertise need to be able to "dip" in and out at any stage of the cycle. Therefore, the thinking underpinning any stage of pattern development, i.e. how participants conceptualise their own area of expertise, has to be available for open critique in a manner that is accessible to others. Such critique should be supported throughout the development process.

In designing the d²n social configuration, it is clear not only that interaction between participants is primarily mediated by data (i.e. hierarchical and cross-linked knowledge-domain typologies – visual mindmap overviews of subject areas, narrative form case studies and structured design patterns, what we term *design objects*) but these data are generated and contributed by participants themselves. The data form the basis for on-going process of pattern development. The critical point is that as d²n continues over time, the *patterns become the central construct and focus around which interaction between participants occurs*. The model's distributed nature foregrounds participants' *data contribution and analysis* over discussion via 'sage-layman' relationships. This is a very important structural component in supporting interdisciplinary practice. We want to avoid a dipolar structure between educational and technology strands (broadly defined) and instead facilitate a practice of *informed mutual development*. The ideal is that the two strands become almost indistinguishable.

4. The d²n web toolkit

Maintaining an effective distributed development network requires a set of supporting tools which are functionality rich while being easy to use. The d²n web toolkit was designed and developed with this aim in mind. This toolkit supports identifying, developing, mapping, sharing, discussing and classifying design objects, i.e. elements of design knowledge. The design of the system was iterative, and in a sense auto-reflexive, as it embodies many of the patterns it hosts. Indeed, most of the issues noted here are represented as patterns in our language. The full technical description of the system is beyond the scope of this chapter, and can be found in (Pratt et al, 2007). In this section we highlight some of the key issues that emerged from or experience in designing, developing and using the system.

4.1. Form follows practice: embedding the social configuration in interface design

To a large extent, the success of the toolkit was due to the measure by which it was attuned to the social practices of the community it served. Starting from a minimal set of features, enhancements were continuously added as participants' needs were identified.

These needs, and hence the emerging features of the system, reflect the social dynamics within the community. Examples include the manifestation of collaborative development, demarcation of authorship and contribution, and an open-process culture. Each of these aspects was supported by particular mechanisms of interaction. These are the "protocols, formal structures, plans, procedures and schemes [that] reduce the complexity of articulating cooperative work" (Schmidt and Bannon, 1992 cited in Grinter, 1995). Thus, the mechanisms are standard operating procedures that govern how a team interacts.

(i) Manifestation of collaborative development
Design objects are in "perpetual beta", constantly being refined and reconfigured by participants. Design objects are simultaneously a representation of existing knowledge and a means for constructing new knowledge. The particular balance between the two shifts over time. A newly minted design pattern is often little more than a flagged issue for investigation. This investigation proceeds through analysis of case studies and interdisciplinary debate. Eventually, the pattern matures to an encapsulated unit of knowledge, which can be used as a building block in larger structures. This trajectory of design object refinement, and the social process which drive it, need to be represented in the user interface. In our case, these were captured by elements such as the pattern status and ranking, and the design-object discussion forums.

Each pattern is assigned a state depending on its level of completeness: *seed*, *alpha*, *beta* and *release*. Seed patterns often represent ideas, which were noted during discussion or while developing other patterns. They are essentially placeholders, which would probably not make much sense to anyone other than their authors. Once they undergo the initial editing cycle, they are promoted to alpha state. This state signifies patterns which require refinement before they are submitted to public review – the beta state. The feedback from this review will be used to bring the pattern to its final release state. We note here that the process by which a pattern moves through these states is non-trivial – see (Winters and Mor, 2008) for a methodology of how to do so. The second indicator, 'rank' provides the authors with a meta-review of how significant the community of patterns authors view this pattern. This can be a guide as to where the user may wish to allocate their time on pattern development. State and rank are displayed clearly on the header of each pattern page (see Figure 2 and Figure 3) and in the table view of all patterns (see Figure 4).

Content morph		Category:	Bootstrap
Created	Michele Cerulli, 26 May, 2006	Modified:	20 November, 2006
Status	release	Rank	★★★★ 4
Summary	Keeping the structure of the game and the educational principles, but changing the addressed mathematical content.		

Figure 2: header of a pattern in release state, with a ranking of 4

A Key Question		Category:	Man man
Created	Karl Alfredsson, 26 June, 2006	Modified:	16 November, 2006
Status	seed	Rank	2
Summary	Provide a context and inspiration for posing a big question to which pupils can relate. The key question may often be a dilemma to be discussed among the pupils, creating a room for learning activities.		

Figure 3: header of a pattern in seed state, with a ranking of 2

Name	Submitted By	First listed	Last Edited	Summary	category	status	rank
A Key Question	Karl Alfredsson	26 June, 2006	16 November, 2006	Provide a context and inspiration for posing a big question to which pupils can relate.	Man man	seed	
Abilities based task assignment	Matthew T. Atkinson	26 June, 2006	27 July, 2006	In a context with people with different abilities how to divide labour in order to keep the activity challenging for everyone	Man man	alpha	
Active worksheet	Yishay Mor	07 June, 2006	21 August, 2007	Scaffold learners work by an on-line worksheet which they edit as they go along the task.	Man idea	beta	

Figure 4: status and rank columns on the left of the patterns table

The collaborative dynamics of discussion, disambiguation and refinement are captured by the design object discussion forums and versioning. A discussion forum and list of historical versions is attached to each design object – typology, case study, pattern (Figure 5) or the structure of the language as a whole (Figure 6). With the discussion forum for each pattern the critical issues relating to how the pattern developed will have been detailed, as it is often the case that pattern development will have been spurred on by discussion with other pattern authors.

