

**USABILITY IN USER GENERATED LEARNING SPACES**

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER

FOR THE DEGREE OF MASTER OF SCIENCE

IN THE FACULTY OF ENGINEERING AND PHYSICAL SCIENCES

2011

By

UDOISANG, Blessing Sunday

School of Computer Science

Table of Contents

[List of Acronyms iv](#_Toc301700004)

[List of Tables iv](#_Toc301700005)

[List of Figures iv](#_Toc301700006)

[Abstract vi](#_Toc301700007)

[Chapter 1: Introduction 1](#_Toc301700008)

[1.1 Research Motivation and Questions 1](#_Toc301700009)

[1.2 Project Objectives 3](#_Toc301700010)

[1.3 Project Scope 3](#_Toc301700011)

[1.4 Thesis Structure 4](#_Toc301700012)

[Chapter 2: Background 6](#_Toc301700013)

[2.1 Basic Terminologies 6](#_Toc301700014)

[2.1.1 Learning 6](#_Toc301700015)

[2.1.2 Cognition 6](#_Toc301700016)

[2.1.3 Theory 6](#_Toc301700017)

[2.2 Learning Theories 7](#_Toc301700018)

[2.2.1 Behaviorist Theories 7](#_Toc301700019)

[2.2.2 Behaviorism in Learning 8](#_Toc301700020)

[2.2.3 Cognitivist Theories 8](#_Toc301700021)

[2.2.4 Cognitivism in Learning 8](#_Toc301700022)

[2.2.5 Constructivist Theories 8](#_Toc301700023)

[2.2.6 Constructivism in Learning 9](#_Toc301700024)

[2.2.7 Social Constructivism 9](#_Toc301700025)

[2.3 Comparison of Learning Theories 9](#_Toc301700026)

[2.4 User Generated Content, Social Learning and Constructionism 10](#_Toc301700027)

[2.4.1 User Generated Content and Social Learning Environments 10](#_Toc301700028)

[2.4.2 Constructionism 12](#_Toc301700029)

[2.5 Personal Development Planning 12](#_Toc301700030)

[2.5.1 Supporting Learning through PDP 14](#_Toc301700031)

[2.5.2 Potential Benefits of PDP 17](#_Toc301700032)

[2.6 Learning Space 18](#_Toc301700033)

[2.6.1 Trends in Learning Space Design 18](#_Toc301700034)

[2.6.2 Virtual Learning Spaces 20](#_Toc301700035)

[2.6.3 Usability of Learning Spaces 21](#_Toc301700036)

[2.7 Existing PLE Architecture to Support Virtual Learning Spaces 23](#_Toc301700037)

[2.8 Summary 23](#_Toc301700038)

[Chapter 3: System Design 24](#_Toc301700039)

[3.1 Initial considerations and Assumptions 24](#_Toc301700040)

[3.2 Problem Analysis 24](#_Toc301700041)

[3.2.1 The Basic Building Blocks 25](#_Toc301700042)

[3.3 Design Goals 27](#_Toc301700043)

[3.4 Architecture and Layout 28](#_Toc301700044)

[3.4.1 Media Spaces Architectural Overview 28](#_Toc301700045)

[3.4.2 The Space Client Layout 31](#_Toc301700046)

[3.4.3 The MVCS Architecture 32](#_Toc301700047)

[3.5 Inversion of Control and the SWIZ Framework 35](#_Toc301700048)

[3.5.1 Inversion of Control Strategy 35](#_Toc301700049)

[3.5.2 The SWIZ Framework 37](#_Toc301700050)

[3.6 SUMMARY (@TODO) 38](#_Toc301700051)

[4 Components Design 39](#_Toc301700052)

[4.1 Overview of the Personal Development Planning (PDP) Component 39](#_Toc301700053)

[4.2 Initial Design of the PDP Component 40](#_Toc301700054)

[4.2.1 Evaluation of the Initial PDP Component Design 41](#_Toc301700055)

[4.3 The Refined Design of the PDP Component 42](#_Toc301700056)

[4.3.1 Design Details for the Refined PDP Component 43](#_Toc301700057)

[4.3.2 Mapping the PDP Learning approach to the Refined PDP Component 43](#_Toc301700058)

[4.3.3 Improvements to the Refined Design 44](#_Toc301700059)

[4.5 The Audio Component 44](#_Toc301700060)

[4.6 The Twitter Search Component 45](#_Toc301700061)

[4.7 The Space Painter Component 46](#_Toc301700062)

[5. Implementation 46](#_Toc301700063)

[5.1 Implementation Platform, Languages and Tools 46](#_Toc301700064)

[5.1.1 Adobe Flex Framework and Flex SDK 47](#_Toc301700065)

[5.1.2 Adobe ActionScript 48](#_Toc301700066)

[5.1.3 Adobe Flash Builder 48](#_Toc301700067)

[5.1.4 How Flex Works 48](#_Toc301700068)

[5.1.5 Version Control System (VCS) 49](#_Toc301700069)

[5.2 Preparation for Implementation 49](#_Toc301700070)

[5.2.1 Understanding and Improving the Existing Implementation 49](#_Toc301700071)

[5.3 Adding New Components 50](#_Toc301700072)

[6 Evaluation and Analysis 51](#_Toc301700073)

[3.1 Project Description, Delivery Strategy & Evaluation 56](#_Toc301700074)

[3.1.1 Project Description 57](#_Toc301700075)

[3.2.2 Communication & Collaboration Tools 58](#_Toc301700076)

[3.1.2 Research Methodology 59](#_Toc301700077)

[3.1.3 Evaluation Plan 60](#_Toc301700078)

[3.2 Project Tools 61](#_Toc301700079)

[3.3 Project Management 61](#_Toc301700080)

[3.3.1 Agile Project Management 61](#_Toc301700081)

[3.3.2 Risk Management & Issue Resolution Plan 61](#_Toc301700082)

[Chapter 4: Conclusion 62](#_Toc301700083)

[List of References 63](#_Toc301700084)

[Appendix A: Project Plan Gantt Chart 66](#_Toc301700085)

# List of Acronyms

CPD Continual Professional Development

PLE Personal Learning Environment

PDP Personal Development Planning (Planner, Plan)

PSC Private Shareable Component

FLV

F4V

MP4

3GP

MP3

F4A

M4A

AAC

[@TODO Why not just list them in the Appendix of Media Formats – Appendix…..]

# List of Tables

[Table 1 - Sample PDP Schedule 16](#_Toc300544133)

[Table 2 - RTMP Connections and Supporting Media Servers 31](#_Toc300544134)

[Table 3 - Some objects managed by Swiz Framework 38](#_Toc300544135)

# List of Figures

[Figure 1 - Behaviourist Model 7](#_Toc300544117)

[Figure 2 - Continuum of Learning Theories 10](#_Toc300544118)

[Figure 3 - The PDP Learning Process 15](#_Toc300544119)

[Figure 4 - Linear Arrangement vs. Collaborative Arrangement 19](#_Toc300544120)

[Figure 5 - Attributes of System Acceptability 21](#_Toc300544121)

[Figure 6 - Overall Architecture of MPLE 23](#_Toc300544122)

[Figure 7 - Learning Phases 25](#_Toc300544123)

[Figure 8 - Whiteboard, Components and Edges 26](#_Toc300544124)

[Figure 9 - Sketch and Actual Implementation of the Toolbox 27](#_Toc300544125)

[Figure 10 - Media Space Architecture 29](#_Toc300544126)

[Figure 11 - A screen shot of the existing space client application 30](#_Toc300544127)

[Figure 12 - Space Client Layout 31](#_Toc300544128)

[Figure 13 - The MVCS Architecture 33](#_Toc300544129)

[Figure 14 - Objects instantiation in a traditional application 36](#_Toc300544130)

[Figure 15 - Objects instantiation in an IOC application 36](#_Toc300544131)

[Figure 16 - Research Methodology Flow 47](#_Toc300544132)

# Abstract

The focus of this work is to help enable learning in shared, collaborative multi-media learning spaces, by improving the facilities of, and the user interface to, the Manchester Personal Learning Environment (the PLE). The PLE and its learning spaces are designed to support learning as a social process: People learn with each other and from each other. The effective design of learning spaces can enhance the way learning takes place and consequently the outcome. Established theories of learning give rise to strategies for learning facilitation. Behaviourism, cognitivism, social constructivism and Papert’s constructionism are surveyed as an initial step in seeking informants for the work proposed here. Particularly, the view is taken that for present day learners who in part exist in a Web ecosystem, learning is all about knowledge creation and discovery, sharing and reusing content. Such learners prefer active and participatory learning. This dissertation focuses on improving usability in the PLE’s virtual learning spaces to support this kind of learning. Process aspects of implementing improved learning spaces include assuring both usability and pedagogic usability, through a process of ongoing formative evaluation. Some of these process aspects are discussed here. In a pedagogical context, it is important to measure the learner’s satisfaction, not only in the product but also with regards to learning goals, including metacognitve processes in setting, refining and realizing those learning goals. Thus the central question of the work proposed here is what principles of educational theory will improve the usability of user generated learning spaces, and how might these be realized in a practical way in the Manchester PLE’s virtual learning spaces.

# Chapter 1: Introduction

Learning spaces “encompass the full range of places in which learning occurs, from real to virtual; classroom to chat room” (Brown, 2005). The effective design of physical and virtual learning spaces can enhance the way learning takes place and consequently the outcome. With the advancements in technology leading to convergence of platforms and proliferation of high capacity mobile devices, learners are increasingly shifting their preferences for learning environments from the physical to the virtual. For any interactive system that implements virtual learning spaces, usability is a key determinant in uptake. Usability is defined by the International Organization for Standardization (ISO 9241) as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use".

Designing for learning requires an understanding of how learners construct knowledge. Relevant learning theories need to be explored and applied within the learners' context to capture the learning activity in a way "natural" to the learners. Established theories such as social constructivism, constructionism, self-directed learning, communities of practice, etc engender strategies for facilitating learning. However, the context of learning keeps evolving. It is therefore important to re-examine these theories with a view to adapting them to the prevailing contexts in which learning occurs. This becomes more challenging when viewed from the virtual learning perspective. Interestingly, information and communication technology (ICT) provides a lot of tools that can help us realise our objectives. These ICT tools have been applied in various ways but the question is how usable are they?

This project is concerned with improving usability in user generated learning spaces. User generated learning spaces are “(learning) spaces which are populated with content by one or more learners” (van Harmelen, 2011). This content according to Harmelen “might be learner generated, or co-opted from elsewhere and used unchanged, or modified, or mixed with other content”. Whatever the case might be, the purpose remains the same, “to help the user(s) learn about a particular topic area, or fulfill one or more learning goals” (Harmelen, 2011). The project is motivated by the need to support learning among diverse categories of learners. Doing this raises vital questions which are clearly discussed in the next section.

## 1.1 Research Motivation and Questions

Learning is a continuous process; one does not stop learning (Dechant, 1991, Doukidis, Mylonopoulos & Pouloudi, 2004, Bergan, 2007). However, there are stages in our life when learning is formalized such that one is more bounded by rules (rather than choice) as to what to learn, when to learn it, how to show that what is required has been learnt and a reward (or otherwise) given for successfully showing it. Apart from this kind of formal system which majorly happens in academic institutions, one can also learn on his own. This kind of learning is referred to as personal learning, personal study, personal development, etc. As an example, a graduate lawyer who majored generally in oil and gas law might become particularly interested in oil financing laws. He does not have to return to law school to get a degree for such knowledge; rather, he takes up a personal study and gets acquainted with the practice of oil financing laws. This kind of scenario is more formally emobodied in what is known as Continuing Professional Development (CPD) (Rughani, Franklin & Dixon, 2003). However, it must not be formal to produce results if properly supported. The challenge is that such kind of personal learning is usually not very effective in the absence of a formal learning framework. This is because there is no external authority mandating the learner to carry out such learning effectively. Nonetheless, the serious learner, for personal reasons will proceed to carry out his personal studies and with the availability of a proper supporting framework, will do it effectively thus reaping the expected outcomes (JISC-CRA, Undated, @TODO Try and Find Date). Such supporting framework can be in form of formal/informal processes, coaching, mentoring, group studies and of course technology. This is where Personal Learning Environments (PLE) comes to play. A personal learning environment can support the learner to achieve his learning goals in many ways among which are:

* It can act as a repository of knowledge for the learner
* It can enable the learner to store and bookmark learning resources
* It can provide a social layer where the learner can collaboratively share knowledge with other learners
* It can help the learner monitor progress of his learning
* It can help the learner reflect upon his learning and evaluate the outcome of such learning.

