*Journal of Universal Computer Science, vol. 18, no. 16 (2012), 2252-2273*
*submitted: 10/1/12, accepted: 11/6/12, appeared: 28/8/12 © J.UCS*

# The Educational Affordances of Widgets and Application Stores

**David Griffiths**
(The Institute for Educational Cybernetics, The University of Bolton, Bolton, UK
D.E.Griffiths@bolton.ac.uk)

**Mark William Johnson**
(The Institute for Educational Cybernetics, The University of Bolton, Bolton, UK
M.W.Johnson@bolton.ac.uk)

**Kris Popat**
(The Institute for Educational Cybernetics, The University of Bolton, Bolton, UK
K.J.Popat@bolton.ac.uk)

**Paul Sharples**
(The Institute for Educational Cybernetics, The University of Bolton, Bolton, UK
P.Sharples@bolton.ac.uk)

**Scott Wilson**
(The Institute for Educational Cybernetics, The University of Bolton, Bolton, UK
S.Wilson@bolton.ac.uk)

**Abstract:** In order to provide interoperable services to a range of applications, platforms and devices a number of open source applications have been developed, many of them within the Apache Software Foundation. We analyse the way that these relate to research and development in education, which has also informed the functionality which they offer, providing a case study of the relationship between generic open source infrastructure development, and the discourse around pedagogy. The functionality foreseen for Personal Learning Environments and for the learning design approach to face-to-face learning is identified. The capabilities of Apache Wookie (incubating) W3C Widget Server are compared with this desired functionality, and the unfulfilled functionality identified with a particular focus on the need to support teachers control over their technological environment in response to emerging conditions in the classroom. The application store ('app store') is identified as a key software paradigm for meeting the unfulfilled functionality, and the ways in which it can support teaching practice are explored. A number of current software projects, and collaborations between them, are described which are contributing to providing a coherent infrastructure for building app stores. Finally some areas of functionality which remain pending future research and development are identified.

**Keywords:** W3C, Widget, affordance, Personal Learning Environment, learning design, app store, Apache, Wookie, Rave, EDUKapp, iTEC, Omelette, iCamp, ROLE
**Categories:** L.2.1, L.2.2, L.3.0, L.3.6

## 1 Introduction

The boundaries which demarcate educational technology as a subset of information technology are not self evident, and there is often debate about whether it is best to develop a specifically educational technological approach, or to make use of generic technologies. In The Institute for Educational Cybernetics (IEC) at the University of Bolton, where all the present authors work, we make use of generic technology wherever possible, as well as stressing the importance of interoperability specifications for both educational and generic technological development. At present we are engaged in building a widget based application store, and the principal contribution of this paper is to consider the ways in which this software paradigm is relevant to the problems of delivering flexible services across physical and virtual spaces in education.

A major focus for educational technology in recent years, both for the field at large and for our own work, has been the provision of services and their management by learners and teachers. From a technical perspective the challenge has been to deliver them across a wide range of virtual and physical spaces, while from the pedagogical side the ability of teachers and learners to define, select and make use of services has been an on-going issue, both in personal learning practice and in the design of classroom activities. This led the authors to develop the Wookie server, now Apache Wookie (incubating) [Apache Foundation 2011] (henceforth 'Wookie'), in order to deploy and deliver W3C widgets. However, this focus on the provision of generic technology creates a need for a theoretical and practical exposition of the ways in which a particular technology can be applied in education. This is the task which we undertake in the present paper.

In another paper we describe the important role of interoperability specifications in enabling the educational use of technology [Griffiths et al. 2012]. In it we set out the approach taken to technical development of the Wookie server and its relationship to particular interoperability specifications. In this paper we turn our attention to the ways in which the infrastructure meets the functionality foreseen as being required by the pedagogic perspectives which informed it, and the areas of need which remain outstanding, or which have emerged in practice. To do this, we revisit the pedagogical challenges as they were formulated when development was commenced, and relate these to the existing infrastructure, that which is under development, and desirable but unaddressed functionality. In this way we provide a case study of the relationship between generic open source infrastructure development, and the discourse around pedagogy.

We commence our analysis in section two by analysing the functionality foreseen as being necessary for the Personal Learning Environment (PLE), and discuss the degree to which Wookie fulfils this.

In section three we consider the pedagogical challenges raised by the use of computers and interoperability specifications to orchestrate or mediate classroom activities, and the learning design approach. Building on this we consider what might constitute 'good teaching', and how it might be supported, and the degree to which Wookie provides this.

In section four we summarise the unfulfilled functionality which we have identified in the previous two sections. We argue that there is substantial overlap in the infrastructure required for both approaches to educational technology, and propose that app stores have significant potential in addressing the pedagogical and technical issues which have been identified. We place emphasis on the needs of teachers who are working in a technologically enhanced educational environment, as it is this context that has driven our current implementation work.

In section five we consider the ways in which an application store could resolve this, and describe a number of inter-connected projects which are establishing a solid open source infrastructure for development of app stores. Finally in section six we indicate some emerging areas of research and development.

## 2 Institutional systems and the individual learner: the Personal Learning Environment

In a seminal paper of 2001 [Olivier and Liber 2001] it was pointed out that "A problem for the lifelong learner is that since they may be registered at multiple institutions, they may have to use different learning environments at each". The learner is expected to navigate across virtual spaces in order to reach a specified web location each time they want to engage with a course, and to remember the relevant passwords and how to use each user interface. Olivier and Liber proposed an alternative approach which they called a Personal Learning Environment, in which the relationship between the learner and the institution was inverted. It was envisaged that each institution would provide services relating to a learner's courses and progress, which could be brought together by the learner in an environment or environments of their own choice. In this way a group of learners could participate in activities which were distributed across a number of virtual spaces. This proposal gained substantial traction, leading (amongst others) to the Mashed Up and Personal Learning Environments (MUPPLE) workshop series at the ECTEL conferences of 2008, 2009 and 2010, the Personal Learning Environment Conferences of 2010 and 2011, and to a special issue of the Journal of Interactive Learning Environments which we draw on in our discussion below [Johnson and Liber 2008 (a)];

An analysis of the dimensions of challenge to the viability of the learning process when using a PLE is provided in [Johnson and Liber 2008 (b)]. This is based on an analysis of the individual's use of technology, and was classified by drawing on Beer's Viable System Model [Beer 1985]. They identify four dimensions of challenge to the viability of the learning process, shown in italics in the following paragraphs. These are not technical challenges, but they have technical implications, some of which are shared between challenges, which we identify below. We do not suggest that the technical interventions resolve the challenges, rather that they remove at least some of the barriers to addressing them. We examine each in turn and consider the degree to which Wookie provides the necessary functionality, and the degree to which it falls short.

*(a) The challenge of unmanageable complexity within the domain of tools to be managed.* An attempt by teachers and learners to understand and manage the whole Web confronts them with an unmanageable variety of resources. The task of the user in identifying tools needs to be simplified, so that they can focus on fulfilling the objectives which led to their engagement with the technology. The proposition is that

the restrictions of a more limited range of functionality are more than offset by an increase in personal efficacy. A virtual learning environment performs this function [see Britain and Liber 2004], but does so by reducing the range of available tools to a degree that many users find too constraining, and their use is restricted to the servers of the institution which controls them. Apache Wookie (incubating) provides more flexible constraints by attenuating the variety of available technological choices in two ways.