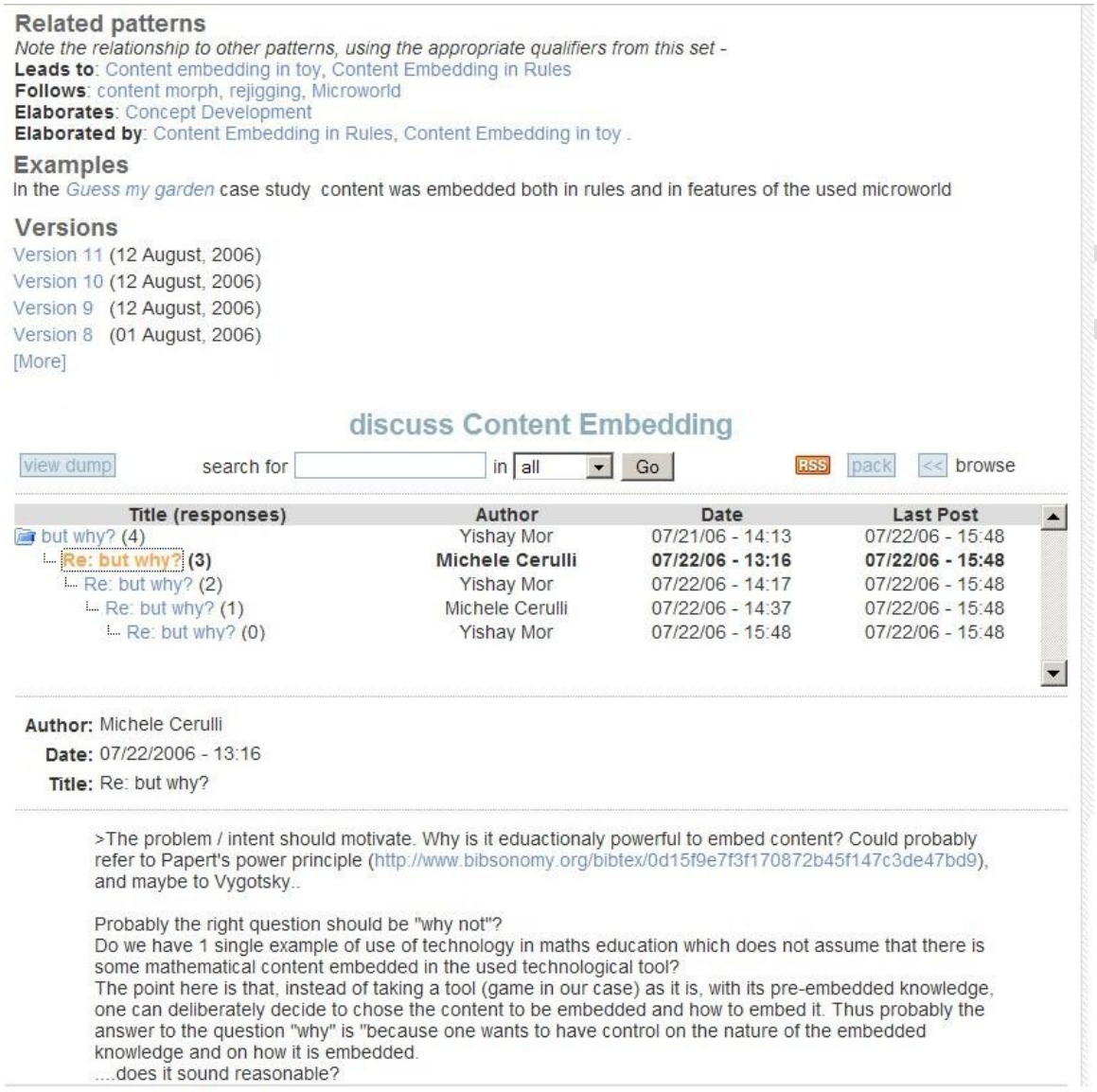


Figure 5: fragment of the Content Embedding pattern, with versions and discussion forum

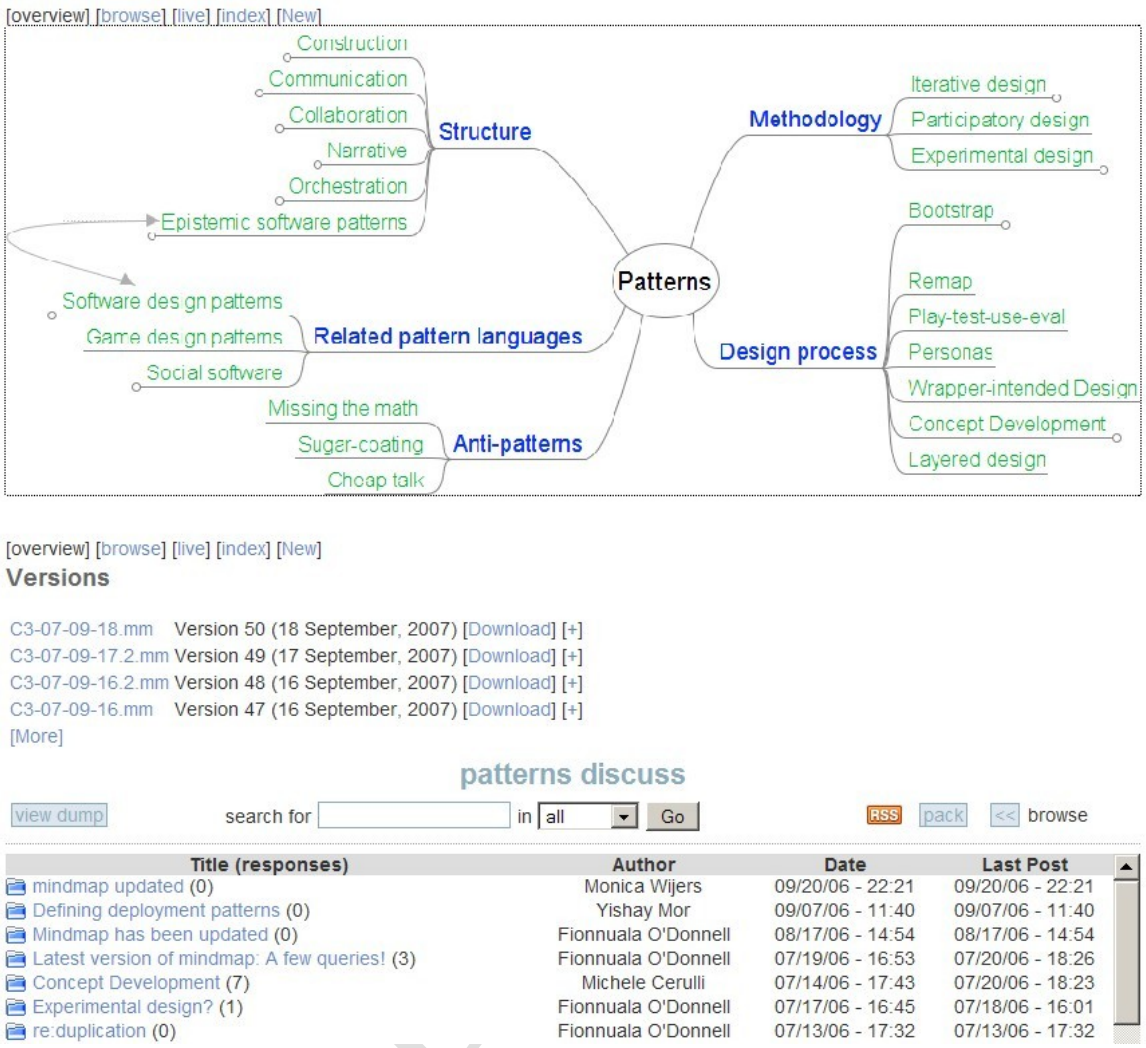


Figure 6: Overview view of the pattern language, with versions and discussion forum

Discussion forums proved highly valuable in cases requiring intensive coordination across disciplines, thus involving multiple authors and multiple objects. Notably, in the development of the typologies, we needed to remove redundancies and identify intersection points. This need was addressed by leaving comments on each other's forums.