The motivation behind this research project is to support the learner in the ways listed above and in many more ways to achieve his learning goals. In order achieve this, we focus on usability in user generated learning spaces which has already been briefly discussed in the introduction and is further discussed in section (@TODO Which section). The project focuses on how to use interactive virtual learning spaces to support the user in pursuing his personal development in the presence or absence of a formal learning framework. Although there are various interactive multimedia tools in use today for similar purposes, this project is unique in that it does not focus on interactive multimedia technology in isolation; rather, it attempts to lay solid psychological and pedagogical foundations for the tools, processes and methods provided. Consequently, the research seeks to answer the following vital questions:

* Can one use interactive multimedia spaces to actively support a learner in achieving his/her learning goals?
* Can one develop a usable multimedia learning space that will run smoothly despite the limitations of the internet?
* Can a learner develop a personal development plan (PDP) in the multimedia learning space and use it to pursue a learning goal successfully?
* Can a personal development plan so created by a learner be converted to a learning artefact which can be used and/or re-used by other learners to achieve similar outcome(s) as the initial creator/learner?
* Can a learner share his personal development plans and pursue his learning goals collaboratively?
* How can the learner use Private Shareable Components (PSCs) in pursuing his learning goals?
* Can a learner recover components that have been deleted from the multimedia learning space intentionally or mistakenly?
* Can the usability of such multimedia learning space be evaluated by users?
* Is the creation of personal development plans by the learner and the transformation of these plans into explicit learning resources a useful way to support learning?

[@TODO possibly re-order this list of questions in a more logically flowing manner]

These questions are the motivating factors for this research and they were all answered successfully as can be seen in the concluding section (@TODO Where?). The objectives of the project were set so as to ensure that the research questions raised are properly answered. These are discussed in the next section.

## 1.2 Project Objectives

The target goal of the project is to investigate usability in user generated learning spaces. The findings will then be applied in improving the usability of virtual learning spaces in a Personal Learning Environment (PLE). As part of this I will be working on improving the user interface of the multimedia learning space in the Manchester PLE; performing development in FLEX® 4 and complementary technologies.

In order to ensure the goal of the project is achieved, it has been decomposed into broad objectives that will act as guiding posts to achieving the main goal. They are as follows:

* To understand the application of relevant learning theories to the design of learning spaces
* To improve active construction of knowledge by users in virtual learning spaces
* To improve self-directed as well as collaborative learning in virtual learning spaces
* To Investigate (by user evaluation) the usability of current learning spaces in the PLE
* To develop improved user interfaces (that are being tested by users) for the current learning spaces in the PLE
* To assess and improve by user testing, the usability of the developed user interfaces

This project covers a broad spectrum because it involves designing for learning. However, given the time available for the project, a limited scope that ensures the target goal is achieved successfully was defined.

## 1.3 Project Scope

The project scope includes investigating and improving the usability of virtual learning spaces in the Manchester PLE. Usability evaluation for pedagogical applications comprises two parts namely: technical/functional usability evaluation and pedagogical usability evaluation. This involves the use of testing and formative evaluation in iterative process of design & implementation. A careful comparison of various frameworks for usability testing was made to select the most appropriate one that can be applied. Cooperative evaluation was selected for its inherent benefits. This is further reported in section x.x (@TODO Section What?). Web usability is also within the scope of the project. This is because virtual learning spaces are mostly implemented on web pages. Consequentially principles of web usability design were incorporated in the design and implementation of the improved learning space. Finally, evaluation of the improved learning interface was carried out periodically using formative evaluation with users and reported accordingly in section x.x (@TODO Section What?). Some innovative features of the improved learning space are as follows:

* Personal development planning (PDP) which enables the learner to take control of his learning. This is discussed in details in section (@TODO Which section?).
* The ability to convert static PDPs to reusable ‘Learning Artefacts’.
* The ability to track and highlight changes in the space both in private and collaborative mode
* The ability to undo actions including in collaborative mode, considering the fact that some other actions might have been performed based on the action to be undone.
* The transfer of desirable classroom and informal space characteristics into the virtual learning space such as ambience, immersion principle, attention and motivation theories, layout re-configurability, knowledge discovery, etc.

The scope defined above was broadly covered and the target goal of the project was achieved. The rest of this report gives a background of the project as well as design and implementation details. The structure of the report is discussed next.

## 1.4 Thesis Structure

The project has been introduced; the motivation for it given as well as the major questions that motivated the research. The goal and scope has also been clearly stated. The remaining part of this thesis describes how the goal was achieved and is structured as follows:

**Chapter 2 – Background**

This section discusses relevant background materials with the aim of situating the project into a wider research theme. Relevant learning theories and how they apply to learning are discussed. Learning theories as we shall see do not only apply to learning but also to the design of learning spaces. This is discussed here also. Current trends in learning space design are examined and then usability in learning spaces design with focus on usable virtual learning spaces is discussed. Finally, a firm theoretical foundation is laid in support of user generated content in learning spaces as well as personal development plans.

**Chapter 3 – System Design 1**

Chapters 3 and 4 cover the design phase of the project. Chapter 3 in particular only covers a part of the system design and is largely devoted to the existing implementation of the multimedia learning spaces in the Manchester PLE. Subsequent chapters cover the design of the new features for the PLE. In this chapter, basic assumptions, problem analysis and design goals are discussed first. An architectural analysis of the existing and proposed system is made. Frameworks, micro-architectures and design pattern (both existing and proposed) involved are also discussed.

**Chapter 4 – System Design 2**

Chapter 4 gives a clear description of design considerations for the main features to be added to the PLE. These include the Twitter Box, Space Painter, Personal Development Planner (PDP), etc. The design of each new component or subcomponent is clearly described.

**Chapter 5 – Implementation**

In order to achieve the kind of rich user experience expected of this kind of system, Rich Internet Application (RIA) platforms were chosen. This section sets out with a proper introduction of the tools of the trade; giving justification for the choice of Adobe Flex ® and complementary technologies. In order to ease entry into the existing code base, some form of code reading and optimising was carried out. This involved debugging certain parts of the existing implementation and also improving the user interaction of existing multimedia components. Having become comfortable with the existing code base, further development was then carried out to implement additional features. This is fully reported in this section also.

**Chapter 6 – Evaluation and Critical Analysis**

This section captures the details behind the evaluation methodology adopted for this project. The strategy as well as the results is reported accordingly. The chapter ends with identification and analysis of the limitations of the system.

**Chapter 7 – Reflection and Future Work**

In this section a brief reflection on the project is done. The research questions are re-visited and a summary of achievements is made. Additionally, areas for future development are identified.

**Chapter 8 – Conclusion**

A run through of the research is done and the project is brought to a logical conclusion.

**Appendices**

**Appendix A:** Contains ….

**Appendix B:** Contains….

# Chapter 2: Background

This chapter explains the background behind this project. Significant learning theories are introduced which generally specify ideal ways to learn and teach. This is followed by discussion on Virtual Learning Spaces. We then turn to the importance of usability in virtual learning spaces and finally conclude with reviewing related work.

## 2.1 Basic Terminologies

### 2.1.1 Learning

Being a complex process, it is not easy to define learning. According to Domjan & Burkhard (1993), “Learning is such a common experience that we hardly ever reflect on exactly what we mean when we say that something has been learnt”. They went ahead to confirm that “a universally acceptable definition for learning does not exist” (Domjan & Burkhard, 1993). However, in the following definition, they attempted to capture many critical aspects of the concept of learning:

“Learning is an enduring change in the mechanisms of behaviour involving specific stimuli and/or responses that result from prior experience with those stimuli and responses”

In this definition, Domjan & Burkhard view learning from a behavioural perspective which is usually inadequate in defining learning when considered in isolation. The following definition (commonly used but source cannot be traced yet) provide the missing link by defining learning as “a process that brings together cognitive, emotional, and environmental influences and experiences for acquiring, enhancing, or making changes in one's knowledge, skills, values, and world views”. This definition attempts to capture the process as well as the product. A noteworthy fact in the definition is that learning is a product of the interplay between the cognitive, the emotional and the environmental. However, the environment can affect both emotion and cognition, positively or otherwise. It therefore becomes a very important factor in learning.

### 2.1.2 Cognition

Cognition has to do with “how our brain works or how our mind works” (Leonard, Noh, & Orey, 2007). Cognition is the psychological result of perception, learning and reasoning. To put it in simpler terms, it is the act of knowing. Cognition can also refer to the process of knowing. So in my own words, cognition can be a “process” as well as a “product”. To clarify any ambiguity, when we talk about how the brain works, we are not making a biological reference to the brain, “most cognitive theories are more conceptual and therefore it might be more accurate to talk about how the mind works rather than a biological reference to the brain” (Leonard, Noh, and Orey, 2007).

### 2.1.3 Theory

The term theory is a frequently used word in everyday vocabulary. However, the meaning of a theory in science is not the same as the colloquial use of the word. Marx (1970) defines a theory as “a provisional explanatory proposition, or set of propositions, concerning some natural phenomena”. Leonard, Noh, and Orey (2007) share this “explanatory” perspective. According to them, a theory is “a hypothesis that describes, speculates, or defines a relationship between a set of facts or phenomena through a body of principles, policies, beliefs, or assumptions”. It follows from both definitions that there exists a subtle tone of assumption in every theory.

## 2.2 Learning Theories

A Learning theory attempts to help us understand the complex process of learning by describing how people (and animals) learn. Learning theories have two chief values:

* Providing a vocabulary and a conceptual framework for interpreting the examples of learning we observe.
* Suggesting where to look for solutions to practical problems.

Interestingly, the theories do not provide solutions to practical problems. However, it should be noted that they do direct our attention to important variables that are crucial in finding solutions. The meaning of “Learning” has been discussed in a previous section above. Learning theories are generally categorized under three philosophical frameworks namely:

* Behaviourism
* Cognitivism
* Constructivism

### 2.2.1 Behaviorist Theories

Behaviorism is a learning theory based on the idea that all behaviors are acquired through conditioning which occurs through interaction with the environment. J. B Watson, widely regarded as the father of Behaviorism, defined learning as “a sequence of stimulus and response actions in observable cause and effect relationships” (Chowdhury, 2006). Thus behaviourism assumes that the learner is essentially passive, responding to environmental stimuli. According to LTKB (2011), “the learner starts off as a clean slate (i.e. *tabula rasa*) and behavior is shaped through positive reinforcement or negative reinforcement.” Positive indicates the application of a stimulus while negative indicates withholding of a stimulus, thus learning is observable by the “change in the behavior of the learner” in response to the stimuli (LTKB, 2011) as shown in figure 1 below. There are basically two kinds of conditioning in Behaviorism namely classical conditioning and operant conditioning.

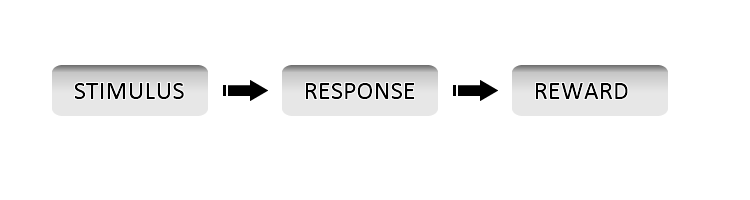


Figure 1 - Behaviourist Model

Behaviorism has been applied in the fields of psychology and medicine but our interest lies in its application in learning improvement. Educational approaches such as applied behaviour analysis, curriculum based measurement, and direct instruction have emerged from this model (Kim & Axelrod, 2005). The original theory of behaviorism is now more commonly referred to as “classical behaviorism”. New lines of thought have been extracted from classical behaviorism thus giving rise to Neo-Behaviourism (second Generation) and Social-Behaviorism (Third Generation). Of these two, Social Behaviorism focuses more on learning. It considers learning as a relatively stable behavior modification arising from experience.