Firstly there is the technical constraint of focusing exclusively on widgets (initially compliant with the W3C specification, but now also including OpenSocial). A significant task in the development of Wookie was to extend the functionality of widgets so that they could provide a sufficiently rich set of functionality, for example to provide multi user collaboration, and to enable specified users to be able to manage the settings of the widget (for example, a teacher role which can decide when a vote process should be closed). The constraint in the selection of tools is balanced by an expansion in virtual and physical spaces where they can be used.

Secondly, the choice of tools made available to the user is determined by the selection of widgets by the administrator of the particular Wookie server to which they have access. We refer to this as *curation*, to indicate the intention to support the provision of a set of widgets gathered in order to serve the needs of a particular group of users. However, the functionality to support this in Wookie is rather limited. *Unfulfilled functionality: support for curation of widget collections*

*(b) The challenge of over-focusing on particular technological practices at the exclusion deeper process of environmental adaptation.* When tools are complex users may be reluctant to try out alternative technological practices, because the time and effort involved are too great. They may also be locked-in to a tool-set by institutional policy (use of a specific platform or application), or by vendors (through the use of proprietary formats or anti-competitive practices). In order to ensure that their tools support their adaptation to their environment, users must be able to install and use alternatives as simply as possible, and to avoid personal or institutional lock-in to technologies (which may result directly from technical constraints or indirectly from the users disposition). Apache Wookie (incubating) ensures that tools are very simple to install, and minimises lock-in by being open source, and making use of open specifications for the functionality offered. However, in the current version of Wookie, the user looking for a widget is limited to browsing a web gallery of those available, with no social annotation, tagging, record of use, or other support for discovery. *Unfulfilled functionality: support for discovery of widgets by the user*

*(c) The challenge of being unable to tie technological action to personal identity.* It has been shown that the desire to customise technology is to be seen across the world, as discussed by [Cui et al. 2007] in relation to mobile phones. According to [Marathe and Sundar 2011], p.788, this customization of mobile phones "is tied to two psychological constructs – sense of identity (SOI) and sense of control (SOC) – and shows significant effect on both constructs, compared to simple browsing of content". Mobile platforms provide users with relatively constrained opportunities for determining the functionality of their devices (as they can in general only choose between applications, and cannot themselves program their device), but these constraints enhance the ease with which technological action can be tied to personal identity. The example of mobile phones provides evidence of the importance of this

challenge, but this functionality is only available to a very limited extent in institutionally controlled environments, such as education. Wookie provides a means of delivering a constrained set of components with powerful functionality, but the interface is less streamlined than that of a mobile phone (see the also the next challenge). *Unfulfilled functionality: richer opportunities for users to structure their technological environment, balancing constrained functionality with ease of modification.*

*(d) The challenge of being unable to predict changes in the environment and adjust technological practice accordingly.* Systems which address institutional problems are relatively stable, and so agile changes in functionality and patterns of use are not essential. The user of a PLE needs to be able to respond to their changing view of the needs generated by their environment, and to be able to easily introduce changes in functionality. This depends not only on technical interoperability, but also on the 'ready-to-hand' availability of tools and services, in Heidegger's sense [see Heidegger 1962]). The user can introduce changes with the current Wookie infrastructure, but the processes involved are rather inflexible, and more appropriate for an environment which evolves over weeks or days rather than hours or minutes. Computational models may be able to enhance predictive capabilities, and we have done work in this direction [Johnson and Sherlock 2011], but it is out of scope for this paper. *Unfulfilled functionality: greater 'to-handness' of the available services.*

All this unfulfilled functionality relates to the technical problem of enabling learners to work in an environment which they own and control, but which draws services and resources from across a number of different virtual environments provided by institutional systems. Severance [see Severance et al. 2008], writing in the same special issue as [Johnson and Liber 2008 (b)], provides a perceptive analysis of the technological implications of a PLE. The three issues which he identifies provide us with the context for understanding the technical role of interoperable widget services in enabling learners to participate in learning activities across different spaces.

Severance identifies the technical problem underlying the PLE as being that extension points in Virtual Learning Environments (VLEs) are proprietary (and so lead to lock-in), and that control of the extension of functionality is typically restricted to VLE system developers and administrators. He identifies three areas which need to be addressed by open specifications in order to support PLE functionality. At the same time parallel work was taking place on the Wookie widget server [Wilson et al. 2008]. In retrospect it can be seen that Wookie fulfils much of the functionality which Severance identified. We review this progress, and will return to the outstanding issues in later sections. We give these Severance's requirements in italics, followed by the functionality of Wookie which addresses the requirement.

*(1) provisioning to initially establish the learning context and the agreement between the consumer and producer (this is often done by the instructor or owner of the learning context before the first learner can use the learning context).* To achieve this, the user (a teacher or a learning technologist) must be able to designate an area within a learning context (available to a cohort in a course) for the display of a resource or service. As Severance adds, "For this VLE/PLE architecture to work, tools must operate and respect the learning context in which they operate. The tool must act as if it is dedicated to the learning context and take all configuration and

authorization from it. There are very few general-purpose tools that understand this notion of context". Widgets are natural fit for this task, because they focus on small chunks of functionality which can be deployed in a wide range of containers in differing contexts. Wookie deploys W3C widgets across a range of spaces, while ceding the responsibility for configuration and authorization to the container. In this way the same widget based tool can be used in, for example, Moodle and WordPress [Wilson et al. 2011]. Indeed the same instance of a tool can be used, and so learners on both platforms could participate in the same conversation.

*(2) establishing a user session for an individual joining a particular learning context.* This is handled transparently by the Wookie server, without the need for intervention by the teacher or the learner.

*(3) run-time services needed by the producer application and provided by the consumer application.* Wookie provides a set of run-time services to support the functionality of widgets. These include all the functionality associated with W3C compliant widgets, which will run in a wide range of Web platforms. In addition, the services enable users who have access to a context to collaborate with each other through multi user widgets, or to enable teachers to set the preferences that control the functioning of a widget.

When the VLE was developed it was safe to assume that all interactions with learning technology would be by means of a desktop computer, or possibly a laptop. This assumption remained valid when the PLE approach was developed. However, in recent years the situation has changed completely, with the emergence of successive waves of innovation in phones, tablets and eBook readers. This has created a set of new physical spaces to be traversed, and with them a new set of problems. These could be safely ignored by educational technology as a whole, and a PLE approach in particular, at the time when Severance was writing, but now become urgent. In the first place, will the target device be able to run the necessary software (for example a Java applet, or a Flash file)? This is a classic interoperability issue, and a major attraction of the use of W3C widgets is not only that they are a standard format, but also that they make use of the most widely implemented technologies which make up the Web: HTML and JavaScript. This ensures that services and resources provided through Wookie can be run on any standard Web platform, no matter what the device. However other issues remain. Is the interface of the services usable on the screen of the target device comprehensible (alone or in combination), and can the user conveniently provide input on their device?