They were used significantly less in the case of self-contained objects, such as case studies and patterns. Partially, this is due to the lack of a robust notification mechanism: since the typologies discussion was localized in time and web-space, participants could keep track of the forums activity. By contrast, sporadic comments on a large number of objects are hard to follow. Both the success of forums, in the case of typologies, and their relative failure, in the case of patterns, support the argument that interface design should follow social configuration. Had we found the time to apply this principle in the later case, we would have provided a means for notifying authors of all activity on all objects

they are involved in (e.g. by email or RSS).

(ii) Demarcation of authorship and contribution
Collaborative authoring systems often either highlight individual authorship (e.g. blogs) or blur it altogether (e.g wikis). Yet most communities engaged in the collaborative construction of digital artefacts employ a much finer social structure of authorship and contribution. Typically, each object will have one primary author, several secondary authors, and many ad-hoc contributors. Such is the case in most open-source projects Crowston and Howison (2005) and in pattern language communities (Schuler, 2002, for example).

In our community, each typology had one editor-in charge, each case study was offered by one or two participants, and each case study had one main author – although, in the case of workshops, this author represented a group. Occasionally, the lead on a particular design object would shift from one author to another. In all cases, there was a wider group of contributors who would review and critique the design object under development.

In order to streamline the collaborative process, these structures of authorship and responsibility need to be made salient. This was achieved in the toolkit by distinguishing discussions from edits, and displaying author names along each version and forum comment, as well as on the index view of pattern and case studies.

(iii) Open-process culture
One of the early decisions of the project team was to make not only the products of our work free and open, but also to expose the process itself. This decision implies that all versions and all discussions of all our design objects are accessible to the public through the project website. The rationale behind this policy is twofold. Obviously, it creates a possibility for unexpected feedback and contributions from experts from outside the group. It also enriches out offering: by exposing the social configurations and dynamics from which our language emerged, we enable others to evaluate and hopefully adopt these to serve similar endeavours.

Openly sharing our work process raises a risk of overloading newcomers with excessive, immature knowledge. For example, roughly a third of the patterns in the database are still in seed state. Such patterns will make little sense to a casual viewer. If such a user would browse through the collection unguided, she might be overwhelmed and confused by these.

To address this issue, the site provides two views on our work (Figure 7): the *outcomes* view aims to present the fruits of our work in the most accessible form, while the *workspace* view presents them in full detail, including historical versions and discussions.

Figure 7: outcomes vs. workspace views

The trails view, described in Section 4.2, is another mechanism for tackling the tension between veterans and newcomers views of the language.

4.2.Multiple contexts, multiple representations

The d^2n social configuration is diverse not only in the domains of expertise it encompasses, but also in the intensity and character of the activities participants undertake. These dimensions define a space of contexts of use. While the underlying set of design objects may remain the same, the users' perception of them and their desirable varies dramatically. To support the various contexts effectively, the web toolkit offers targeted representations of design objects. The problem of navigating the pattern language is a primary example of this issue. Possible contexts for this activity include: a public review (e.g. at a conference), groundwork for a new design initiative, resource management, structural editing, and newcomer's enculturation.

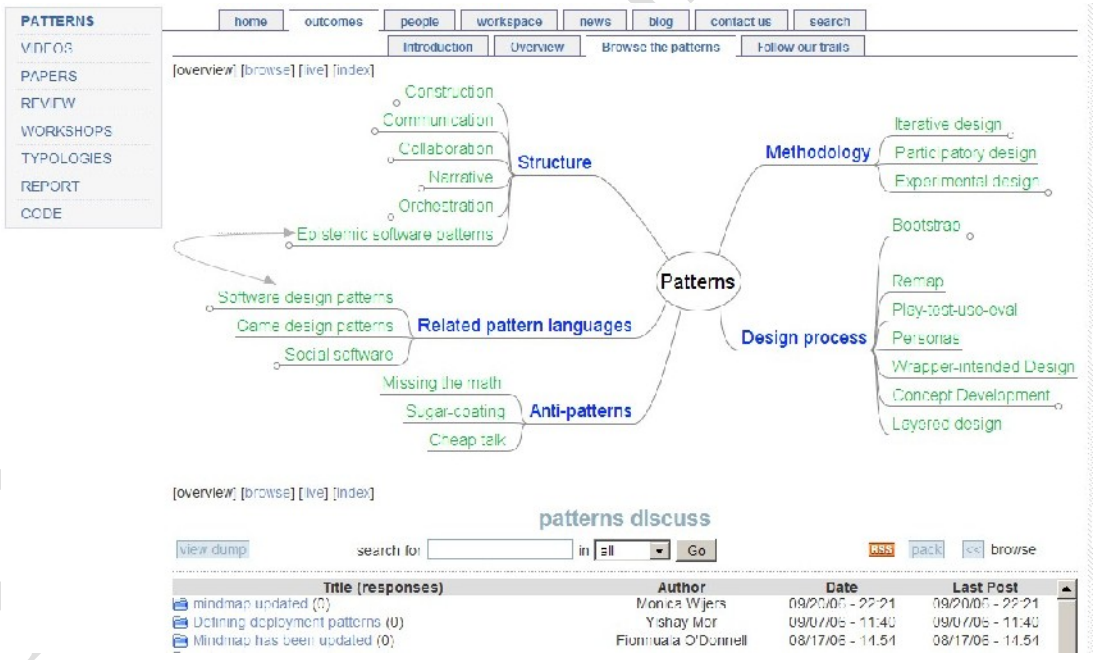


Figure 8: overview visualization of the pattern language

When presenting the language for public review, it is critical to offer reviewers a lucid and immediately accessible visual representation of the language. This representation would serve as a backdrop for a frontal verbal presentation, leading to a more in-depth discussion. The overview perspective affords such a representation (Figure 8). This

perspective does not support navigation, but provides a good initial impression of the language.