### 2.2.2 Behaviorism in Learning

In Behaviourist approaches, learning is centred on the teacher. The teacher is given the role of transferring his knowledge to the learner which is confirmed done by observing a relative permanent change in the behavior of the learner. This approached is marked by reinforced and programmed learning (LTKB, 2011). According to Standridge (2002), “Behaviorist techniques have long been employed in education to promote behavior that is desirable and discourage that which is not”. Below is a summary of some features of a behaviourist learning model:

* Learning is done in small, concrete, progressively sequenced tasks
* Learning is marked by repetition in order to increase retention and speed of learning.
* Consistent use of reinforcements during the teaching-learning process. For instance, with verbal acts such as congratulatory remarks and non verbal reinforcements such as awards.

### 2.2.3 Cognitivist Theories

Cognitivism as a learning theory looks beyond behaviour to explain “brain” based learning. In other words Cognitivism attempts to improve learning by considering how the human memory works. Cognitivism shares a similarity with behaviourism on the basis that both view knowledge as “given” and “absolute” (LTKB, 2011). However, Cognitivism is based on the assumption that human beings are logical beings and thus make choices that are most sensible to them. Pure cognitive theory largely rejects behaviourism on the basis that behaviorism reduces complex human behavior to simple cause and effect (Fritscher, 2011). However, current trends in past decades have been towards merging the two into a comprehensive “cognitive-behavioural theory” (Fritscher, 2011).

### 2.2.4 Cognitivism in Learning

Cognitivism approaches learning from a learner-centred perspective. From this perspective, learners need to develop deeper understandings, not just produce the right behaviors (Wortham, 2003). Since these deeper understandings cannot be imposed on learners, they must construct their own mental models with sufficient guide from the teacher. Cognitivism views learning as a change in the learner’s understanding, hence the focus is on elaboration. The teacher plays the role of a coach or a facilitator. As a facilitator, he has to provide clues and teach mnemonic strategies (Fortin & Rousseau, 1989), to introduce context. As a coach, he has to constantly evaluate the learner’s knowledge to keep the learner as active as possible. Tardif (1992) lists some basic principles that characterize the cognitive learning approach as follows:

* Learning is an active and constructive
* Prior knowledge a crucial factor in learning and believes that knowledge is essentially cumulative.
* Learning permits a link between the new pieces of information and the information already in memory.

### 2.2.5 Constructivist Theories

Constructivism as a learning theory views knowledge as a “constructed” entity (LTKB, 2011). In contrast to the view that knowledge is absolute and given, constructivism asserts that knowledge is constructed by reflecting on our experiences thus fabricating our own understanding of the world we live in (LTKB, 2011). According to the constructivism paradigm, human learning is an active attempt to construct knowledge based on previous knowledge and the present context. Therefore, every person will construct their own unique set of knowledge. In other words, no two people will start with exactly the same knowledge base, and no two people will construct exactly the same knowledge structures from given experiences or information.

### 2.2.6 Constructivism in Learning

Constructivism approaches learning from a learner-centred perspective also. However, learning to the constructivist is “discovery and construction of meaning”. In the constructivist view, knowledge cannot be poured in, from one person to another. It holds also, that knowledge does not become part of the learner after memorisation of external objective information but is continuously built as the learner interacts with the outside world thus producing his own interpretations about it. According to DeVries et al. (2002), in most pedagogy based on constructivism, “the teacher's role is not only to observe and assess but to also engage with the students while they are completing activities, wondering aloud and posing questions to the students for promotion of reasoning”. This promotes learning by experimentation and exploration, not by being told what will happen. The constructivist pedagogy involves the following characteristics (Richardson, 2003):

* Student-centredness, evident in attention to the individual and respect for students' backgrounds
* Facilitation of group dialogue that explores an element of the domain with the purpose of leading to the creation and shared understanding of a topic
* Provision of opportunities for students to determine, challenge, change or add to existing beliefs and understandings through engagement in tasks structured for this purpose

### 2.2.7 Social Constructivism

Social constructivism proceeds from Vygotsky’s social development theory. Social development theory argues that social interaction precedes development; consciousness and cognition are the end product of socialization and social behavior (LTKB, 2011a). Vygotsky focused on the connections between people and the socio-cultural context in which they act and interact in shared experiences (Crawford, 1996). It follows from the ideas of Vygotsky and others that learning is a social process. This is why the environment within which learning occurs plays a very important role in social constructivism.

Social constructivism emphasizes the benefits of collaborative learning (Fruchter and Emery, 1999). The role of the educator in this context is to provide what is known as “scaffolding”. Scaffolding refers to guidelines and hints which help the learner build strong, complex and relevant ideas (Vygotsky and Cole, 1978). The learner progressively removes this scaffolding and tends towards self-directed learning replacing scaffolding with his own ideas and plans.

## 2.3 Comparison of Learning Theories

Learning theories are based on different assumptions and focus on different perspectives in explaining learning. Nevertheless, they bear close relationship to one another. Learning styles and behaviours may be viewed as existing on a continuum as shown in figure 2 below. While it may be said that most educational models in use today combine concepts that are mostly drawn from cognitivism and constructivism, that does not mean that the behaviourist theories are not still applicable. For example according to Perraudeau (1996 cited in Ughade et al, 2007), “to develop high intellectual level abilities such as analysis or problem resolution, the teacher will tend to privilege constructivist and cognitivist approaches, whereas for information memorization, a behaviorist approach can be better”.

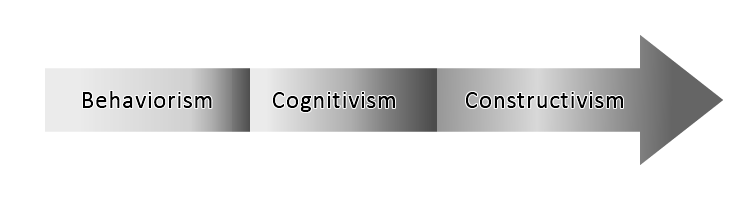


Figure 2 - Continuum of Learning Theories

## 2.4 User Generated Content, Social Learning and Constructionism

The theories of learning discussed thus far attempts to explain how learning takes place. Having understood how it does, we can then improve the learning process in order to improve the product. In this section, the concept of user generated content is discussed and situated within the wider context of social learning environments. Papert’s constructionism is then introduced as a way to ensure effectiveness in social learning through user generated content.

### 2.4.1 User Generated Content and Social Learning Environments

User generated content (UGC) is “one of the main features of this so-called participative web” (Wunsch-Vincent & Vickery, 2007 cited in Clever, Kirchner, Schray, & Schulte, 2009). The term User generated content refers to data, information, or media that is contributed by ‘regular people’ to the knowledge and information space, usually the web (Krumm, Davies, & Narayanaswami, 2008; Clever, Kirchner, Schray, & Schulte, 2009). The term ‘regular people’ implies that the contributors need not be experts in the field they are contributing to. They do not need to be expert journalists either; rather they are increasingly being referred to as “the amateur creators” (Wunsch-Vincent & Vickery, 2007). Examples of UGC on the web include ratings and reviews (such as restaurant ratings, Amazon reviews, Google buzz, Rotten Tomatoes, etc), wikis, blogs, forums, videos, online radio, classifieds, shared files, group-based aggregation (e.g. del.icio.us, podcasting), etc.

Although there is no commonly agreed definition of user generated content despite its frequent use (Wunsch-Vincent & Vickery, 2007; Clever, Kirchner, Schray, & Schulte, 2009), the Organisation for Economic Co-operation and Development (OECD) has proposed three main characteristics. These characteristics give a more solid understanding of the concept and they are as follows (Wunsch-Vincent & Vickery, 2007; Clever, Kirchner, Schray, & Schulte, 2009):

1. **Publication Requirement**: UGC has to be published in some context (e.g. on a website or on a social networking site).
2. **Creative Effort**: A “certain amount of creative effort (has to be) put into creating the work” or adapting existing works to construct a new one. This implies that the users must add their own value to the work.
3. **Creation outside of professional routines and practices**: Typically UGC is created without the “expectation of profit or remuneration”. The OECD identifies the desire for fame, notoriety or prestige and the desire for self expression as motivating factors in the absence of remuneration.

The third characteristic given by the OECD highlights an important point that provides a common unifying factor for UGC creators and that is ‘the desire for self expression’. These users enjoy being creative and entertaining others (Clever, Kirchner, Schray, & Schulte, 2009). They seek the opportunity to share knowledge, experiences and document their lives. These shared values usually bring them together to become part of online communities and collaborative projects. Thus Social Learning Environments (SLEs) are directly or indirectly formed. A Social learning environment provides a space for individuals to collaboratively work and learn together formally or otherwise. Jane Hart, a Social Business Consultant, and Founder of the Centre for Learning & Performance Technologies, suggests a definition for social learning environments that explains almost every facet of the concept. According to her, a social learning environment is:

“a place where individuals and groups of individuals can come together and co-create content, share knowledge and experiences, and learn from one another to improve their personal and professional productivity; and is also a place that can be used both to extend formal content-based e-learning to provide social interaction with the learners and tutors, as well as to underpin informal learning and working in the organization” (Hart, 2009).

Although Hart’s definition talks about the activities in a SLE and the benefits (i.e. learning from each other), it is silent however on the social processes involved as wells as the tools that are required to perform the listed activities. With respect to the required tools, Hart list 100 tools for learning in another article (Hart, 2009b). These include social networking tools, tagging content tools, social bookmarking tools, file sharing tools, blogging tools, etc. The figure below gives a visual representation of the tools necessary for constructing a social learning environment.

C:\Documents and Settings\iXeonite\My Documents\My Pictures\proj-Social-Learning-Elements2-TIFF.tif

Figure 3 - Tools for constructing a social learning environment

(Adapted from Hart’s Top 100 tools for learning of 2009 and Kadle’s elements for constructing SLEs)

These tools give learners the power to unleash their creativity in various ways. Many learners have used them to create and share different forms of media though sometimes just for the fun of it or to help other learners. However, creating tangible artefacts for the purpose of learning is the focus of Papert’s constructionism. The next section discusses how constructionism can be applied in social learning environments.

### 2.4.2 Constructionism

Constructionism is based on the constructivist learning theory (see [section 2.2.5](#_2.2.5_Constructivist_Theories)). The constructivist theory suggests that learners construct mental models to understand the world around them. Constructionism takes this further by encouraging the learner to also build a **tangible physical artefact** in the real world to represent the knowledge acquired. Constructionism holds that learning is most effective when part of the learning activity involves the construction of a meaningful product (Papert et al, 1986) hence the phrase “learning-by-making” suggested by Papert and Harel (Papert & Harel, 1991). According to them, learning involves "building knowledge structures" and “this happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe” (Papert & Harel, 1991). In the context of the web, it can be said that constructionism encourages user generated content in the context of learning. In other words, users can generate and publish content for the purpose of learning new ideas or enhancing their skills.

However, Learning is a social process as have already been established in [section 2.2.7](#_2.2.7_Social_Constructivism). Also, in a social learning environment, individuals co-create content in order to share knowledge, experiences, and learn from one another (Hart, 2009). Thus, if I take Papert’s phrase “learning-by-making” and combine it with the idea of “learning-by-sharing” as suggested by social learning environments, I end up with another catchy phrase; “learning-by-making-for-sharing”. This phrase however suggests two possible scenarios:

1. A learner constructs a public artefact for sharing with a community of learners.
2. A community of learners come together to construct a public artefact based on their collective knowledge of a particular subject.

Irrespective of the scenario, a social learning environment can provide the required platform for it to occur. A very useful artefact that can be created to guide the learning process is the personal development plan. Personal development planning deals with how such a plan can be created and used. An interesting idea will be to apply constructionism to personal development planning by converting personal development plans to knowledge artefacts which can be shared and re-used to achieve a learning goal. The next section discusses the concept of personal development planning.

## 2.5 Personal Development Planning

The Dearing Enquiry (National Committee of Inquiry in Higher Education; 1997) recommended the introduction of ‘progress files’ by universities as a means by which students monitor, build and reflect upon their personal development (Cottrell, 2010; NCIHE, 1997). In 2001, the Quality Assurance Agency for Higher Education (QAA) published the Guidelines for HE Progress Files and now expects all universities to ensure that students undertake personal planning using the progress files (Gosling, 2003; Quinton & Smallbone, 2008; Cottrell, 2010).