*Unfulfilled functionality. Support for multiple screen factors and user input devices.*

## 3 Supporting learning activities in the technology enhanced classroom

We now turn to teacher led activities in the technology enhanced classroom, and how these might be supported. This is closely related to the discourse around the learning design approach, and also to the IMS Learning Design (LD) specification [IMS Global Learning Inc. 2003]. There is an extensive literature around the merits and demerits of IMS LD, to which we have contributed (for example [Griffiths and Liber

2008]). Suffice it to say here that its purpose is to provide a formal description of a pedagogic plan which enables a computer to orchestrate the learning activities which are represented. While originally conceived for distance learning, it has also been proposed as a support for face-to-face learning. However, in this regard it has been criticised for restricting the ability of the teacher (or learners) to respond to emerging circumstances (although apportioning blame for this between the specification and the infrastructure for making use of it has been a moot point). As we have discussed elsewhere [Griffiths et al. 2012], it was implementation of this specification which led originally to our original implementation of Wookie, and a desire to provide greater flexibility in the available services was a prime motivation.

In face to face learning today there are highly developed and standardized curricula and expensively developed learning materials (particularly in the form of text books). There are also increasingly sophisticated frameworks for lesson planning (which are often compulsory) for example No Child Left Behind in the US [107th US Congress 2002], and the increasingly influential role of the Office for Standards in Education (OFSTED) in the UK. The pedagogical approaches which support these initiatives include 'Direct Instruction' promoted by organisations such as the Success For All Foundation. They argue that “Education is not about guesswork or shooting in the dark. It’s about expanding the use of proven solutions in classrooms and schools”, and that teaching is a matter of applying the correct method [Success for All 2011]. Despite these efforts, the results obtained by teachers are seen to vary greatly, as shown by the efforts by government agencies to reward “great teaching”, and to take action against “bad teachers” [Neal 2011]. However, in education policy and training there is often little idea of what this corresponds to in classroom practice, and the difference between teachers is frequently ascribed to vague concepts such as inspiration, experience or personality. This lack of clarity may (or may not) be problematic in face-to-face learning, but when we seek to use technology to facilitate these processes it is potentially disastrous, because we can only implement that which we make explicit. The danger is that as technologists we find ourselves in the position of implementing systems which attempt to bottle an unspecified magic dust which “great” teachers deploy in transforming curricula into marvellous learning experiences.

In attempting to make this problem tractable we have found two theoretical frameworks valuable, which we briefly introduce here in order to indicate our perspective on the task of the teacher. The first of these is Pask and Scott's Conversation Theory [see Pask 1975], which was adapted by Laurillard [Laurillard 1993], and was applied by Britain and Liber in developing their framework for the evaluation of VLEs. According to this model “The process of learning is supported by the creation of interactive “micro-worlds” (learning activities) in which the student can actively engage in practice that enhances and reinforces the ideas that have been formulated through discussion. The model emphasizes that these activities should be created and adapted on the basis of the conceptual dialogue, rather than pre-set in advance.” [Britain and Liber 2004].

This is problematic for approaches which take as their starting point the separation into a design process and a runtime process, as does much of the work in the learning design tradition [see Koper and Tattersall 2005]. This is not necessarily a mortal blow to learning design approaches, as it corresponds closely to the traditional phases of curriculum design and lesson planning (design time) and the teaching process (runtime). However, in a face-to-face class the teacher has considerable freedom in creating and changing activities, which is constrained when computers are involved. One reason for this is that the installation of applications, and their provisioning with user information, takes time and expertise. Consequently the teacher is under pressure to state, before the class starts, which applications they would like to use. This goes hand in hand with the increasing managerialism in educational institutions which is keen to gather information on the activities being carried out in the classroom, in order to apply quality control measures and cost controls (see [Deem 2007] for a discussion of this in higher education). From this perspective, also, the choice of applications should be made in advance, and unlike other aspects of teachers' delivery, this choice can be enforced.

The second framework is Harré's positioning theory. As we have discussed previously [Johnson et al. 2011], Harré sees role as being conditioned by social structure (e.g. I am a teacher, you are a student; I am a scribe, you are a facilitator), while positioning is identified as the emergent effect of particular normative conditions, particular communicative acts and particular narratives (e.g. something terrible has happened to me, you are sympathizing; I care about your opinion, you are considering your position). An analysis of the exchanges of communicative acts, and evolving or switching narratives, can provide insight into way in which teachers modulate the learning activities within a conversational framework.

From the perspective of these theories we can see the teachers task in face-to-face teaching as being the management of conversations which, in terms of their organisation, are articulated both by the activities defined for the class, and by the introduction of new resources which they have to hand (for example, a map on the wall, or a drawing on the blackboard). In terms of their execution, these activities constitute a frame within which certain kinds of positioning are possible or excluded. For example, the learners may be constrained to listening, or may be able to discuss their own theories. The teacher can further modulate the positioning of the learners within these conversations, through a verbal intervention encouraging or suppressing certain kinds of interaction. These two ways of viewing pedagogy translate, in practical terms, to coordination processes, which resolve questions such as:

- "What alternative activity could I usefully do in my Maths class, because the learners are getting frustrated by so many quadratic equations?"
- "It's the first sunny day of the summer. How can I get that group at the back to stop thinking about cricket and start discussing Plato's idealism?"
- "I only have ten minutes to left in the class, so we can't start on that new topic. What can we do to practise what we have already looked at?"

It is possible to conceive of how computers could provide support for resolving such questions in 'microworlds' and their modulation through positioning, but, to achieve this, (a) if the role of the teacher is seen as valuable, then orchestration of activities needs to be in their hands (rather than computers) and (b) tools need to be as 'to hand' as chalk and duster, switching between them as agile as it is for these technologies in the classroom. At present this is by no means the case. To make this point clear, let us consider the processes which teachers need to go through to use applications with their learners. These will include most, or all, of the following:

**Appropriateness**: a desktop application will often need to be approved by the IT coordinator as being legal and non-harmful. Web applications need to be checked to avoid violating institutional guidelines, and perhaps need to be approved for use.

**Access**: desktop applications will need to be installed on all computers by the IT coordinator. For Web applications the teacher must ensure that the class can access the site (perhaps by requesting that the Web address be included on a whitelist)

**Navigation**: When the time for the lesson arrives, the teacher will have to instruct the learners, or demonstrate, how to launch a desktop application, and how to run it. A great deal of time can be spent with instructions of the type

Teacher: “No, not the green button, it's the drop down menu on the top left”
Learner: “What is a drop down menu Miss?”

For a Web application, the teacher must either set up a list of links for use in the class, or guide the learners in navigating to a website, or both

**Monitoring**: The teacher walks around the classroom checking progress on each machine, ensuring that the learners are looking at the right page, and are carrying out the right activities.

None of these steps is technically challenging, but their cumulative weight, and in some cases the need for prior administrative approval, makes agile deployment of applications completely impracticable. Consequently, there is a risk that teachers will be unwilling to use IT applications in their classes, because a) they make it harder for the teacher to articulate the conversations and positioning which (in our view) is at the heart of the pedagogic process, and b) the set up and provisioning of the applications occupies time which could have been used for more agile interactions without computer support. The consequence is that the introduction of technology makes it harder for the teacher to manage the variety of the states and interactions which the learners manifest. As we have seen with the PLE, a more ‘to-hand’ toolset is needed.

Writing independently of the technology enhanced learning questions which we have been examining, Sawyer has described teaching as an improvisational activity, since “Conceiving of teaching as improvisation highlights the collaborative and emergent nature of effective classroom practice” while accepting that “The best teaching is disciplined improvisation because it always occurs within broad structures and frameworks [Sawyer 2011] (p.2). This perspective leads to a view of the teachers task being the judicious combination of, on the one hand, preparation which is appropriate to the learning context, and on the other an improvised response to emergent circumstances within this context. For both these purposes, particularly the latter, teachers need control over the resources and services they provide for their learners, just as a PLE user needs to control of their own technological environment.