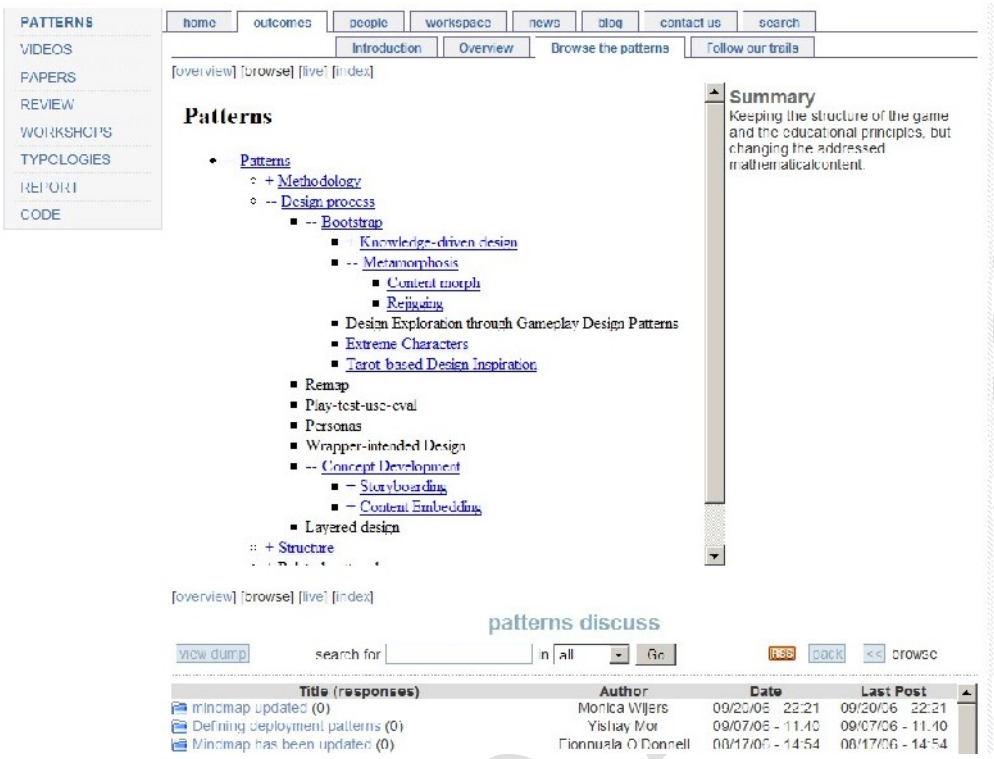


Figure 9: Browsable tree view of the pattern language

The overview perspective quickly becomes ineffective when shifting to a more detailed review, either in the course of a review discussion or when using the language as a resource in design practice. The browse perspective (Figure 9) was designed to support such contexts. This perspective allows the viewer to traverse the hierarchy of patterns with a quick view of each pattern's summary, homing in on patterns of interest. While effective for its intended context of use, this representation lacks a lot of meta-data which is essential in other contexts.

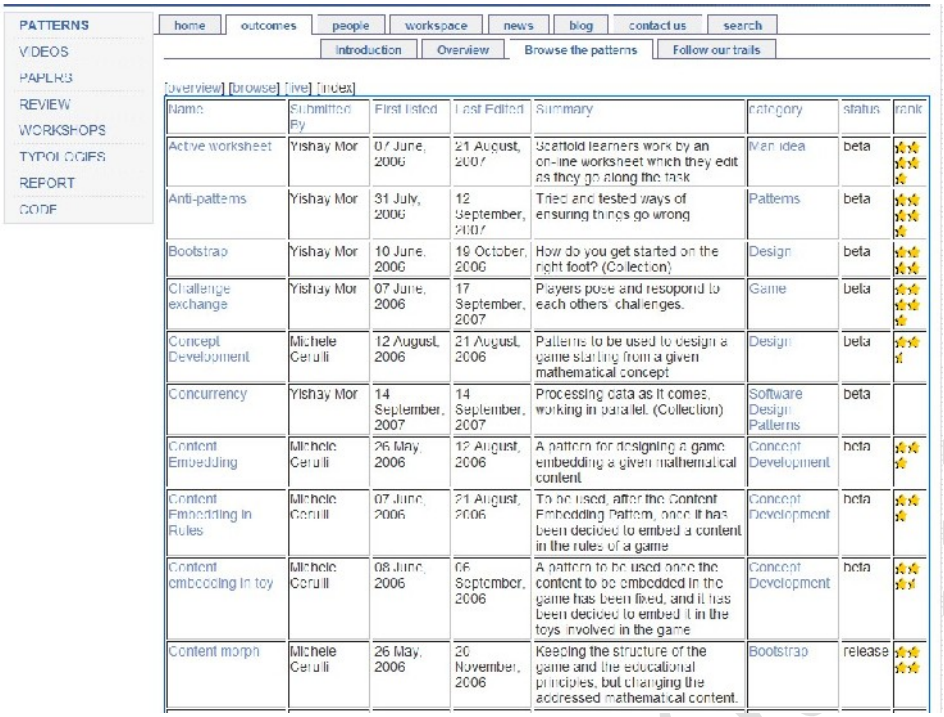
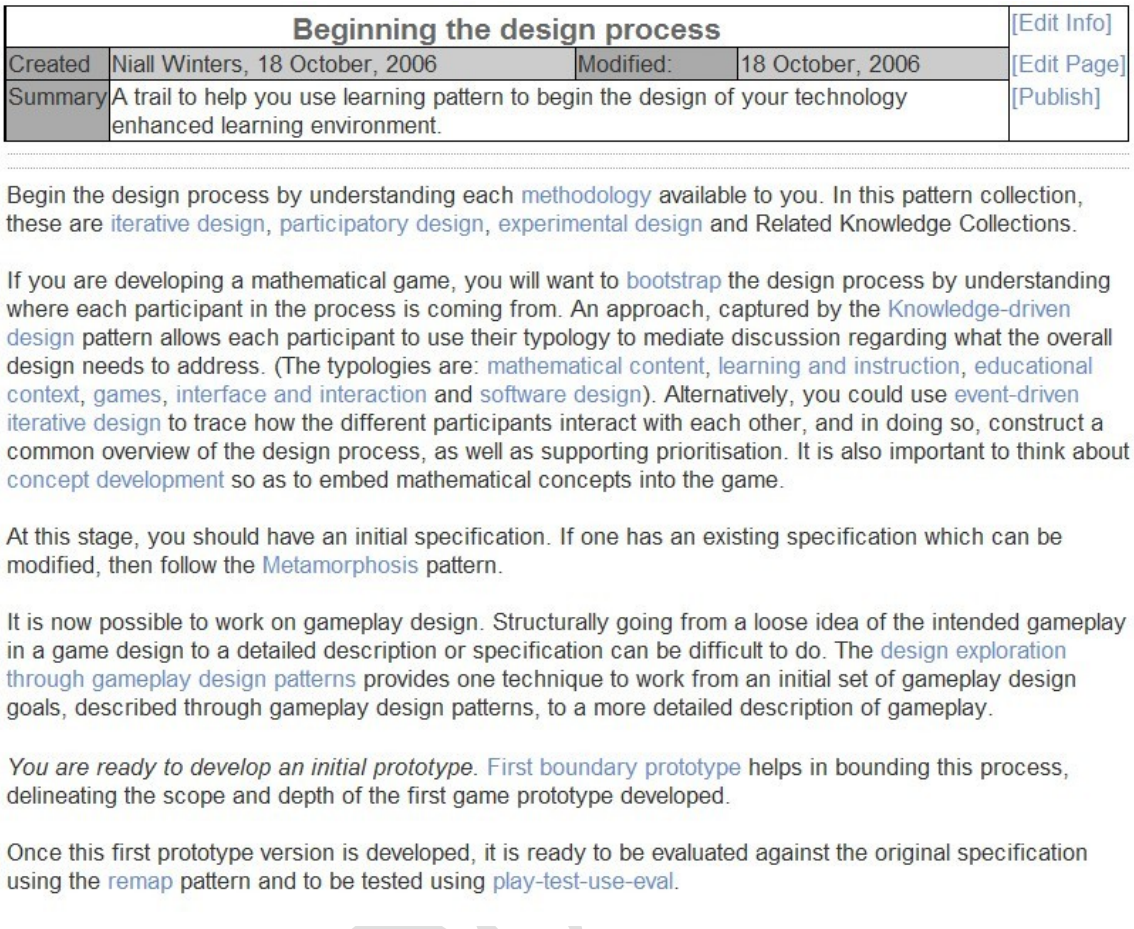