Progress files consist of three elements (Cottrell, 2010):

1. personal development processes
2. student records that guide personal reflection and planning
3. the formal university transcript

The processes of Personal Development Planning (PDP) according to Cottrell (2010) are at the heart of the progress files initiative and are the most important aspect of it. Personal Development Planning is defined as

“A structured and supported process undertaken by an individual to reflect upon their own learning, performance and/or achievement and to plan for their personal, educational and career development” (QAA, 2001; Ward, 2001; THEA, 2011).

Being a structured process implies it has to be organized and where possible integrated into a broader framework. It also implies that specific outcomes be listed for the process. This is probably the reason why the information recorded during the process is usually used to prepare curriculum vitae (CV) for the individual. The definition also talks about the process being supported. According to O’Connell, “early experiences of introducing PADPs at the University of Manchester had shown that students are unwilling to engage in ‘standalone’ schemes” (O’Connell, 2001). It is therefore important that the processed be structured and supported if the expected outcomes are to be met.

Reflecting upon the learning and making plans for personal, educational and career development is the expected outcome of the process according to the definition. Gosling reveals the underlying activities in the process and also states a way of measuring the success of the process. According to him,

“PDP refers to a set of activities which engages students in reflecting on their learning and personal goals, creating personal records, planning and monitoring progress towards the achievement of personal objectives and which is intended to improve the capacity of students to articulate their learning goals for themselves and to communicate the outcomes of their learning to others, for example to academic staff and employers” (Gosling, 2003).

Creating personal records is central to the whole process because reflecting and planning is best done using available records of progress. Additionally, progress cannot be effectively monitored in the absence of records of activities. The success of the process according to Gosling can be determined if the students can clearly:

* Articulate their learning goals for themselves
* Communicate the outcomes of their learning to others e.g. academic staff and employers

Communicating the outcomes of their learning to academic staff is usually done by documenting and submitting the progress files to an academic tutor. A brief discussion may follow during which the tutor reviews progress with the student. To the employer on the other hand, a curriculum vitae is probably the best way to communicate such outcomes. It can be observed that in both instances, communicating the outcomes of the process involves creating an artefact, the principle of constructionism. This idea is emphasized by Rughani, Franklin, & Dixon (2003) who define PDP as

“a process by which we identify our educational needs, set ourselves some objectives in relation to these, undertake our educational activities and produce evidence that we have learned something useful” (Rughani, Franklin, & Dixon, 2003).

Their definition also clarifies a fact that is implicit in the other definition, ‘identifying the educational needs’ of the learner. Several other definitions in the literature of personal learning all agree to the ideas presented so far which can be summarised as follows:

* PDP is a structured process
* PDP involves identifying one’s educational needs
* PDP requires reflection upon one’s learning, performance and achievements
* PDP requires planning and monitoring progress
* PDP requires some form of recording
* PDP should be supported
* PDP is personal

In the context of social and collaborative learning, the last point can be an obstacle to sharing PDPs. However a closer look at the literature doesn’t imply that PDPs are not shareable, rather it implies that subtle modification may be necessary to adapt one learner’s PDP to another learner. Sharing PDPs has benefits that cannot be ignored. First of all, it encourages reuse which is important in learning. Also, one learner’s PDP can act as a pointer to the right direction for another learner who is interested in the same subject. Furthermore, shared PDPs can improve the PDP itself since ideas from different learner can be used to modify and improve the PDP. Lastly, if a PDP can be converted to a knowledge artefact, that is, a PDP with a collection of resources for achieving the goals intended, then such knowledge artefact can be preserved, shared and reused as teaching and learning aids in that field. This last point is one of the design goals of this research project. It involves using multimedia components to convert a PDP into a knowledge artefact that can be shared among learners distributed in time and space.

The main purpose of personal development planning is to support learning. The ‘planning’ word is not enough to achieve the expected outcomes, there has to be an ‘executing’ part also. The next section explains how learning is supported using PDPs.

### 2.5.1 Supporting Learning through PDP

The reflective and planning process in PDP can be framed around the following questions (Jackson, 2001):

* Where have I been? Retrospective reflection
* Where am I now? Reflection on current situation
* Where do I want to get to? Review of opportunities and identification of personal goals or objectives
* How do I get there? Review of possibilities and decisions on the best way of achieving goals/objectives
* How will I know I've got there? Strategy for setting targets and reviewing progress

These questions can help the learner reflect on what has been learnt prior to the time; what is to be learnt; what is required to facilitate such learning; how to go about the learning process and lastly how to ascertain that the learning goals have been met. The questions raised above by Jackson covers a more generalised approach to supporting learning through PDP. However in an academic context, these questions can be re-framed as follows (Jackson, 2001):

* What have I learnt or done? Retrospective reflection
* What do I need to learn or do to improve myself? Reflection on current situation
* How do I do it? Review of opportunities and identification of personal goals or objectives
* How will I know I've done it? Strategy for setting targets and reviewing progress

Irrespective of the context, the following steps can be identified (Rughani, Franklin, & Dixon, 2003; LLC-UoM, 2010):

* Auditing
* Planning
* Executing
* Reflecting/Evaluating
* Recording

Recording occupies a special place in the listed steps because each step involves some form of recording to be effective. The figure below captures the PDP learning process:

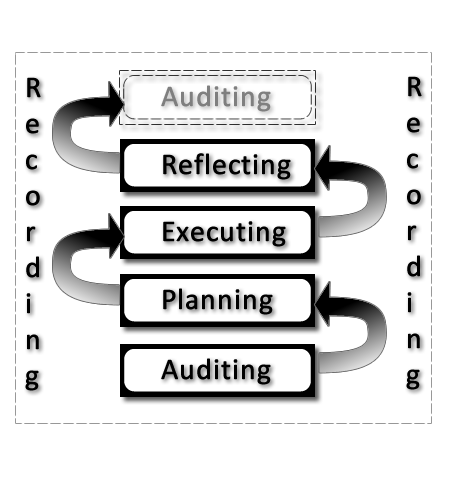


Figure 4 - The PDP Learning Process

The figure depicts the fact that the PDP learning process is ‘spiral’ rather than ‘cyclic’. This means that learner begins with auditing and then proceeds to planning, executing, and reflecting on his learning after which the process returns to auditing again. However, we do not view it as a circular process rather it is viewed as a rising spiral indicating the progress the learner is making. In the words of Rughani, Franklin, & Dixon (2003), spiral is used rather than cycle “to emphasise that with every step, we are further on in our educational development than we were at the same time the preceding year”.

In an academic environment, a Personal Development Plan is usually a collection of documents to guide the students as well as help them document the process. Each document is designed to be completed at a certain time during the academic year (LLC-UoM, 2010). A good example is the PDP Schedule of the Language and Linguistics centre, University of Manchester (LLC-UoM). The schedule consists of (LLC-UoM, 2010):

* **Skills Audits**: covering topics such as time-management, IT and computer skills, handling information, essay writing, etc.
* **Action Plans**: helping the student to improve these skills
* **Reflection and Review**: recording experiences, and development of academic and transferable skills

(@TODO Put Samples of Skills Audit, Action Plan, etc in Appendices)

At the beginning of each academic year, students are encouraged to download the PDP documents and participate in the process. Academic Advisor meetings are also scheduled for the students where they can discuss their progress, receive guidance and possibly ask questions. A sample schedule for a given semester may look like the table below:

|  |  |
| --- | --- |
| **Semester 1** |  |
| **Week 0**  **(Welcome Week) Thursday, 2pm** | You will meet your Academic Advisor and some of your fellow students to chat informally about your experiences and expectations of the University so far and to receive a lot of practical information. |
| **Week 5** | An opportunity to discuss your progress so far and any problems arising from your first months at university. |
| **Week 10 or 11** | An opportunity to discuss questions arising from your Semester 1 PDP ‘Reflection and Review’ and to share experiences with fellow students. |
| **Semester 2** |  |
| **Week 9 or 10** | An opportunity to discuss your forthcoming exams and course work deadlines. You may also want to discuss the Semester 2 ‘Reflection and Review’ and share experiences with fellow students. |

Table 1 - Sample PDP Schedule

(Adapted from LLC-UoM, 2010)

Unlike the academic context, in a non academic environment, the learner has to manage this process himself. This can lead to the learning process not being as effective as it should be. However, with support from PDP tools such as the one developed in this project, the learner can follow the process consistently and without much stress. Such tools can also motivate the learner to continue with the process.

The PDP learning process can therefore be summarised in the following steps:

1. Take a skills/knowledge audit
2. Write an action plan
3. Document and keep records as you execute your plans
4. Reflect upon the process, what has been learnt and evaluate the outcome
5. Repeat the steps again in a new direction of learning

PDP is potentially beneficial to the learner, the academic staff and the institution provided the process is carefully implemented and practised.

@TODO If there is time: Representation of Personal Development Plans

Mind Maps, Directional Acyclic graphs, Tables, e-portfolio? A rep or a collection of docs

### 2.5.2 Potential Benefits of PDP

Personal development planning is beneficial to the student/learner as well as to the institution.

PDP will help students/learners in the following ways (Jackson, 2001; LLC-UoM, 2010):

* integrate their personal and academic development and improve their capacity to plan their own academic programmes
* become an independent learner
* be more effective in monitoring and reviewing their own progress
* be more aware of how they are learning and what different teaching and learning strategies are trying to achieve
* recognise and discuss their own strengths and weaknesses
* develop an increased awareness of their skills
* identify opportunities for learning and personal development outside the curriculum;
* be better prepared for seeking employment or self-employment and be more able to relate what they have learnt to the requirements of employers
* prepare their CV and write good job applications
* be better prepared for the demands of continuing professional or vocational development when they enter employment.

For departments and institutions PDP will (Jackson, 2001; O’Connell, 2001):

* facilitate more effective monitoring of student progress
* result in more effective academic support and guidance systems
* enhance their capacity to demonstrate the quality of support they are giving to students in external review processes.
* Administrative efficiency

Much has been said about ‘how’ learning is done; however, ‘where’ learning is done is also an important factor. The environment where learning occurs influences the learning style, process and the outcome. The next section discusses this in details.

## 2.6 Learning Space

In defining the term “learning spaces”, Malcolm Brown, started out with a question. “What does the term learning space mean? Why not use a classroom instead?” (Brown, 2005). Learning spaces as defined by Brown (2005) “encompass the full range of places in which learning occurs, from real to virtual; classroom to chat room”. According to Brown

“Just a decade ago, classrooms were the primary locus for learning in higher education. Other spaces included the library, the faculty office (for individual mentoring), and perhaps the café in town. But classrooms were by far the single most important space for learning.”

However, a great deal has changed over the years with regards to learning theories, styles and activities. Advancements in learning theories have led to a rethink in designing learning environment. The word “room” (as in classroom, lecture room, etc) is no longer descriptive enough as it has been realised that learning can happen everywhere. The term “Learning Space” is increasingly being used to describe places where learning occurs. Information and Communication Technology has also contributed to changing the notion and location of learning as we shall discuss later, thus leading to the evolution of not only modern physical learning spaces but also virtual learning spaces.

### 2.6.1 Trends in Learning Space Design

“Learning Spaces often reflect the people and learning approach of the times, so spaces designed in 1956 are not likely to fit perfectly with students in 2006” (Oblinger, 2006a). Consequently, there have been moves to redesign learning spaces not only to conform to the advancements in learning theories but to also conform to the new generation of learners which Oblinger and other choose to call the “Net Generation Learners” (Oblinger & Oblinger, 2005). According to Oblinger (2006a), there are 3 driving forces behind the move to redesign learning spaces viz:

* Changes in students
* Information Technology
* Our understanding of learning

This view is also corroborated by Brown and Long (2006). According to them, “three major trends inform current learning space design” viz:

* Design based on **learning** **principles** (or theories), resulting in intentional support for social and active learning strategies.
* An emphasis on **human-centered** design
* Increasing ownership of **diverse devices** that may enrich learning.

Obviously these agree with Oblinger’s views. These trends as pointed out by Brown and Long, “have been catalyzed by constructivism, digital technology, and a holistic view of learning”.

The constructivist learning paradigm as earlier discussed focuses on the learner rather than teacher. Thus in constructivism, we drop the “transmitter-centric” mode of learning in favour of the “active construction of knowledge” by the learner. We drop the focus on “teaching” in favour of the focus on “learning”. This emphasis on learning according to Brown and Long (2006), means that we must also “think about the learner” in designing learning spaces. Learning Spaces, according to them, “are not mere containers for a few, approved activities; instead they provide environments for people”.