Control is often understood (or misunderstood) in some discussions around education as being synonymous with discipline, as in the phrase 'classroom control'. However, this is not the meaning here. Rather we refer to the processes which make it possible to manage a system so that it attains or maintains a desired state. Moreover the control which we refer to is highly nuanced. Firstly, it should be remembered that context should be understood as referring to a particular group of learners participating in a particular learning activity. Secondly, control does not simply refer to publishing a tool to the class. It may consist of providing the means whereby a conversation may be redirected by providing a tool for subgroups of the class, or whereby the teacher changes the state of all the screens of the users, or whereby the teacher receives input from learners hand-held devices, or whereby learners are provided with a distributed interface with which they can mutually control a device. If this is to be practicable, the control of services and resources needs to be as intuitive as, and at least as agile as the blackboard, the map on the wall, and the encyclopaedia.

As discussed in the previous section, W3C widgets delivered by Wookie provide a means of delivering services which are specific to a particular learning context and enable that learning context to be spread across a number of virtual and physical spaces. This can be achieved to a high degree of precision, with programmed adaptive capabilities, using IMS LD editors integrated with Wookie, to which the present authors contributed [Griffiths 2009]. However, the complexity of the set up process, and the lack of flexibility in the runtime system, meant that the improvisational aspects of the teaching task were unattainable. Moreover, the learning context is defined by the application through which the user accesses a widget, and these are not capable of the kind of control distinctions for which we give examples in the previous paragraph.

## 4 A focus on teacher's control of their technological environment

To summarise, the PLE discourse identified the categories of control which users required, and the technological characteristics that could provide them. In retrospect it is clear that the richness of these ideas showed that the conception of the technology required to support the teacher in a face-to-face environment was impoverished, and this strongly informed the present authors' view of classroom technology. Our consideration of work carried out around learning design has clarified the central importance of not only providing this functionality, but also making it available 'to hand' for improvised activities. In this we are perhaps sketching out a 'personal teaching environment', though without wishing to understate the importance of personal technology (nor with any desire to establish a new e-Learning acronym!).

It can be seen that both discourses lead to the same general statement of needs: that users need greater support in deploying and managing services across a range of spaces in response to the requirements of teaching and learning of a particular person or group in their technological environment. This is to be done by enabling a finer grained mapping of services to context, and by providing greater control to learners and teachers in deciding which services should be deployed in that context.

Firstly, summarising the points made in section 2, we can see that Wookie supports *context* by

- Enabling services to be made use of in a wide range of applications, platforms and devices, making use of the W3C widget specification
- Providing additional functionality which is required for social and educational contexts (e.g. collaborative widgets, persistent data, etc.)
- Resolving the problem of user authentication on multiple services ceding responsibility for configuration and authorization to the container from which the service is accessed (e.g. a VLE)
- Enabling different sets of services to be made available for deployment to different learning contexts (at the level of an activity rather than a course)

An unmet requirement for the support of context is:

1. improved support for multiple screen factors and user input devices.

Secondly, regarding *control*, the support provided by Wookie for control in a PLE is relatively restricted, enabling users to insert a widget in a context (assuming that they are authorised to do so). From our consideration of learning design in face-to-face teaching we have seen that the functionality envisaged for a PLE maps closely onto that required by a teacher to control the educational activities that they offer to learners. However, there is a need to provide the teacher with greater control in responding to the evolving situation in the classroom (precisely where current infrastructure, including Wookie, is weakest). In seeking to support teachers' control of emerging educational requirements the PLE and learning design perspectives therefore indicate that it would be valuable to meet the unfulfilled functionality identified in section 2:

2. Support for curation
3. Support for discovery of widgets by the user
4. Richer opportunities for users to structure their technological environment, which balance constrained functionality with ease of modification.
5. Greater 'to-handness' of the available services

We now turn to a consideration of how current work on widget infrastructure is addressing these issues.

## 5 Application stores: opportunities and drawbacks

Thus far we have discussed the ways in which the existing functionality of Wookie is related to the PLE and learning design approaches. We now review the ways in which a number of related current projects are developing an infrastructure with an application store at its heart, and consider its relevance to the unfulfilled functionality which we have identified.

Application stores are also known as markets, widget stores, or 'app stores' (as we will henceforth refer to them here). Their defining characteristic is that they provide (a) a very simple means of locating and installing new functionality on a computer or device (b) an easy to use search facility for identifying new functionality, and (c) a means whereby the experience of users can be represented in order to support discovery (for example through comments, user reviews, ratings, recommendations and popularity metrics). The concept is not new, and has developed within the open source software movement. The first store was dpkg for Debian Linux, which was taken further by the Advanced Packaging Tool (APT), a command line tool for installing, and delivering software from a variety of servers. In 2003 a graphical user interface, Synaptic, was added to create the functionality familiar from app stores today [see Wensink 2012].

However, it is the emergence of mobile technologies which has given app stores a high degree of penetration on Android and iOS phones and tablets. The success of these platforms has in turn influenced mainstream computing, and Steve Jobs is reported to have said in 2010 that the iPhone store had "completely revolutionized how people get the apps. And why not the Mac, too?" [Muchmore 2011], and this led to the development of the Mac App Store for Apple computers, to be followed by the Windows Store.

App stores provide control to the user by making it a trivial task to install new functionality on a phone or tablet, and hugely simpler than installation of software on a PC. Very large numbers of applications are available, and support is provided in finding them. Given that app stores are so effective, why is there a need to develop a new infrastructure? A practical response is that existing app stores tend to focus on either computers or mobile devices, rather than attempting to bridge the two. A deeper answer returns to the theme of control. While Richard Stallman's statement that the iPad “represents a serious threat to individual freedom” which is “like handcuffing yourself with a pair of digital shackles” [see Manacorda 2010] may seem extreme, there are clearly issues around who owns the infrastructure, and many educational institutions may feel uncomfortable about handing over control of services to an external organisation. Secondly, the content of app stores is controlled by those who own and run them, and in the case of Apple this control is very tight. On the one hand there are limitations on programming the device, which while comprehensible as a measure to increase reliability of Apple's product, it is diametrically opposed to the emphasis on programmable devices adopted by, for example, the Raspberry Pi foundation [Joyce 2012]. On the other hand, Apple controls access to the store, rejecting any apps which, for example, it deems as duplicating functionality of its products or other apps in the store, or which offend against its policies on sex or the consumption of alcohol. While Android app stores are more flexible, the same underlying issues remain. There have been initiatives to provide app stores for learners. For example Kindertown (www.kindertown.com) is an iPhone based educational app store that provides an additional level of control, filtering the apps which are available, so that it is easier to locate educational apps, and apps which may be considered inappropriate for young learners can be excluded from the system. Such a service, however, simply provides still tighter constraints imposed by unnamed educators. We argue, in contrast, that open source app stores should be provided which can be adapted to the needs of teachers and institutions, individually or combined, so that control can be exercised at the level where it is most appropriate for pedagogic, organisational and legal reasons in any given location and jurisdiction.