Figure 10: index view of the pattern language

Core members of the network will need to perform resource management and systematic editing tasks. They might need to work through all patterns in a particular state, promoting them to the next. They might want to review all their contributions and eliminate redundancies, or focus on those patterns that their peers found most useful. To support such a context of activity, the index perspective (Figure 10) offers a sortable tabular view of the language. This view flattens the structure, assuming a familiarity with the language as a whole. While being highly effective for experiences contributors in focused tasks, it is nearly impenetrable for novice viewers.



Beginning the design process				[Edit Info]
Created	Niall Winters, 18 October, 2006	Modified:	18 October, 2006	[Edit Page]
Summary	A trail to help you use learning pattern to begin the design of your technology enhanced learning environment.			[Publish]

Begin the design process by understanding each methodology available to you. In this pattern collection, these are iterative design, participatory design, experimental design and Related Knowledge Collections.

If you are developing a mathematical game, you will want to bootstrap the design process by understanding where each participant in the process is coming from. An approach, captured by the Knowledge-driven design pattern allows each participant to use their typology to mediate discussion regarding what the overall design needs to address. (The typologies are: mathematical content, learning and instruction, educational context, games, interface and interaction and software design). Alternatively, you could use event-driven iterative design to trace how the different participants interact with each other, and in doing so, construct a common overview of the design process, as well as supporting prioritisation. It is also important to think about concept development so as to embed mathematical concepts into the game.

At this stage, you should have an initial specification. If one has an existing specification which can be modified, then follow the Metamorphosis pattern.

It is now possible to work on gameplay design. Structurally going from a loose idea of the intended gameplay in a game design to a detailed description or specification can be difficult to do. The design exploration through gameplay design patterns provides one technique to work from an initial set of gameplay design goals, described through gameplay design patterns, to a more detailed description of gameplay.

You are ready to develop an initial prototype. First boundary prototype helps in bounding this process, delineating the scope and depth of the first game prototype developed.

Once this first prototype version is developed, it is ready to be evaluated against the original specification using the remap pattern and to be tested using play-test-use-eval.

Figure 11: A trail leading newcomers through a usage sceario

One of the hardest challenges for pattern languages is the entry problem (Winters and Mor, 2008). Our approach to this issue employs a trails perspective. A trail is an informal illustrative account of how patterns were derived or how they might be used. The purpose is to provide a starting point for detailing a particular practice that the pattern language covers (for example "beginning the design process", Figure 11) in narrative form, providing links to each of the patterns used. The aim is not to present the narrative as hard data or detailed analysis, but rather as an aid for the reader to gauge the nature of the patterns approach. It offers an initial opportunity for readers to begin to understand the deep, complex and structured relationships between patterns, while knowing that these relationships can, and have been successfully explored and mapped in an interdisciplinary manner. Furthermore, trails allow for exploration at both the abstract and specific levels by constructing the narrative to 'drill-down' through the levels of the language hierarchy.