Consequently, designing a learning space as an architectural master-piece alone is insufficient for the present day learner. Placing high priority on how the learning space enhances learning is also crucial. This must be what Torin Monahan had in mind when he used the term “**built pedagogy**” to refer to “architectural embodiments of educational philosophies”. In other words, “the ways in which a space is designed shape the learning that happens in that space (Chism, 2006). Consider the following examples from Chism:

* A room with rows of tablet arm chairs facing an instructor’s desk in front of chalkboards conveys the pedagogical approach “I talk or demonstrate; you listen or observe.”
* A room of square tables with a chair on each side conveys the importance of teamwork and interaction to learning.

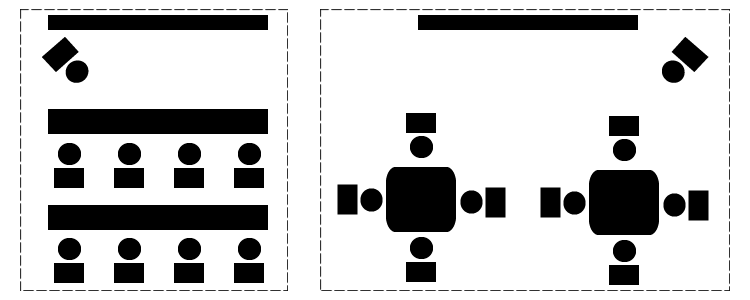


Figure 5 - Linear Arrangement vs. Collaborative Arrangement

Present day students do not like the idea of sitting in front of an instructor like dummies and listening “attentively” to the teaching. Their attention shifts quickly from the instructor to other items such as their mobile devices, course mates, etc. Such learners will definitely not fancy the first example given by Chism above. Oblinger (2006a) describes this kind of learners as favouring “active, participatory and experiential learning”. This kind of learning according to Neill & Etheridge (2008), “requires a flexible space”, and as such the second example will appeal to them since it is more natural to the learning styles they exhibit in their personal lives.

Information Technology is also another very potent factor shaping the trends in learning space designs. Trends in Information and Communication Technology continuously redefine the meaning, boundaries and styles of learning. The recent proliferation of low-cost devices as well as the integration of platforms has given learners a whole new universe of learning – learning that is distributed in “time” and “space”. We focus on this in the next section when we discuss virtual learning spaces. One big problem with technology is the pace of change. The unrelenting pace of technology change, according to Brown and Long (2006) “can make IT decisions rapidly obsolete”. Interestingly, “While platforms and applications come and go, the psychology of how people learn does not” (Brown & Long, 2006). This is why the field of “Instructional Technology” focuses on adapting the changing technology to fit the psychology of learning. According to Jonassen & Land (2000), “Technology foundations determine what is technologically possible, but grounded practice requires determination of how capabilities should be exploited”. Rather than designing learning to suit the technology, the trends with regards to technology focuses on:

* Designing Technology to support Learning
* Adapting Technology (new and existing) to encourage active, collaborative and experiential learning such as the use of Web 2.0 tools, podcasting, mobile devices, Social Networks, etc.

Current trends in learning space design show the following desired characteristics (Chism, 2006):

* Flexibility
* Comfort
* Sensory Simulation – Colours, Lighting, Ambience
* Technology Support and
* Decentredness

A flexible learning space “better enables innovative approaches to teaching and learning when compared to the traditional classroom” (Neill & Etheridge, 2008). With the right approach, “the entire campus can become a learning space” (Mitchell, 2004). Indeed, in the virtual approach adopted here, both on and off campus learning experiences are to be supported. The 3 trends discussed here underlie this emerging reality (Brown & Long, 2006) while the desired characteristics stated above are the features these trends will produce to support active, participatory and experiential learning.

### 2.6.2 Virtual Learning Spaces

While physical spaces are tangible and fixed in time and space, virtual learning space is the direct opposite. Also, whereas physical spaces exist around us, virtual spaces exist on machines and devices.

To eliminate all doubts, the term “Virtual Learning Space” doesn’t in any way suggest that the “learning” is virtual and not original or authentic. It is the “environment” that is virtual, not the learning. I personally will like to call it “Virtual Spaces for Learning” but for purposes of consistency with the literature stick with “Virtual Learning Spaces”.

While there is no single definition for a virtual learning space, most writers define them by specifying their purpose, components and characteristics. Van Harmelen (2011) defines it as “a place where one or more learners can assemble learning materials that are relevant to the pursuit of their learning goals”. Such learning materials might be quite diverse, for example, they could be something like a personal development plan which expiates both learning goals and learning strategies. Alternatively, the content might be a corpus of student work, generated on a particular topic (Van Harmelen, 2011).

It follows therefore that virtual learning spaces are “**designed**” to support and enhance learning through the use of computers, multi-media devices, mobile devices and other technology based tools. Dillenbourg, Schneider & Synteta (2002) suggest the following about virtual learning spaces:

* A virtual learning space is a designed information space.
* It is a social space: educational interactions occur in the environment, turning spaces into places.
* The virtual space is explicitly represented: the representation of this information/social space can vary from text to 3D immersive worlds.
* Students are not only active, but also actors: they co-construct the virtual space.
* Virtual learning spaces integrate heterogeneous technologies and multiple pedagogical approaches.
* Virtual learning spaces overlap with physical environments.

Virtual Spaces are continually being improved to support active, collaborative and experiential learning. The goal in improving virtual spaces is not to use them as replacements for physical spaces as some might wrongly envisage, rather both spaces are meant to be complementary.

### 2.6.3 Usability of Learning Spaces

Usability is defined as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” (ISO 9241-151, 2008). Usability is a “Quality” attribute, therefore in order to define clearly what usability implies in any context, some form of concrete criteria or attributes must be defined through which usability can be measured. A popular model for doing this is based on five quality components or criteria namely learnability, efficiency, memorability, errors and satisfaction.

Jakob Nielsen is a renowned authority in usability engineering. He observes that usability is a “narrow concern compared to the larger issue of system acceptability” (Nielsen, 1993). The diagram below shows the position of usability with regards to system acceptability.

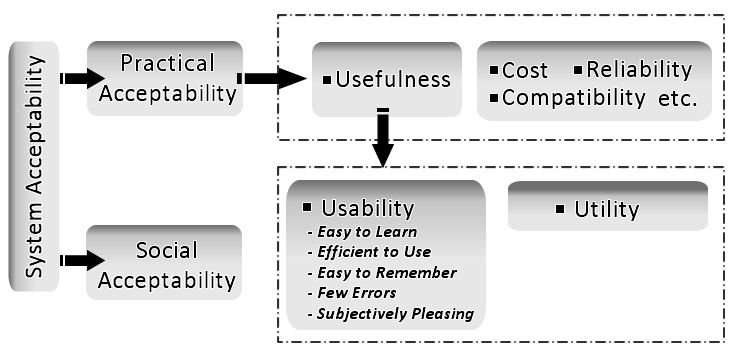


Figure 6 - Attributes of System Acceptability

System acceptability according to Nielsen (1993) “is the question of whether the system is good enough to satisfy all the needs and requirements of the users and other potential stakeholders.” Generally, a system that will be able to “satisfy all” will be a utopian dream. Usually, there is an acceptable level of satisfaction that a system is required to meet. In the framework of social acceptability proposed by Nielsen and corroborated by Ben Shneiderman (1980), Usability, is a defining component of “Usefulness” and is composed of the five attributes identified above which are described below:

* LEARNABILITY: How easy it is for the user to learn to use the system. According to Nielsen (1993), “The system should be easy to learn so that the user can rapidly start getting some work done with it”.
* EFFICIENCY: The level of productivity in use after learning to use the system. In other words, how quickly can the user perform tasks? A high level of productivity is desired in this case.
* MEMORABILITY: The system according to Nielsen, “should be easy to remember”. This will enable the user return to the system after a period of not using it and re-establish proficiency without having to learn about the system from first principles again.
* ERRORS: The error rate of the system should be very low. This does not imply that errors may not occur but if and when they do, how severe are they and how easy it is for the user to recover “gracefully” from these errors. For Nielsen, “catastrophic errors” must not occur.
* SATISFACTION: This is a measure of how “pleasant” it is to use the system. Among all the criteria, this is the most subjective one and is not easy to measure.

For learning support systems, Nokelainen (2006) expanded Nielsen’s usability model by adding “Pedagogical Usability” to the “Utility” branch of the System Acceptability tree and renaming usability to “technical usability”. Nokelainen defines Pedagogical Usability as being “dependent on the goals set for a learning situation by the student and teacher”. It follows thus that evaluating the usability of a virtual learning space becomes more challenging since the technical usability alone is not enough. The environment must also meet the pedagogical demands in terms of achieving learning goals. How do we measure and ascertain that these learning goals have been achieved? Zaharias & Poylymenakou (2009) agree that “evaluating the usability of e-learning applications is not a trivial task”. In order to do a successful usability evaluation, the users, task and context must be identified. According to Zaharias and Poylymenakou, in terms of e-learning, “the main task for the user is to learn, which is rather tacit and abstract in nature”. I believe this is why most e-learning tools have poor usability records. They are either not being evaluated for usability at all or the evaluation is not properly done. To develop an effective usability evaluation framework for virtual learning spaces, the evaluator must familiarize himself with learning theories, learning styles and educational testing research (Zaharias & Poylymenakou, 2009). Three widely used methods for usability evaluation (Hertzum & Jacobsen, 2001) are

* Think Aloud (TA)
* Heuristic Evaluation (HE)
* Cognitive Walkthrough (CW).

Other methods include Questionnaires, Direct Observation, Interviews, Focus Groups, etc.

## 2.7 Existing PLE Architecture to Support Virtual Learning Spaces

The Manchester Personal Learning Environment (MPLE) is an integrated PLE that aims at providing machine support for people to learn together in collocated and distributed settings (van Harmelen, 2010). Its design is based on social constructivism, Papert's constructionism and self-directed learning which have been discussed above. The MPLE provides a layer of general purpose service which can be used for educational purposes. It also contains multi-user, multi-media spaces that can either be used for personal learning or learning in groups. PLE users may either create and use their own spaces, or join a group with other learners, and meet in community created spaces to pursue common learning activities and realise learning goals (van Harmelen, 2009).

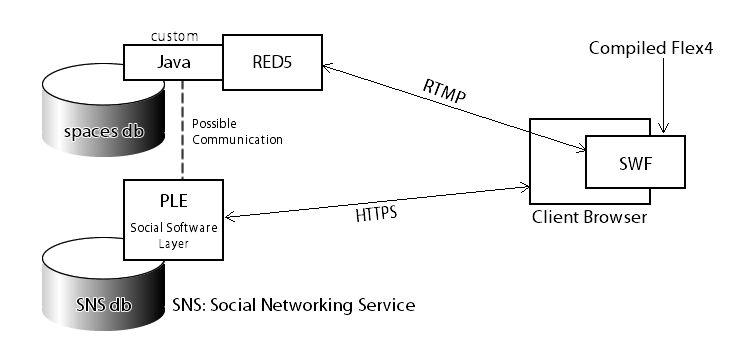


Figure 7 - Overall Architecture of MPLE

The diagram above depicts the architecture of the current MPLE in relation to the media spaces. The media space is written in Flex which compiles to SWF format. This format is executable in Adobe Flash runtime environment available in major web browsers. The spaces connect to a Red5 back end which uses Real Time Messaging Protocol (RMTP) for streaming multi-media content to the spaces. Red5 is written in Java and supports live stream publishing. This enables concurrent editing and updating within the spaces. In previous versions of the Manchester PLE there was some communication from the spaces backend to the PLE social software layer, but this is not exploited currently. Such communication does not involve real-time streaming of media space content. An example can be a notification feed to a user about changes in a shared space.

This architecture will be preserved, as will the use of the Red5 backend. The latter will require modification if this work extends to re-establishing feeds from the spaces to the social software layer.

## 2.8 Summary

Learning spaces are where learning happens. Great spaces foster great learning whether on physical or virtual spaces. Learning spaces should be designed to enhance learning. Doing this requires that the designers of these spaces understand the principles behind learning which are embodied in learning theories and styles. As more learners shift their focus to virtual learning spaces, it also become necessary to consider the design of virtual learning spaces in order to ensure they meet their pedagogical objectives.