Considering the five aspects of unfulfilled functionality identified in the previous section, the technical issues 1 and 2 concern the description and discovery of widgets, while point 4 concerns the to-hand quality of the services. We consider these in separate sections below. Regarding point 3, the success of app stores in encouraging phone owners to customise their devices with software strongly suggests that they provide a convincing trade off between constrained functionality and ease of modification. We do not claim that the provision of app stores resolves the problem of enabling users to structure their technological environment, but it does seem reasonable to propose that an open source, customisable, implementation of an app store will provide a rich tool for exploring the potential which this paradigm offers.

## 6 Describing and discovering widgets in an app store

The fundamental role of an app store is to make available a set of applications for use on a given platform. However, for our purposes it is also necessary that authorised users can create collections of tools for particular groups, a process we refer to as *curation*. In many contexts, for example a university or school district, support for communities of practice will be valuable in defining the needs, and data related to use in context of the widgets made available. Looked at from the point of view of the teacher or learner this is a question of how to *discover* suitable widgets. From both perspectives the problem is one of appropriate and effective descriptions.

The first version of Apache Wookie (incubating), release 0.9.0, provided a Web page displaying a list or grid of the available widgets, which could be embedded in a plug-in for the host application where the widget was to be displayed [REF Scott]. This is manageable on a small scale, but as the number of widgets increases it becomes much harder to use. The iTEC project is using Wookie as a key element in its infrastructure for organising large scale pilots to promote innovative practice in schools across Europe [see Vuorikari, R], and in our work on this project we have discussed having hundreds of widgets to provide a wide choice of tools and resources. Clearly the current interface would be impractical on this scale.

As well as iTEC other projects are making use of Wookie, including ROLE (supporting PLE functionality, see www.role-project.eu) and LTfLL (supporting Life Long Learners, see www.ltfll-project.org). iTEC and ROLE have come together with an Open University initiative to create a UK higher education app store, and are participating in the JISC funded EDUKApp project, including contributions from the present authors (see http://code.google.com/p/edukapp/). The outcome will be a cross-institutional store for widgets related to teaching and learning, building on Apache Wookie, Apache Shindig, Apache Solr and Apache Shiro. The store itself is a backend service primarily accessed using JSON APIs to make it easier to create a new user interface, although there is a default interface that uses Twitter Bootstrap, which is straightforward to customise. The app store will enable widgets to be tagged and commented by users and administrators, grouped by categories and searched, and it will be possible to embed it in plug-ins in a similar way to the present Web interface to Wookie. It is then foreseen that each participating project will use the infrastructure to build their own widget store.

Rather than attempt to extend the existing interface, it was decided to abstract this functionality out of the Wookie server itself. Currently work is being undertaken within Apache Wookie (incubating) to remove the present Web interface, and to create an API to be the sole source of information about the widgets held by the server. In parallel EDUKApp is developing an app store which communicates with a Wookie installation. Following the example of ROLE [see Law and Chatterjee 2010] support will be provided for a Social Requirements Engineering (SRE) to allow external communities of users to submit and vote on requirements for the development of new widgets, and improvements in existing ones. They will also be provided with social tools to comment on and tag widgets, to aid discovery and to inform curation decisions. Further support for curation will come from the paradata provided by the SPAWS project, discussed below. Teachers and/or students will be able to upload and tag widgets they have created or found, and this necessitates that the curator of the collection is provided with an approval mechanism.

A challenge identified by iTEC is to describe widgets so that it is clear if they are appropriate to the scenarios and activities that teachers carry out. In the first place, there is a substantial overlap in the functionality of online tools, and hence in the way in which they can be applied in different spaces. For example a forum and an email list are both ways of managing threaded discussions. Secondly, when confronted by more than one tool that can do a job, users' selection depends on factors beyond functionality (personal preference, custom and practice, institutional policies, technical barriers, etc.). This problem has been addressed in prior work, particularly in the 'Soft Ontology' approach developed by the EU funded iCAMP project [Laanpere et al. 2006] which has provided the basis for the approach adopted for the EDUKApp store. Rather than classify the widgets into categories using a taxonomy, they are related to their affordances [Gibson 1977] [Norman 1988] for the various 'things that need to be done', which are separated from descriptions of their technical capabilities (operating environments, language, interoperability capabilities, requirements, etc.). Figure 1 below indicates how tools are related to activities and affordances, which are represented on a numerical scale, indicating the degree to which they meet the affordance. When planning an activity, a teacher identifies the affordances they require (i.e. "in order to do this activity, we need to be able to…") and these are mapped via scalar linkages to provide recommendations for appropriate tools.

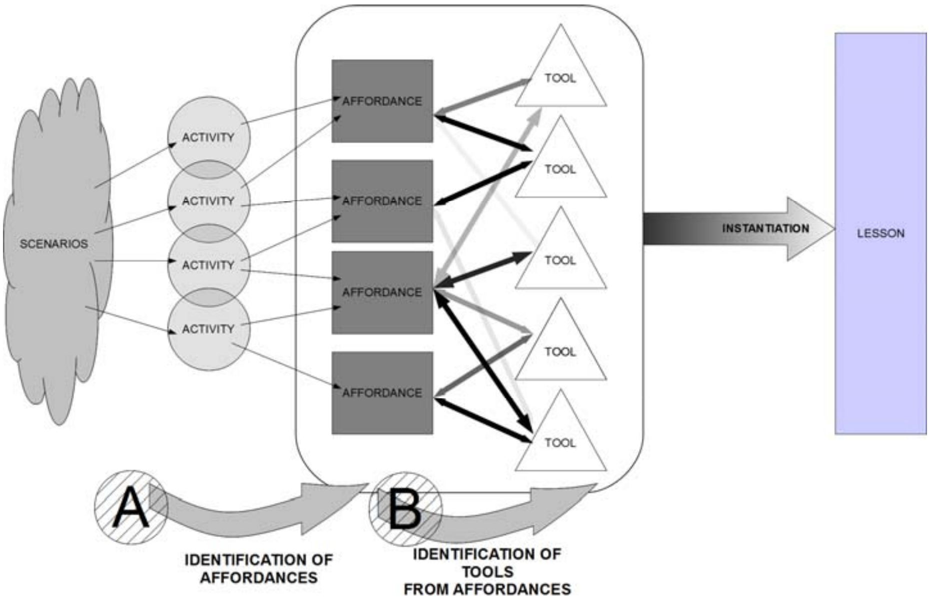


*Figure 1: From scenario description to affordances and tools in iTEC (taken from iTEC deliverable D8.1 [see Griffiths et al 2011])*

The EDUKapp store is open source, and it is intended that it should be customised and deployed by organisations targeting different sectors and regions. However, this also means that download statistics, comments and ratings, become fragmented between the various installations, and valuable information cannot be taken into consideration in recommending tools to users. Such information about usage is becoming known as *paradata* [Van Gundy 2011], and an open source project in the United States, the Learning Registry (http://www.learningregistry.org/) is building an infrastructure for sharing this kind of metadata structures. Another project, Sharing Paradata Across Widget Stores (SPAWS) has been funded by JISC in the UK (http://www.jisc.ac.uk/whatwedo/programmes/ukoer3/rapidinnovation.aspx) in order to pool the paradata generated by the different widget stores, and so to enhance searching across them. This work is being developed in close collaboration with EDUKApp, where it will be applied, and makes use of both the Learning Registry Paradata “actor-verb-object” JSON approach [Activity Streams Working Group 2011], and Contextualized Attention Metadata (CAM) [Schmitz et al.]