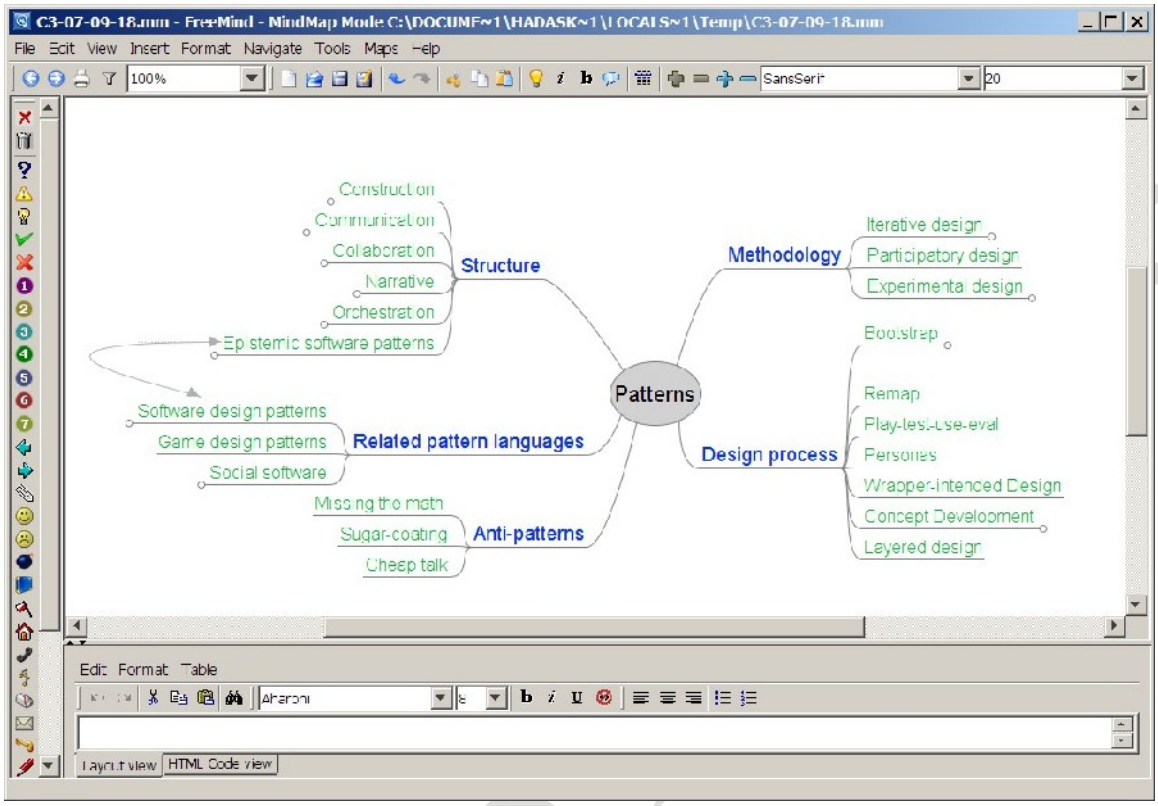


Figure 12: Editing the language structure in FreeMind

Finally, there is the occasional task of restructuring the hierarchy of the pattern language, incorporating new patterns and reconfiguring categories. This complex task requires a malleable comprehensive representation of the language structure. The nature of this task suggests that it would be done by one or two members, who are intimately familiar with the language, sitting at a single computer, at a single session. Since structural changes can be far-reaching, they need to be done en-bulk. With this context in mind, the logical mode of work is to download a map of the language – pattern names and links – and manipulate it using a graphical desktop editor. This was achieved by using the open source FreeMind program. The language structure is discussed by the core team using a forum, as mentioned in Section 4.1. Following each round of discussions, one or two team members will edit the map in FreeMind, and upload a new version – to feed into the next round of discussions.

4.3.Blended usage

When considering a web-based environment for collaborative development of design objects, and its context of use, one would expect these to be limited to distributed scenarios. Initially, this was the assumption underlying the design of the d^2n toolkit. With time, we realized that it provided invaluable tools for a much broader set of situations. On one hand, the toolkit proved to be an effective individual design research workspace. On the other hand, it emerged as a powerful resource in pattern workshops (Mor and

Winters, 2008b). In the former case, the authors have used the toolkit as an aid for detailed analysis of their work in previous projects. The results of that analysis are currently being prepared for publication.

The later case is perhaps the more surprising. The project conducted a series of pattern workshops, in which practitioners and researchers from diverse fields met to share their knowledge and discuss questions which emerge from their experience. These workshops used participant-contributed case studies as a central resource. Working in groups, these case studies were mapped to the typologies and compared to peer's experiences. Eventually design patterns were distilled from the case studies.

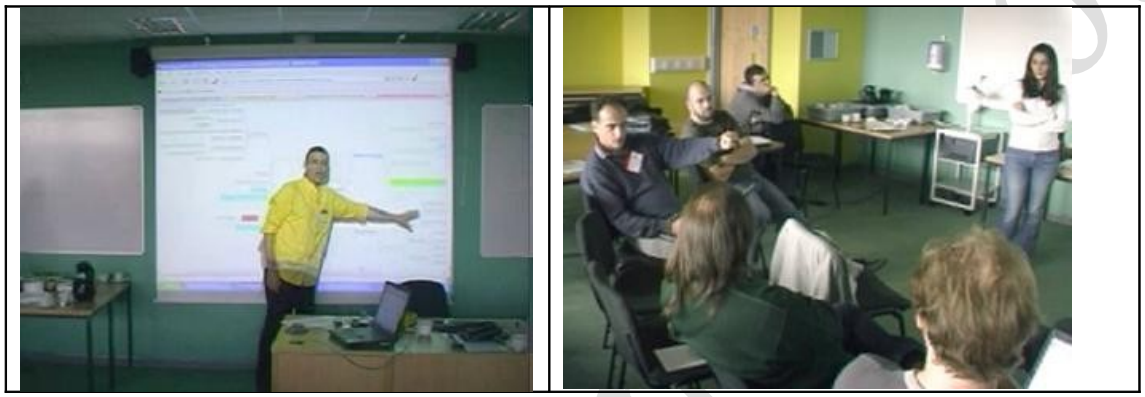


Figure 13: using the toolkit in pattern workshops

The toolkit was an enabling asset from the moment participants registered for the workshop until they returned to their homes. Using it, participants contributed case studies in advance. These contributions were used to anchor the group discussions. The typologies guided the discussion and allowed participants to quickly orient themselves with the work of peers from remote disciplines. Patterns were recorded on the site as they emerged in conversation and then revisited and refined. Finally, each group presented its findings to the assembly by displaying the new design objects they have created on the whiteboard.

In retrospect, perhaps the utility of the toolkit in blended contexts is not so surprising: if a tool is good enough to support collaboration in distributed communities, it should first be productive in less demanding contexts, which do not require intensive collaboration, or where communication is unmediated.

5. Discussion

The key driving consideration behind the distributed development network (d^2n) was to promote a social configuration that supports the practice of collaborative pattern development. In this section, we delineate the complexity of this process along three lines:

- Facilitating abstraction by participants
- Web toollkit usage for supporting pattern development

- Designing for collaboration

5.1.Facilitating abstraction by participants

One of the key problems to be faced when developing design patterns is supporting participants to think in an abstract manner. As a prerequisite to writing patterns, participants must develop the skills to generalise from the specific contexts of their everyday practice – to see the general in the particular (Mason & Johnston-Wilder, 2004). There were a number of problems to deal with: (i) initiating the process of abstraction, (ii) understanding the relationships between patterns as the language grew and (iii) understanding where in the hierarchy a pattern fitted.