# Chapter 3: System Design

This section covers how the theoretical concepts were brought to practical implementation. Due to the nature of work to be done, the design phase was divided into two parts. The first part of the design phase involved understanding the architecture and structure of the existing system. In order to do this an initial study of the existing system was carried out. Expert reviews were also done to reveal some short comings in the existing system. This led to improvements being carried out in the existing system.

## 3.1 Initial considerations and Assumptions

As already discussed in [section 1.2](#_1.2_Project_Objectives), the main focus of the project is improving usability in user generated learning spaces. User generated learning spaces according to van Harmelen (2011) are “(learning) spaces which are populated with content by one or more learners. This brings some concepts as well as questions to mind but before we attempt to outline any of these, it is very important to state clearly some implicit facts with regards to this definition. The basic assumptions are as follows:

1. The content in the learning spaces can come from a variety of sources
2. The content in the learning spaces needs to be persisted so the learner can continue adding to the knowledge base rather than starting from first principles each time
3. The space is meant to be a collaborative space. This implies that the learner needs to be given the ability to share the content of the space with other learners.
4. In a collaborative mode, other learners should be able to contribute to the content by either generating additional content themselves or modifying already existing content.
5. In collaborative mode, there should be a way to synchronise activities between the learner to ensure that one learners activity does not over-write the others’

In order to ensure that the goals of the project are met, it is important to design the system such that it is easily extensible in the future and this requires a careful analysis of the challenges involved in designing this kind of interactive, collaborative system. This is being discussed in the next section.

## 3.2 Problem Analysis

There is no doubt about the fact that learning with multimedia is fun, however, according to Lindgaard, Brown, & Bronsther (2005), “it takes more than an understanding of technology to design useful and usable multimedia interfaces”. The additional requirement Lindgaard et al were referring to is a broad understanding of learning theories which has already been covered in previous sections. Additionally, they also remarked that one must understand “cognitive models describing how sensory information may be perceived, interpreted, stored, and retrieved when we need it” (Lindgaard, Brown, & Bronsther, 2005). The role of animated and static graphics, the amount of information that should be presented on a page, the ease of use of the system, the visual cues and interactive responses etc all impact the cognitive state of the learner while using the multimedia learning space. All these should be considered carefully in the analysis to ensure the end product is not only a product of technical competence but also pedagogically useful and usable.

Research into the learning literature shows that Learning may be described as a three-phased process (Mayer 2002 cited in Lindgaard, Brown, & Bronsther, 2005):

1. The learner should select relevant material(s);
2. The learner should organise the selected material; and
3. The learner should integrate it with existing knowledge.

The first phase of this process is the responsibility of the learner. It can sometimes be very easy for him to select relevant materials to support his learning; however, in situations where he cannot easily find relevant materials, a search using the internet, library resources, asking other learners, etc can lead to discovery of relevant materials. The second phase is usually where it gets tricky and sometimes complicated. The materials selected or discovered need to be organized is such a way that the following can be met:

1. The materials can be easily identified
2. The relevance of the materials can be easily identified
3. The relationship between the material and other materials identified can be easily understood
4. The source of the material can be identified for further research.

The third phase is more of a cognitive process and given that the second phase is well handled, the third happens almost implicitly. However, I have proposed a fourth phase which should act as a guarantee that the third phase was actually achieved. This phase is known as the ‘reflection’ phase and it involves the learner objectively reflecting on the goals that motivated the learning process in order to ensure that they were achieved. This is discussed further under personal development planning in [section 2.5](#_2.5_Personal_Development). The figure below captures the phases. It shows that reflection should not be left till the end, rather every phase can benefit from some form of reflection to improve that phase.

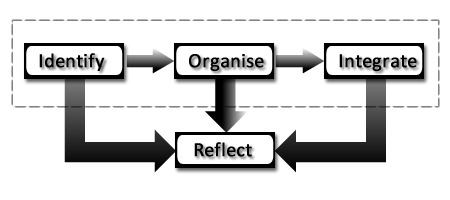


Figure 8 - Learning Phases

In order to enable the learner organise his learning materials and easily integrate them into existing knowledge, the multimedia space had to be equipped with basic building blocks that the learner can use to achieve these. These basic building blocks are the media components, the edges and the toolbox.

### 3.2.1 The Basic Building Blocks

The media components are laid out on a drawing canvas (referred to here as a **‘Whiteboard’**) and organized by dragging, dropping and interconnecting them using directional and non-directional edges. The relationship between the Whiteboard, media components and edges is shown in the figure below:

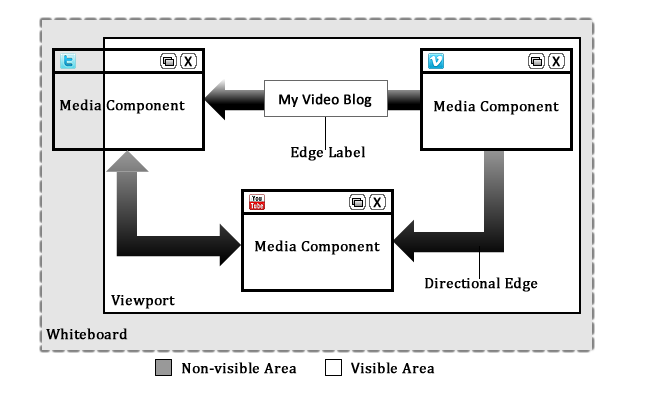


Figure 9 - Whiteboard, Components and Edges

The diagram shows that the whiteboard can be an unlimited span or area where components can be laid out and interconnected to form an organised body of knowledge. Notice however that the entire whiteboard need not be seen all the time; a viewport is used to visualize a portion of the whiteboard at a time. To clarify any doubt, a viewport is a framed area on a display screen for viewing a specified portion of that screen at any given time. The part of the screen that is visible is the part covered by the area of the viewport. Other parts are invisible until they are brought into view either as a result of a user action or automatically in response to some system event. The user can bring other parts if the white board into focus in the viewport by scrolling the viewport in a horizontal or vertical manner. A better way to do this will be to enable dragging the whiteboard into view. This will eliminate the scrollbars and also provide a more interactive; user friendly user experience to the user. This is because while scrollbars need to be activated at specific points to trigger a response, dragging can be done anywhere on the whiteboard that is unoccupied by a component or an edge. It is important to note that while many viewports are considered to be rectangular, there’s no limitation to the shape a viewport can take, depending on its intended use.

The components as shown in the diagram are displayed on the whiteboard and where necessary are interconnected using edge lines. The edge line can either have a single directional arrow head, bi-directional arrow heads or no arrow head at all. This gives the learner the ability to represent the relationship between the components clearly and further helps in the organization phase of learning.

The toolbox is also a basic item required here. It is a collection of icons which are pictorial representations of the media components. The user elects to display a media component by activating its icon in the toolbox. Figure 9 below shows a sketch of the toolbox as well as a screenshot of its implementation.

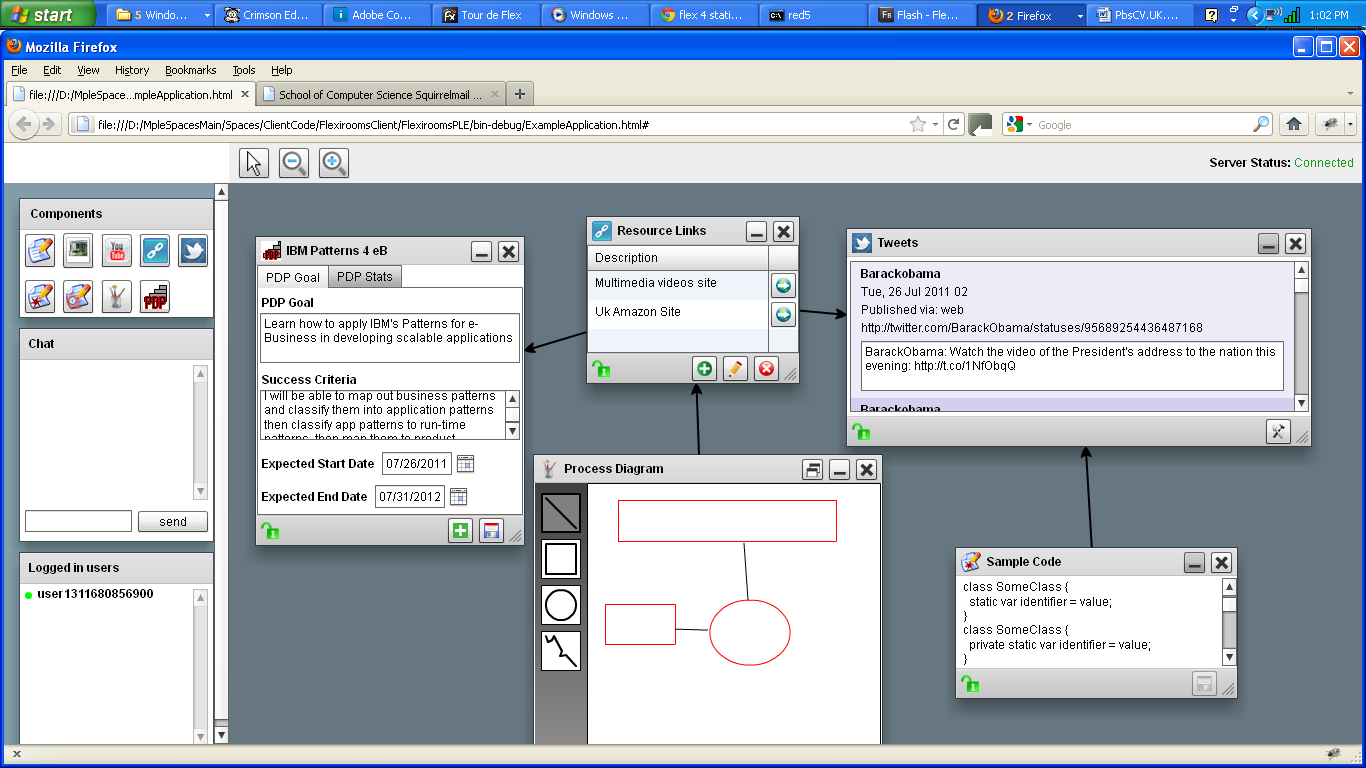
 

Figure 10 - Sketch and Actual Implementation of the Toolbox

Having analysed the problem and devised the basic building blocks which can be used in solving it, the next section examines the design goals of the system.

## 3.3 Design Goals

The design goals of the system revolve around the main objectives of the project as listed in [section 1.2](#_1.2_Project_Objectives). It is important to have clear design goals so as to be able to focus the design and implementation of the system in line with the expected outcome. The main design goal of the multimedia learning space is to allow learners represent knowledge in various formats using visual elements. These visual elements can be textual or graphical. They can be static or animated. In order to achieve this, the elements are represented as components on a whiteboard as described in the problem analysis section ([section 3.2](#_3.2_Problem_Analysis)). The design of the components should be such that the user can easily drop a component on any part of the whiteboard or drag to reposition them on a desired area of the whiteboard. Also, the component should be able to exist in different sizes so that the user can minimize, maximize or drag an edge of the component to resize it to his taste. Finally, with regards to the components, it should be easy to remove a component from the whiteboard when it is no longer needed by the user.

One thing I have not mentioned so far is the process for adding a component type to the whiteboard. In this case, the most widely used approach will be applied. This involves presenting the user with a ‘toolbox’ from which a component type can be selected and then the component should either appear automatically on the whiteboard in its default state or appear when the user clicks on a portion of the whiteboard (maybe drag to lay it out to the desired dimension). The eventual design goal in this case is the former. It is easier and more intuitive to simply make the component appear on the whiteboard as soon as the user selects its representation on the toolbox. In the other scenario, the mouse cursor needs to be changed to signal to the user that something has to be done as a follow up to make the component appear. In many cases however, it is not always intuitive as to whether the user should click or drag to make it appear. Some users may not even realize the need to do any of the above. The chosen plan to make the component appear immediately its representation is selected on the toolbox, is further enhanced by making the component appear at the centre of the whiteboard thereby alerting the user’s attention to the presence of a new item on the screen.