## 7 A tool to hand

In the previous section we describe how users can discover widgets, and how they can choose between tools. However, as we identify in point 4 of section 4, to be useful in responding to emerging needs, tools also need to be ready-to-hand. Clearly this is a function of the user's disposition, rather than something inherent in the tool or the infrastructure, but these can create the conditions whereby the possibility of tools being ready-to-hand is not excluded. In the following discussion of how this should be done we draw closely on our work in the iTEC project [see Griffiths et al. 2011].

In practical terms, the launching of a widget/gadget in a personal space should be unobtrusive and actions required from the user side should be limited to a minimum, irrespective of the actual platform the user is embedding the widget into, which may be a personal technology space or an institutionally-managed platform. iGoogle, one of the prominent initiatives to create a personal gadget place, fully integrates a market place for gadgets giving the end-user the option to install new gadgets in a one-click operation. This pattern is followed by Netvibes and PageFlakes. Opera provides a default widget handler using the application/widget content type (and .wgt suffix) to automatically prompt users to install and launch a widget from their desktop when a link to download it is clicked. These mechanisms mean that for the user there is little difference between launching an application and installing a widget; so long as they know that the functionality is available, they can easily find the place to launch it.

It is this ease in adapting a technological environment in response to emerging needs which makes it feasible for tools to be ready-to-hand. The Wookie server provides some initial support for this, as it is possible to switch between the widgets displayed in a context at any time, but in practice this capability is constrained. For example in Moodle, in common with many portal environments, integration of widgets is achieved by a plug-in (in Moodle a *block*), which has to be inserted into the page design [Wilson et al. 2011]. For example, in Moodle a user with an admin role has to enter the edit mode, and add a “block” to a course page from a drop-down list of available block types. While this is not complex, the teacher may not have the necessary access privileges, and in any event the process may be sufficient to deter changes to the widgets provided to support activities as they are being carried out.

### 7.1 Scenarios for embedding a widget

The process for embedding a widget will vary according to their activity and the environment that they are using, which leads us to identify the following scenarios which we defined during our work on iTEC [see Griffiths et al. 2011].

1. The user is logged into an existing institutional platform, such as a learning management system or interactive whiteboard space, from which they directly trigger the launch of the widget store within the platform interface. When the user selects a widget, it is automatically added to the space from within which the launched the store (for example, using IMS Basic LTI, or a connector that uses an web API exposed by the widget store).

2. The user is not logged into a platform that has a standard means of integration with the widget store, and instead launches the widget store from their browser. The user is however using the Opera web browser, and so a "launch widget" button will trigger Opera to download, install, and launch the widget on the user's desktop.

3. The user is not logged into a platform that has a standard means of integration with the widget store, and they instead access the widget store using a browser which is not "widget-aware". The user is an iGoogle user, however. The store provides a range of embedding options to the user from which the user selects the "add to iGoogle" button.

4. The user is not logged into a platform that has a standard means of integration with the widget store, and they instead access the widget store using a browser which is also not "widget-aware". The store provides a range of embedding options to the user, from which they select the option of downloading the .wgt file to side-load onto a mobile device (e.g. Blackberry, Android device running Opera etc.).

5. The user is not logged into a platform that has a standard means of integration with the widget store, and they instead access the widget store using a browser which is not "widget- aware". The store provides a range of embedding options to the user from which they select a gadget.xml URL which they copy and paste into another OpenSocial-aware platform.

6. The user is not logged into a platform that has a standard means of integration with the widget store, and they instead access the widget store using a browser which is not "widget-aware". The store provides a range of embedding options to the user from which they select JavaScript embed code which they then copy and paste into a generic web platform.

7. The user accesses the widget store directly from their browser, and wants to use a widget directly there and then without embedding it or including it in another platform. They select an option to launch and use the widget in a new window.

8. The user is accessing the widget store in one of the above ways, but wants to actually use it in another. For example, they access the store from Moodle, but want to install the widget onto a Blackberry; or they access the store from Opera, but want to embed the widget in iGoogle.

This set of scenarios makes plain that the route of greatest convenience for each user depends heavily on their context - how they arrived at the store, what they are using to access it, what other services and devices they have available. So a single embedding strategy will be insufficient - instead a widget store needs to support as many options as possible and to present them to a user based on their context. However, scenario 8 also means the store has to provide users with the means to choose alternatives, as the context in which they access the store may not be the context in which they wish to actually use the widget.

In principal the EDUKapp store infrastructure can meet all these scenarios, but the actual functionality provided will depend on the specific app store implementation and deployment. At an architectural level, the scenarios indicate that the store infrastructure has to be deployable in three different configurations:

- the widget store is directly user-facing, and any embedding is handled in a very loosely-coupled fashion - for example, an “add to iGoogle” link, or copying and pasting JavaScript or a URL, or providing a download link
- the widget store is directly user-facing but is embedded within another platform, such as a learning management system or portal, from which embedding of widgets is handled using an API such as IMS Basic LTI or connector component to create a seamless platform experience
- the widget store is a back-end service used by another platform to integrate into its own widget selection interface, maintaining the end-user platform’s look-and-feel but extending the range of widgets available to users

### 7.2 Simplifying the host environment for widgets

Widgets need to be embedded in host environments, because they have no means themselves of managing authentication and access. However, many portal platforms were designed before the advent of widget technologies, and are not set up to make this easy (hence the complexity of the scenarios above). Moreover, in order to provide a container for widgets, it is often necessary to deploy a heavyweight portal environment, even though much of the functionality is not required. A dedicated widget based portal could be much lighter and easier to administer, as well as providing support for widget services. Apache Rave (henceforth ‘Rave’) fills this gap, and provides “an out-of-the-box as well as an extendible lightweight Java platform to host, serve and aggregate (Open)Social Gadgets and services through a highly customizable and Web 2.0 friendly front-end. … It will also provide strong context-aware personalization, collaboration and content integration capabilities and a high quality out-of-the-box installation as well as be easy to integrate in other platforms and solutions.” [The Apache Software Foundation 2012] In addition to OpenSocial gadgets, Wookie is also integrated to provide W3C widgets, and Activity Streams can be deployed.

Rave also enables us to address the first two aspects of unfulfilled functionality identified in section 4, point 1: *Support for multiple screen factors and user input devices.* It is anticipated that Rave will be aware of the form factor and interface of the device on which it is being displayed, and will be able to adapt accordingly. For example, on a mobile phone the workspace will consist of a collection of tiled widgets, rather than an arrangement on a page.

Using Rave it will be much simpler for educators to provide a set of widgets which support a learning activity, making use of an app store (perhaps implemented using EDUKapp) which assists them in choosing the most appropriate widgets for their particular context. It will also be simple for them to adapt this by changing the widgets or adding more, as the learning activity progresses. It is also anticipated that Rave will support inter gadget/widget messaging. The fact that it can be integrated in other platforms raises the interesting prospect of embedding a rave container in a

VLE, easing the path to adoption. Apache Rave has already attracted interest from educational institutions, with participation in its development from Indiana University (USA), The University of Bolton (UK), and Groningen (Netherlands), the last of whom described it as "a perfect platform for our personalized University Portal" [Khudairi, S. 2012]. The UK JISC funded the Rave in Context project, which is using Rave and Wookie to build a new user interface for the popular eScience tool myExperiment [JISC 2011].