(i) Initiating the process of abstraction

This is a complex and difficult topic that sought to address as part of our IDR methodology for pattern development, detailed in (Winters and Mor, 2008). Here we only provide a brief outline. A first step in recognising aspects of experience that may have more general significance is to explore ways for participants to conceptualise their own area of expertise in a way that is accessible to others. This needs to be described in a manner which pertains to the problem domain at hand. To address this aim we used the coupling of typologies with case studies. By mapping the practices and content detailed in the case studies to the set typologies, the aspects of practice are immediately categorised for discussion. This provides a starting point for participants to see the general across instances captured by specific cases. Moreover, the process of doing abstract is a process of learning. In a sense, the participants learned how to develop patterns as a process of doing abstraction. Béguin (2003) support this perspective when referring to the close relationship between design and learning. He suggests that effective design should be constructed as a process of mutual learning involving users and designers and argues that the products only reach their final form through use. This should be reflected in an iterative design process, which allows the users and designers to collaboratively shape their concept of the product and its actual form simultaneously. Such an approach, if sometimes not explicitly stated in these terms, motivates the social dynamics of collaborative abstraction, elaboration and refinement of design objects. As noted in Section 4.1 the web toolkit follows these dynamics closely.

(ii) Understanding the relationships between patterns through visualisation

In our case, this proved particularly difficult for those from a non-computer science background. The difficulty lies in the apparently semantic-free nature of a high level design pattern for participants, typically teachers, whose normal practice is rooted in concrete action – for example, planning to teach specific children with very particular aims and objectives. Only by drilling down to lower level patterns could such participants find the level semantics that could be related to their normal practice (see Tripp (1985) on educational generalisation and (Winters and Mor, 2008)). Knowing which high level pattern might contain within it familiar practice involves seeing the particular in the general (Mason & Johnston-Wilder, 2004), the inverse challenge to that discussed in (i) above. As noted in (Winters and Mor, 2008), this was a problem because the inheritance relationships between patterns (i.e. *Elaborates*, *Elaborated by*, *Follows* and *Leads to*) proved complex to understand. This was compounded by the fact that, aside from a small

number of face-to-face project meetings, the relationships had to be conveyed within the web toolkit. The various viewpoints were a partial solution to this problem: the principle was to *foreground the visual*. In particular the live-view – in essence a clickable hierarchical map of direct (*Follows* and *Leads to*) relationships between all of the design patterns – proved helpful. This was because the level of complexity in understanding all of the relationships at any one time was removed. By simply viewing the mindmap, a participant could see where any pattern fitted in the "bigger picture". Furthermore, at the highest level, patterns were clustered into categories, thus dividing the language into more manageable sub-components. In the live-view the five categories were the entry points into the language. When any pattern was clicked, its direct relations are shown.

(iii) Understanding where in the hierarchy a pattern fits
Once the pattern language begins to be populated the problem of where to place new patterns and how they are to be related to the patterns that are already part of the language structure arises. One way for participants to gain an insight into this problem was to cluster related patterns into mini-language or trails. Structuring a language by a distributed multidisciplinary network is a hard task, and involved work in several contexts and lengthy discussions. Typically, the pattern contributor – or a more experienced editor – would search the index view for related patterns and study them, then download the map and rearrange it to accommodate the new pattern. Once uploaded, team members would observe the changes in overview and browse perspectives, and discuss them in the forum. This might lead to further off-line edits, and occasionally to the addition or merging of patterns.

While the issue of abstraction is common to any social configuration for developing design patterns, in d^2n it was a particularly acute problem, given the primarily distributed nature of development. Seeing the general in the particular and the particular in the general requires a structuring of attention more easily arranged through face-to-face encounters. The real-time interactive nature of face-to-face discussion together with cues implicit in gesture and tone seemed critical in teasing out the patterns in experience and the meaningfulness of patterns during the workshops.

5.2.Web toolkit usage for supporting pattern development

Interaction between participants within the d^2n social configuration was heavily reliant upon the web toolkit. As is common in the open source community, there were core pattern developers, co-developers and active developers. In the main, the core pattern developers took ownership of the seed patterns they submitted. It was often the case that pattern development proceeded via the input of 2-3 other co-developers. In particular, these interventions were to detail the pattern from another disciplinary perspective – adding information that they felt was missing. This was primarily evidenced by the versioning of patterns, where the evolution can be seen.

A common way of iterating a pattern was by filling in the 'problem' section with a few sentences, often linked to a particular example or case study. Next, a bulleted pointed list of the pattern steps would be filled in. At a later stage the context section would be added

to. Here is where co-developers primarily played a part – illuminating the context via the typology structure. If the pattern related other examples, they were added to further illustrate the context (Winters and Mor, 2008).

It was usual for core developers to become much more experienced at developing the patterns than active developers. In cases where active developers submitted patterns in an incorrect format, or where they were missing crucial details, the core developers would sometimes contact them for clarifications regarding their intention for the pattern. This again goes back to the problem of having to think in an abstract manner when developing patterns.

The pattern relationships were either added at this stage or at the very beginning of the process. Indeed, as the language emerged over time, this proved to be somewhat problematic as dealing with multiple inheritances was encumbered by the toolkit design. In any future version, this would be taken care of automatically, as inconsistencies did arise.

5.3.Designing for collaboration

How any tool facilitates collaboration can be analysed using the design principles promoted by Gross and Do (2007). The functionalities of the web toolkit did support people in performing their intended actions, i.e. developing patterns. This is evidenced by the fact that as a distributed team we developed over 120 patterns, many of which where seeded at workshops but developed via the toolkit. The key design consideration was to promote the patterns as the central construct around which collaboration occurred. The interplay between developers was focused on making patterns 'better'. This means that as the pattern language developed, the number of resources available on the web toolkit increased, thus promoting further collaborative shaping of the language. In the best case, this was an iterative cycle between design and deployment. The d^2n social configuration thus supported collaboration in a participatory manner, where the evolution of techno-pedagogic design patterns was the goal.