In terms of relationships, edges should be drawn to link components together. In terms of the design goals here, they are as follows:

1. It should be possible to link one component to many other components
2. It should be possible to define a direction for the relationship using arrow heads which can either be unidirectional or bidirectional.
3. It should be possible to label edges using free text to annotate the relationship between components.
4. It should be possible to route edges so that they can trace a flexible path between two components.

There are also design goals for the system in terms of interaction and collaboration. One such design goal is the ability to use a remote cursor controller to track the movement of other users’ mice around the whiteboard. This is a useful feature as each user will be able to see the position of the fellow users’ mouse cursors and as such be able to pre-empt their actions. Additionally, during a presentation, a user can use such functionality to point to elements on the screen while explaining them to the remote audience. The remote cursor can be displayed as a shape (e.g. a small oval or rectangular shape) on the whiteboard but an additional design goal is to attach the user identity to it such that in the presence of multiple remote cursors, the user can distinguish one from another. Another design goal with respect to user interaction and collaboration is the ability to synchronise activities and changes in the learning space among collaborative users. Changes initiated by one user should be immediately updated on the screens of other users. This task requires a careful design of the system as it can be tricky to do this due to the intrinsic concepts behind synchronization such as locking mechanisms and network considerations.

The design goals discussed so far may not be easily achieved given the time frame for the project. Fortunately, some of the goals described have already been achieved in the existing system. Extending and improving the existing system in order to achieve the other design goals therefore become a less difficult task. The next section examines the architecture of the existing system which will give a clearer picture of how some of these goals were achieved. Frameworks, design patterns and concepts used are fully explored here.

## 3.4 Architecture and Layout

The architecture of the existing system was design carefully in line with most of the goals discussed above. The design ensured that every part of the system is cohesive and well decoupled from other parts to foster extensibility. This section gives a clear description of how the system is structured and describes each part that makes up the system.

### 3.4.1 Media Spaces Architectural Overview

In [section 2.7](#_2.7_Existing_PLE), the overall architecture of the Manchester Personal Learning Environment (MPLE) was discussed, describing clearly the relationship between the multimedia learning space and other parts of the PLE. In this section, the architecture of the multimedia learning space is discussed showing the different components that work together to deliver the functionality of the multimedia learning space.

The learning space is made up of three components namely:

1. The Media Server
2. The Space Client
3. The RTMP Connector

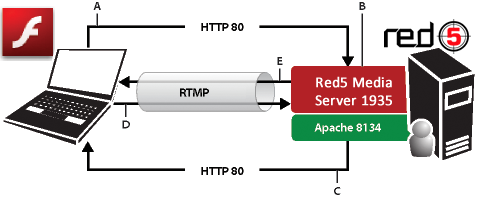


Figure 11 - Media Space Architecture

(Adapted from Adobe Core Server Architecture; Adobe 2009C)

The figure depicts the interaction between the 3 components. Each labelled interaction is explained as follows (Adobe, 2009):

1. Client application (compiled SWF file) executing in Adobe Flash run-time makes a request to the server for content over HTTP. Authentication can be done if necessary.
2. The request is passed by proxy to Apache Web Server.
3. Apache web server delivers the requested content to the client
4. Client application makes a request for content over Real Time Messaging Protocol (RTMP)
5. Red5 media server progressively streams content to client application

The media server is a connection hub. The basic principle behind the media server is that it acts as a central resource point for all connecting clients and thus enables them to share real time information as well as enjoy integrated real-time communication. Using a media server, an application can stream live and on-demand content quickly and easily to a wide variety of platforms and devices (Adobe, 2011b). The following are some of the things that can be done using a media server:

* Live Stream Publishing
* Streaming Video (FLV, F4V, MP4, 3GP) e.g. Live video broadcasts
* Streaming Audio (MP3, F4A, M4A, AAC)
* Recording Client Streams
* Shared Objects which allows clients to interactively share resources in real time.
* Remoting

The abbreviations used above are fully listed in section (@TODO which section, Link it). Flash based applications connect to the media server using Real Time Messaging Protocol (RTMP). This is discussed further below. When connected, the server can exchange (send and receive) data with the connected client applications. It can also invoke methods on target clients. The clients on the other hand can initiate Remote Procedure Calls (RPC) on the server side (Adobe, 2011). Data is transported between clients and server using standard ActionScript objects in Action Message Format (AMF) or other supported formats. Examples of media servers include:

* Adobe® Flash® Media Server (<http://www.adobe.com/products/flashmediaserver/>)
* Red5 Open Source Media Server (<http://trac.red5.org/>)
* Wowza Media Server (<http://www.wowza.com/>)

The existing system uses the Red5 open source media server.

The space client is the flash application that is executed in the user’s browser. The space client is written in ActionScript 3.0 and Adobe® Flex® MXML. The space client acts as the interface between the user and the system. Components are instantiated and organised on the client application while data needed to populate these components are persisted on the server in a database. Due to the collaborative nature of the system, data must be persisted in the server immediately it is inputted into the client components. This will allow other users of the space to be notified of the changes and be updated in real time. A screen shot of the existing space client is shown below.

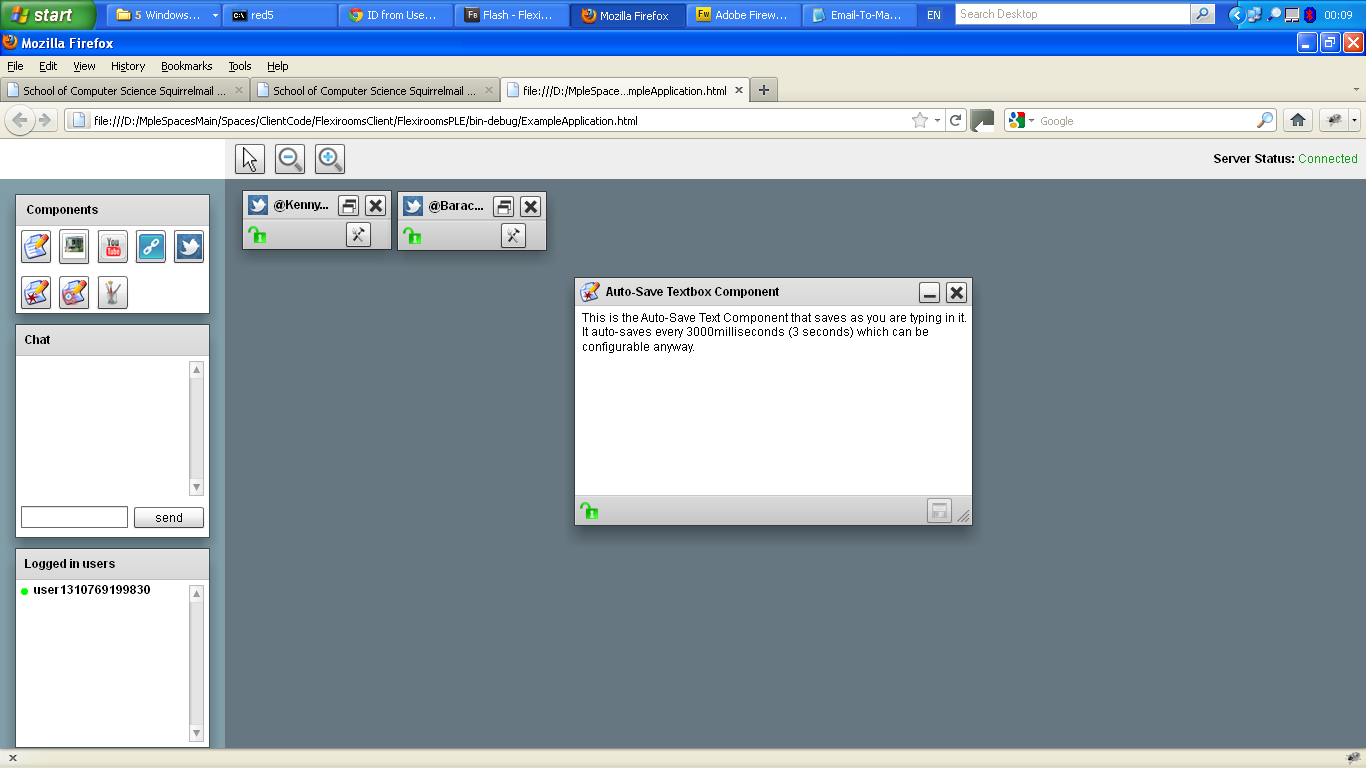


Figure 12 - A screen shot of the existing space client application

(@@TODO Replace with an older version where toolbox is still 4 rows)

The Real Time Messaging Protocol (RTMP) connector also referred to as the ‘Red5Connector’ in this system, manages the communication between the media server and the client application. RTMP is a protocol initially developed by Macromedia (acquired by Adobe) for “high-performance transmission of audio, video, and data between Adobe Flash Platform technologies, including Adobe Flash Player and Adobe AIR” (Adobe, 2011c). The specification of the protocol was later released by Adobe for public use.

|  |  |  |
| --- | --- | --- |
| RTMP CONNECTION | DETAILS | SUPPORTING MEDIA SERVERS |
| RTMP | Real Time Messaging Protocol | FMS, Red5, Wowza |
| RTMPT | Real Time Messaging Protocol (Tunnelled over HTTP) | FMS, Red5, Wowza |
| RTMPS | Real Time Messaging Protocol over SSL also known as RTMP (Secure) | FMS, Red5, Wowza |
| RTMPE / RTMPTE | 128-bit encrypted Real Time Messaging Protocol (Encrypted and Enhanced); RTMPTE (Tunnelled over HTTP) | FMS, Red5, Wowza |
| RTMFP | Real Time Media Flow Protocol | FMS |

Table 2 - RTMP Connections and Supporting Media Servers

(Data source – AskMeFlash (AskMeFlash, 2009); Adobe (Adobe 2009, 2011b, 2011c), Red5 (Red5, 2010))

The architectural structure of the multimedia learning space is very flexible such that any of the components can be easily extended or even replaced to give the system enhanced functionalities. A good example of this is this project which focuses development on the space client, improving the existing components, adding new components and also enhancing the user interaction experience of the space client. The subsequent sections take a closer look at this part of the architecture, describing the basic components (the building blocks), layout, structure and design details.

### 3.4.2 The Space Client Layout

The visual layout of the space client is shown below in Figure 12.

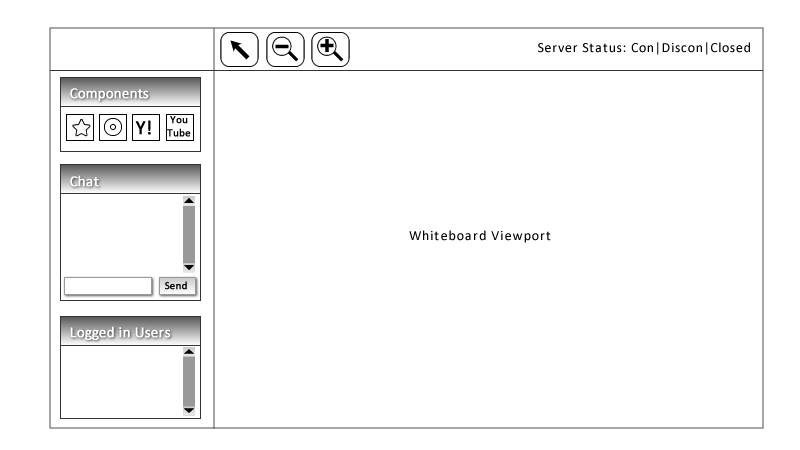


Figure 13 - Space Client Layout

The space client is made up of three sections namely:

1. The Left Sidebar
2. The Top bar
3. The Whiteboard

The left side bar houses the components toolbox, the chats widget and the connected users’ listbox. Icons on the toolbox represent various components and when activated by the user (e.g. via a mouse click), the corresponding component is instantiated on the whiteboard.

The top bar houses the remote cursor control and the zoom tools. The remote cursor control is used by the current user to signal to the system that the mouse cursor position should be tracked and displayed to other connected users. This control is a toggle control which means that activating it when the functionality is active turns of the functionality. The zoom controls on the other hand can be used to increase or decrease the scale factor of the whiteboard viewport thereby making the components appear bigger or smaller on the screen. The extreme right side of the top bar also shows the connection status of the application to the server. The connection status can either be ‘Connection failed’, ‘Connected’, or ‘Connection Closed’.