## 8 Looking towards the future

The infrastructure which we describe above provides a solid basis for the development of innovative applications which deliver services across a range of physical and virtual spaces. However, it is worth mentioning some emerging areas which will provide a focus for future research and development.

Firstly, there is a need for a language which can describe workspaces involving widgets, services and interconnections, as a first step towards an interoperability specification. This could then be used to provision Rave and similar applications. The need for this was already recognised in [Sire et al. 2009] who propose a list of necessary elements, but it is only with the development of a suitable infrastructure that the need has become more pressing. The present authors are participating in the Omelette project (www.ict-omelette.eu), which focuses on the creation and delivery of mashups, especially those including telecommunications services, and is contributing strongly to Rave. This work suggests that such a language would need to provide descriptions at three levels. At the conceptual level it specifies the type (category) of widgets needed to meet a particular user goal. At the logical level it defines the layout and the specific widgets to be displayed in a workspace. At the physical level it provides information about access, quality of service and usage needed by a particular platform. With the promised functionality of inter-widget communication in Rave, there may also be a need for a description language for widget dependencies (e.g. this widget should be bundled with a mapping service). One possibility would be to extend the W3C packaging and configuration proposal for widgets [W3C 2011].

Secondly, work on EDUKapp has made it clear that if users want to select widgets on one platform, and use them on another (scenario 8 of the previous section), this presents interoperability challenges. To take one example, a W3C Widget can be used in any web context without any special platform support, but OpenSocial applications often need often need access to OpenSocial APIs and RPC callbacks in the container platform. Mobile platforms present still greater challenges because there are so many widget platforms to be supported. The degree to which app stores will be able to wrap or convert widgets for use in platforms other than those for which they were originally intended, or filter out widgets which are not usable in the selected context remains an open question.

Finally, while embedded browsers are available on many devices, there are many other devices which require additional work if they are to be included in services across spaces. These include in-car units, home appliances and media reproduction systems. This rich vein of future work is starting to be addressed, notably by the Webinos project (webinos.org) funded by the EU, which working on Web runtime extensions, to enable Web applications and services to be used and shared consistently and securely over a broad spectrum of converged and connected devices.

## 9 Conclusions

We have told the story of a number of related software developments, all of which are related to Apache Wookie (incubating), which provide an open source and standards based infrastructure for delivering widgets across virtual and physical spaces. In doing so we have clarified the way in which this infrastructure relates to pedagogic practice, and in particular to the requirements identified by the educational researchers and developers who have been involved in it. We conclude that the Wiki server went a long way towards fulfilling the need for management of context, as identified by Severance [Severance et al. 2008], but falls short in providing users with control over their technological environment. We identify the areas of functionality which remain unfulfilled, and conclude that the app store paradigm has a strong role to play meeting them. The collaboration between three projects to develop an app store infrastructure is not only a heartening example of the efficacy of open source development, but also provides a missing piece in the infrastructure of the delivery of widget services. The use of an app store, however, does not resolve all problems of embedding services, as the iTEC embedding scenarios demonstrate, and this points to an important role for Apache Rave for delivering and embedding educational workspaces across a range of platforms.

We do not suggest that this reflection on the infrastructure we have described resolves the pedagogical questions underlying the PLE and learning design approaches which we have discussed, but rather that they constitute an informed hypothesis identifying a valuable direction for continuing research into educational technology. This will be carried out by the authors in the context of the large scale iTEC pilots to be conducted in schools across Europe, and by other projects which have contributed to the development of the infrastructure.

**Acknowledgements**

We would like to acknowledge the support provided by the European Commission through the iTEC and Omelette projects which are funding our current work reported here, and their past funding for the TENCompetence, ROLE and iCAMP projects. We acknowledge the support of JISC for the EDUKapp and SPAWS projects, and thank OSS Watch for their facilitation of the process of applying for Wookie's admission to the Apache Incubator.

## References

[107th US Congress 2002] 107th US Congress.: “No Child Left Behind Act of 2001”; Public Law 107-110 (2002). Retrieved from http://www2.ed.gov/policy/elsec/leg/esea02/107-110.pdf

[Activity Streams Working Group 2011] Activity Streams Working Group.: “JSON Activity Streams 1.0.”; (2011). Retrieved from http://activitystrea.ms/specs/json/1.0/

[Beer 1985] Beer, S.: "Diagnosing the system for organizations"; John Wiley & Sons, Inc, New York (1985)

[Britain and Liber 2004] Britain, S., Liber, O.: "A Framework for the Pedagogical Evaluation of eLearning Environments"; JISC CETIS report (2004), Retrieved April 15th from www.elearning.ac.uk/resources/VLEFullReport08.doc

[Cui et al. 2007] Cui, Y., Chipchase, J., Ichikawa, F., Tokyo, N. D.: "A Cross Culture Study on Phone Carrying and Physical"; in N. Aykin (Ed.), "Usability and Internationalization: HCI and Culture", Springer, Berlin / Heidelberg (2007), 483-492

[Deem 2007] Deem, R., Hillyard, S., Reed, M.: "Knowledge, Higher Education, and the New Managerialism: The Changing Management of UK Universities"; OUP, Oxford (2007)

[Gibson 1977] Gibson, J. J.: "The Theory of Affordances"; In R. Shaw, J. Bransford (Eds.), "Perceiving, Acting, and Knowing", (1977)

[Griffiths 2009] Griffiths, D., Beauvoir, P., Liber, O., Baxendale, M. B.: "From Reload to ReCourse: learning from IMS Learning Design implementations"; Journal of Distance Education, 30, 2 (2009)

[Griffiths and Liber 2008] Griffiths, D., Liber, O.: "Opportunities, Achievements, and Prospects for Use of IMS LD"; In L. Lockyer, S. Bennett, S. Agostinho, B. Harper (Eds.), "Handbook of Research on Learning Design and Learning Objects: Issues, Applications, and Technologies", IGI Global (2008), 87-112

[Griffiths et al. 2011] Griffiths, D., Johnson, M. W., Popat, K. iTEC deliverable D8.1: Initial investigative summary of tools and development of mashup connectors, (2011). Retrieved from http://itec.eun.org/web/guest/results

[Griffiths et al. 2012] Griffiths, D., Johnson, M. W., Popat, K., Sharples, P., Wilson, S.: "The Wookie Widget Server: a case study of piecemeal integration of tools and services". Journal of Universal Computer Science, Vol. 18, Issue 11, (2012), 1432-1453

Accepted for publication in the special issue on Service oriented architectures, cloud computing applications, services and interoperability, (2012)

[Heidegger 1962] Heidegger, M.: "Being and Time"; Translated by John Macquarrie and Edward Robinson. SCM Press, London (1962)

[IMS Global Learning Inc 2003] IMS Global Learning Inc.: "Learning Design Specification Version 1"; (2003). Retrieved from http://www.imsglobal.org/learningdesign/

[JISC 2011] JISC: "Rave in Context Project Summary"; JISC Research Infrastructure Programme (2011). Retrieved April 15, 2012, from http://www.jisc.ac.uk/whatwedo/programmes/researchinfrastructure/Rave in Context.aspx

[Johnson and Liber 2008 (a)] Johnson, M. W., Liber, O. (Eds): "Personal Learning Environments". Special Issue of the Journal of Interactive Learning Environments, 16, 1 (2008)

[Johnson and Liber 2008 (b)] Johnson, M., Liber, O.: "The Personal Learning Environment and the human condition: from theory to teaching practice"; Interactive Learning Environments, 16, 1 (2008), 3-15.