We found that the functionality of the web toolkit did affect the intended outcomes of working as a distributed development network. The key to the toolkits success was in the combination of high-level principles, such as those described above, with attention to minute detail, such as a wiki-style quick linking mechanism, and unobtrusive templates for design patterns. Yet, many desired features were never deployed, and their absence was a notable obstacle. For example, the mapping of patterns to typologies was tediously manual, links between objects were not updated automatically, and visualisation of the single pattern is still an open challenge. As noted in Section 4.1, due to the emergent functionalities of the web toolkit, the interaction design did improve over the course of the project, directly addressing participants' needs.

Given the complexities of developing patterns in a distributed manner, d^2n can be considered to have worked well, as evidenced by our outputs. We found definite advantages to capturing practice as it happened over time. In particular, 'process capture' proved useful when mapping the structure of the language. Sustainable collaborative

pattern development, without the supporting resources offered by the web toolkit, would have been problematic. Email, for example would not have provided the team with the necessary overview of all the participants work.

6. Conclusion

The design, implementation and evaluation of TEL artefacts (software, pedagogic plans, learning resources etc.) demand an interdisciplinary approach. The implication of this is that the development process is an inherently complex one, encapsulated by the overarching challenge of supporting the relationships between technologists and educationalists. This led us to study the ways in which collaboration between participants can be characterised by an interdisciplinary model of development, where the social configuration of the team is distributed in nature. The first rationale behind the distributed development network (d^2n) was to support patterns development over a long timeframe, in the order of months. As such, online facilitation was required, fitting in with participants' busy schedules. Furthermore, collaborative development occurring in this way affords expertise from a view community of participants to be leveraged. It also has the potential to scale, harnessing the collective intelligence of the contributors, which is a key underlying principle of Web 2.0 (O'Reilly, 2005).

As noted in the (note to editor: introduction to this book), design patterns are normative. Alexander always viewed them as having wide benefits. In developing TEL resources the same argument is also true: a TEL resource should leverage as much empirical evidence as possible so that combined with high-level technical skills, artefacts of worth can be produced. This provided a second rationale behind d^2n: to provide a means for all participants to share their design knowledge, with the intention of producing worthwhile artefacts. This normative focus on sharing design knowledge formed the bedrock of our collaborative efforts.

However, we must also be clear on the limitations of our approach. Supporting any development network takes time and effort. Each participants need to feel that their contribution is valued and that their perspective is not 'over-ridden' in the quest for interdisciplinarity. Here a potential advantage of the d^2n is evident: by working in a distributed manner, participants have the time to reflect on the pattern development process, providing a slow-burn evolution of the language. Rather than a concentrated focus (that work occur during workshops, for example), d^2n provides that time and space for negotiation around the ways in which each pattern can develop. This can go some way towards building an increased understanding of how participants (in particular teachers) engage in design work (Goodyear, 2005).

From the experience of the Learning Patterns project, the social configuration of any TEL team is critical to its success. In this chapter, we have explored how to support interdisciplinary in a distributed form. We have seen the potential of this configuration to support pattern development. However, it is only a starting point. There are many challenges remaining before patterns become a core resource tool for the TEL community. We see d^2n as a stepping-stone in this direction.

Acknowledgments
We thank X, Y and Z.

References

Alexander, C. (1979) *The Timeless Way of Building*. New York: Oxford University Press.

Baggetun, R., Rusman, E., & Poggi, C. (2004). Design patterns for collaborative learning: From practice to theory and back. In L. Cantoni & C. McLoughlin (Eds.), *Proceedings of the World Conference on Educational multimedia, hypermedia and telecommunications*, AACE, Lugano, Switzerland (pp. 2493 – 2498).

Béguin, P. (2003) Design as a mutual learning process between users and designers *Interacting with Computers*, 15(5), 709-730. http://dx.doi.org/doi:10.1016/S0953-5438(03)00060-2 last accessed 15 September 2006

Caruso D. and Rhoten D. (2001) *Lead, Follow, Get out of the Way: Sidestepping the Barriers to Effective Practice of Interdisciplinarity*. http://www.hybridvigor.net/interdis/pubs/hv_pub_interdis-2001.04.30.pdf, last visited 3 November 2007

Committee on Science, Engineering and Public Policy (2004) *Facilitating Interdisciplinary Research*. Washington D.C.: National Academic Press

Crowston K. & Howison, J. (2005) The social structure of free and open source software development, *First Monday*, 10(2) http://firstmonday.org/issues/issue10_2/crowston/index.html last accessed: 13 October 2007

diSessa, A. A.; Azevedo, F. S. & Parnafes, O. (2004), Issues in Component Computing: A Synthetic Review, *Interactive Learning Environments* 12(1), 109-159.

Froehlich, J. and Dourish, P. (2004). Unifying Artifacts and Activities in a Visual Tool for Distributed Software Development Teams, *Proceedings of International Conference on Software Engineering ICSE 2004*, Edinburgh, UK, 387-396.

Goodyear, P. (2005) Educational design and networked learning: Patterns, pattern languages and design practice, *Australasian Journal of Educational Technology*, 21(1), 82-101.

Grinter, R. (1995) Using a Configuration Management Tool to Coordinate Software Development, *Proceedings of the Conference on Organizational Computing Systems*, California, US, 168-177

Gross, M. & Do, E. (2007) Tools and Principles for Collaborative Design, *Workshop on Tools in Support of Creative Collaboration*, Washington D.C. June 13-15 http://orchid.cs.uiuc.edu/CreativeCollaboration last visited 5 November 2007

Mason, J.H., Johnston-Wilder, S.J. (2004) *Fundamental Constructs in Mathematics*

Education Mason, J. Johnston-Wilder, S. (eds.) Routledge Falmer

O'Reilly, T. (2005). *What is Web 2.0?* Last retrieved from: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html?page=1, 28 October 2007.

Retalis, S. Georgiakakis, P. & Dimitriadis, Y. (2006), Eliciting design patterns for e-learning systems, *Computer Science Education* 16(2), 105-118

Scacchi, W. (2002) Understanding the Requirements for Developing Open Source Software Systems, *IEE Software Proceedings* 149 (1), 24-39

Schuler, D. (2002) A Pattern Language for Living Communication, *Proceedings of the Conference on Participatory Design*, Malmö, Sweden, June 23-25, 434-436, CPSR, Palo Alto, CA.

Tripp, D. (1985) Case Study Generalisation: An Agenda for Action, *British Educational Research Journal*, 11(1), 33-43

Yin, R. (1994) *Case study research: Design and methods* (2nd ed.). Thousand Oaks, CA: Sage Publishing.