[Johnson and Sherlock 2011] Johnson, M. W., Sherlock, D.: "Animating the technological transformation of the University: Agent based modelling and the University Flight Simulator; Proc. CAL 11. Learning Futures: Education, Technology & Sustainability,(2011). Retrieved from http://elsevier.conference-services.net/resources/247/2253/pdf/CALX2011_0304.pdf

[Johnson et al. 2011] Johnson, M., Griffiths, D., Wang, M.: "Positioning Theory, Roles and the Design and Implementation of Learning Technology"; J.UCS (Journal of Universal Computer Science), 17, 9 (2011), 1329-1346

[Joyce 2012] Joyce, J.: "Raspberry Pi computer: Can it get kids into code?"; BBC News 29th February 2012, Retrieved April 10, 2012, from http://www.bbc.com/news/technology-17192823

[Khudairi, S. 2012] Khudairi, S. X.: "The Apache Software Foundation Announces Apache Rave as a Top-Level Project"; public announcement. Retrieved April 15, 2012, from http://mail-archives.apache.org/mod_mbox/www-announce/201203.mbox/%3C1332856837.39034.YahooMailNeo@web29706.mail.ird.yahoo.com%3E

[Koper and Tattersall 2005] Koper, R., Tattersall, C.: "An Introduction to Learning Design"; in Koper, R. and Tattersall, C. (Eds.), "Learning Design: modelling and implementing network-based education & training, Springer Verlag, Berlin (2005), 3-20

[Laanpere et al. 2006] Laanpere, M., Kikkas, K., Väljataga, T.: "iCamp Deliverable D2.2., iCamp Building Blocks, Version 1". Retrieved 15th April 2012 from http://www.icamp.eu/wp-content/uploads/2007/05/d22___icamp___building-blocks.pdf

[Laurillard 1993] Laurillard, D.: "Rethinking University Teaching: a framework for the effective use of educational technology"; Routledge, London (1993)

[Law and Chatterjee 2010] Law, E., Chatterjee, A.: "ROLE D1.6: Review on the Requirements Analysis Methods". Retrieved 15th April 2012 from http://www.role-project.eu/wp-content/uploads-role/2009/02/role-deliverable-1.5-1.61.pdf

[Manacorda 2010] Manacorda, E.: "Stallman: The iBad? A serious threat to individual freedom." The Daily Wired, 12th May 2010. Retrieved from http://daily.wired.it/news/tech/stallman-the-ibad-a-serious-threat-to-individual-freedom.html?page=2#content

[Marathe and Sundar 2011] Marathe, S., Sundar, S. S.: "What Drives Customization? Control Or Identity?" Proc. Annual Conference on Human factors in Computing Systems;. ACM, New York (2011), 781-790

[Muchmore 2011] Muchmore, M.: "Apple's Mac App Store: Hands On"; PC Magazine, 6th January 2011. Retrieved 15th April 2012 from http://www.pcmag.com/article/print/258979

[Neal 2011] Neal, D.: "The Design of Performance Pay in Education"; National Bureau of Economic Research Working Paper, NBER Working Paper No. 16710 (2011). Retrieved 15th April 2012 from http://www.nber.org/papers/w16710

[Norman 1988] Norman, D. A.: "The Psychology of Everyday Things"; Basic Books, (1988)

[Olivier and Liber 2001] Olivier, B., Liber, O.: "Lifelong Learning: The Need for Portable Personal Learning Environments and Supporting Interoperability Standards"; JISC CETIS working paper (2011). Retrieved 15th April 2012 from http://wiki.cetis.ac.uk/images/6/67/Olivierandliber2001.doc

[Pask 1975] Pask, G.: "Conversation, Cognition, and Learning"; Elsevier , New York, (1975)

[Sawyer 2011] Sawyer, R Keith.: "What makes good teachers great? The Artful Balance of Structure and Improvisation"; in Sawyer, R Keith (Ed.) "Structure and Improvisation in Creative Teaching", Cambridge University Press, Cambridge (2011) 1-24

[Schmitz et al. 2008] Schmitz, H.-C., Wolpers, M., Kirschenmann, U., Niemann, K.: "Contextualized Attention Metadata"; In C. Roda (Ed.), "Human Attention in Digital Environments" (p. 344). Cambridge, MA: Cambridge University Press.

[Severance et al. 2008] Severance, C., Hardin, J., Whyte, A.: "The coming functionality mash-up in Personal Learning Environments"; Interactive Learning Environments, 16, 1 (2008) 47-62

[Sire et al. 2009] Sire, S., Bogdanov, E., Palmér, M., Gillet, D.: "Towards Collaborative Portable Web Spaces"; Proc. 4th European Conference on Technology Enhanced Learning (EC-TEL) - Workshop on Mash-Up Personal Learning Environments (MUPPLE'09) Nice: CEUR Workshop Proceedings (2009), 1-12

[Success for All 2011] Success for All: "Success For All - Our Culture and Beliefs"; Retrieved April 13, 2012, from http://www.successforall.org/About-Us/Our-Culture-and-Beliefs/

[The Apache Software Foundation 2011] The Apache Software Foundation: "Apache Wookie (Incubating)"; (2011). Retrieved July 14, 2011, from http://incubator.apache.org/wookie/

[The Apache Software Foundation 2012] The Apache Software Foundation "Apache Rave"; (2012). Retrieved April 15, 2012, from http://rave.apache.org/

[Van Gundy 2011] Van Gundy, S.: "Why Connected Online Communities Will Drive the Future of Digital Content: An Introduction to Learning Resource Paradata"; Connected Educators, issued 9th November (2011). Retrieved from http://connectededucators.org/blog/why-connected-online-communities-will-drive-the-future-of-digital-content-an-introduction-to-learning-resource-paradata/

[Vuorikari, R 2011] Vuorikari, R.: "iTEC Deliverable D4.2 First Validation Report On Large-Scale Piloting (pp. 2-41)"; (2011). Retrieved from http://itec.eun.org/c/document_library/get_file?p_l_id=10307&folderId=37321&name=DLFE-1616.pdf

[W3C 2011] W3C. "Widget Packaging and XML Configuration"; W3C Recommendation of 27th September (2011). Retrieved December 11, 2011, from http://www.w3.org/TR/widgets/

[Wensink 2012] Wensink, C.: "App Stores and the Future of Software Development"; Techetron; Issue of 12th January (2012). Retrieved from http://www.techetron.com/1790/app-stores-and-the-future-of-software-development/

[Wilson et al. 2008] Wilson, S., Sharples, P., Griffiths, D.: "Distributing education services to personal and institutional systems using Widgets"; Proc. Mash-Up Personal Learning Environments (MUPPLE'08), Maastricht: CEUR Workshop Proceedings, (2008), 25-32

[Wilson et al. 2011] Wilson, S., Sharples, P., Griffiths, D., Popat, K.: "Augmenting the VLE using widget technologies"; International Journal of Technology Enhanced Learning, 3, 1 (2011), 4-20