The whiteboard is the main location where components are displayed, interconnected and organised. As previously described, the whiteboard has a much wider size than the available screen size and as such is being viewed through a rectangular frame referred to as a viewport. Hidden portions of the whiteboard can be brought into view by dragging. This is a mouse operation that involves pressing down the mouse button and moving the mouse while holding the button down.

The layout of the interface is achieved by using flex layouts and containers. The three components are flex canvas components and they are laid out using flex group containers. The positioning of the components is carefully done using the system screen coordinates with the origin position (x,y = 0,0) at the upper left corner of the screen. In designing a visual multimedia application, two things are very important; one of them is a simple, easy to use layout with graphically appealing visual components. The other is a carefully designed implementation of the functionalities of these components so as to make it easy for the user to achieve his goals when interacting with the application. In this case, a carefully designed architecture (the MVCS architecture) is employed which is described next.

### 3.4.3 The MVCS Architecture

An architectural pattern helps in decomposing a complex system into simpler ones with “Responsibilities, Relationships and Interactions” (Berkovitz, 2006). An example is the Model, View, Controller, Service (MVCS) architectural pattern. It is employed to achieve a clear separation of responsibilities in the design of the space client. When a system such as this is being designed, it is important that “cohesion” is increased while “coupling” among components is reduced as much as possible. This promotes reusability because the interaction and interdependency among components is greatly reduced.

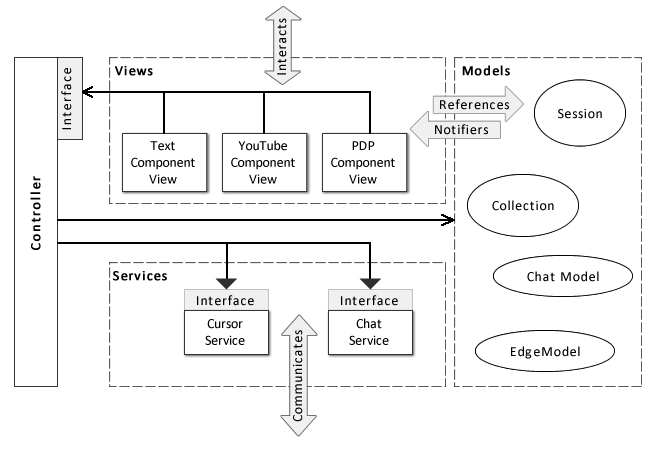


Figure 14 - The MVCS Architecture

The figure above shows a typical structure of the MVCS architecture. Each of the component parts of the architecture is described below.

The **Model** can be viewed as a data store. Its function is to capture and store the state of the application. The state can be represented as objects, collections, properties, etc. The model also watches the stored data and any change to the state causes it to dispatch events notifying the **View** of such changes. The model does not reference any non-model application component (Berkovitz, 2006).

The **View** handles the presentation and interaction. Presenting the state of the application to the user in its raw form (as objects, arrays, etc) will make no meaning to him so the view bears the responsibility of presenting it in a way that the user can understand and relate with. The user, upon understanding the presentation may choose to interact with it for instance ‘select a picture gallery to view from the list of galleries presented’. The view handles such interaction with the user, however, it does not perform the requested operation for the user, it rather hands over control to the **Controller** by invoking one or more of its operations. Another additional responsibility of the view is to respond to notification of changes in the model (as previously discussed) by updating itself to reflect the current state of the application.

The **Controller** is the coordinator of all activities in the application. As shown in Figure 13 above, it is the only part of the architecture that interacts with all the others. The controller can modify the model when necessary; invoke **Services** when needed and coordinate view-to-view relationships. The controller according to Berkovitz (2006) also “acts as Façade for miscellaneous application logic; progress and error reporting; confirmation, validation; security and authentication. The controller however should not bear the responsibility of communicating with the outside world; this is where the service comes in.

The **Service** bears the responsibility for all remote operations and logic. All communication with the outside world such as Web services, Http calls, Remote Procedure Calls, etc are encapsulated by the service. The service also bears the responsibility of populating the model with remote data.

A brief description of how a component gets created in the space client application will give a better understanding of how the MVCS architecture is applied in the design of the space client. The presentation layer of the space client has been described in the space client layout ([section 3.4.3](#_3.4.3_The_Space)), the process is as follows:

1. When the user elects to instantiate a component (e.g. by clicking its representation icon on the [toolbox](#The_Toolbox)), the presentation layer invokes an operation (*addComponent*) on the controller.
2. The controller in turn invokes operations (*initialiseModel*) on the model that causes the model to initialise default data required by the specified component.
3. The controller then goes ahead to invoke an operation on the service component (*createComponent*) passing the initialized model as the single parameter to the operation.
4. The service then makes a remote call (*add*) using the NetConnection and the flash.Net.Responder objects. The NetConnection object establishes the link to the remote server so that the remote call can be invoked while the Responder object handles return values from the server related to the success or failure of the remote operation (ASDocs4, 2011).
5. If the remote operation is successful, the model is updated and a SYNC (sychronize) event is dispatched which notifies the presentation (view) of the changes in the model.
6. The view then updates its display components. In the case of a new component, it is then instantiated on the client side and displayed on the whiteboard. For a component that already existed, the state is updated to reflect the current state in the model.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Use only if time permits

The diagram below is a visual representation of this process using one of the simple components in the space application.

@TODO draws a graphic showing all that has been described above…..

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

The MVCS architecture helps structure the different application parts and organize the relationship between them thus resulting in a highly decoupled system that is easily extensible and scalable. A weak link still exists in the architecture however. This weak link is ‘the object creation logic’. In the next section, this is discussed and a solution proffered to it.

## 3.5 Inversion of Control and the SWIZ Framework

In object oriented programming, encapsulation is supposed to hide the internal implementation details of an object from users by separating its interface from its implementation. The data and the implementation code for the object should be hidden behind its interface. However, this is not usually the case when creating objects. Consider the code below:

ConcreteClass ccObject = new ConcreteClass(actual parameters required)

Obviously, the class creating this object needs to know the actual parameters required by the constructor to create this object as well as the order in which it is required. Creating objects is therefore considered to be the ‘weak link’ in encapsulation. It can therefore become tricky to decide where (and when) to create objects. The GRASP Creator principle (Larman, 2005) and the Gang of four creational patterns (Gamma Et al, 1995) suggest how this might be done for example using the Factory design pattern. An alternative approach to this challenge that has gained popularity over the last few years is the ‘**Inversion of Control**’ strategy.

### 3.5.1 Inversion of Control Strategy

The inversion of control (IOC) design strategy also known as ‘Dependency Injection’ (DI) is “an approach in which a separate object is responsible for populating the fields of other objects with correct implementations, instead of these other objects being responsible themselves” (Eustace, 2009). This approach according to Eustace gives two major benefits:

1. Objects can be decoupled from their implementations by declaring the objects’ fields as interfaces. This is also known as ‘design by contract’.
2. The object creational logic is separated thus making the purpose of the object clearer.

As a follow up on the second benefit, IOC also solves the weak link challenge described earlier thus enabling our MVCS architecture to be fully decoupled as planned. With IOC, the application objects are instantiated in a separate layer and then passed into the application as needed. Sudgen (2009) goes ahead to emphasise the need to decouple the objects of an application “so they can change independently of one another and be tested in isolation”. According to Sudgen, “applying IoC can make it easier to restructure a user interface, substitute the service integration layer, and refactor the models containing the logic and state of your application, amongst other benefits”.

The concept of decoupling the objects of an application can be further explained using an example. The figure below shows how objects are instantiated in a traditional application.

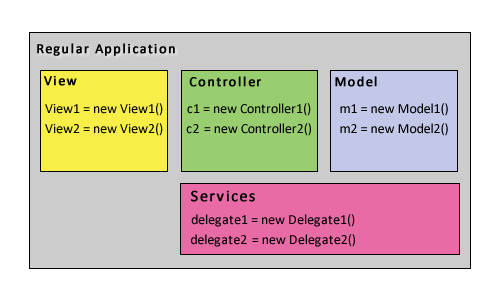


Figure 15 - Objects instantiation in a traditional application

(Adapted from Eustace, 2009)

From the figure it can be seen that the architecture is well structured just as described in [section 3.4.3](#_3.4.4_The_MVCS) using MVCS. However, in creating the objects, the traditional approach is used thus coupling the different layers to each other once again. In order to understand how IOC solves this, the figure below shows an IOC container handling the creational logic as well as the dependency injection logic.

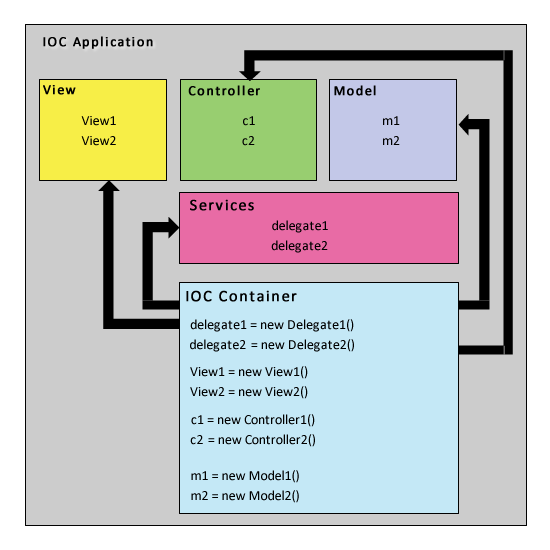


Figure 16 - Objects instantiation in an IOC application

(Adapted from Eustace, 2009)

In this case, the required objects are defined for each layer but not instantiated at compile time. At runtime, the IOC container instantiates the required objects for each layer and injects them as needed. This is also known as ‘**wiring**’ the application.

Inversion of control is usually not as easy to implement as described hence the recourse to IOC containers. An IOC container provides a framework for implementing dependency injection in a consistent and declarative way (Eustace, 2009). Examples of such frameworks for Flex include (Eustace, 2009, Sudgen, 2009):

* Spring ActionScript (<http://www.herrodius.com/blog/>)
* Parsley (<http://www.spicefactory.org/>)
* Flicc (<http://flicc.sourceforge.net/>)
* SmartyPants IOC (<http://code.google.com/p/smartypants-ioc/>)
* Swiz (<http://swizframework.org>)

The Swiz framework was used in designing the space client. A brief description of the Swiz framework and how it is applied in the design is discussed next.

### 3.5.2 The SWIZ Framework

Swiz is “a framework for Adobe Flex and ActionScript that aims to bring complete simplicity to RIA development” (Swiz, 2011). It is more than just an IOC framework; it is meant to be a complete solution for Rich Internet Application (RIA) architecture (Eustace, 2009). Swiz provides the following (Swiz, 2011):

* Inversion of Control / Dependency Injection
* Event handing and mediation
* A simple life cycle for asynchronous remote methods
* A framework that is decoupled from the application code

The basic Swiz configuration can be done in three steps

1. Create the beans file
2. Add a SwizConfig to the application and pass in the beans file
3. Add the metadata to the classes into which you want to inject dependencies

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

@TODO Maybe give example of a Swiz Bean File, a config code and a metadata usage in the Appendices, Use Eustace as base but adapt to Space Application.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Beans in Swiz are MXML (@TODO See Section???) files that extend the *Swiz BeanLoader* class (Orlando, 2009). The content of a beans file is usually a series of object declarations within the beanloader tags. After loading the Swiz configuration, the objects can be wired to the application either by calling ‘Swiz.getBean (“bean-name”)’ or using ‘metadata’ such as Autowire, Mediate, etc. The Autowire tag is used for component wiring, while the Mediate tag is used for dynamic mediation of events during the bubbling phase of the event life cycle (Orlando, 2009).

In the space client application, the Swiz framework is used to inject the dependencies needed and also mediate some events. Controller objects (e.g. whiteboardController), model objects, view objects and service objects are declared in the Beans.mxml swiz beans file. The *SwizConfig* class is then used to inject the beans file into the application. The table below gives a visual summary of some of the objects managed by the Swiz framework in the space client application.

|  |  |  |
| --- | --- | --- |
| ARCH. LAYER | OBJECT | ADDITIONAL DEPENDENCIES |
| CONTROLLER | whiteboardController | whiteboardView, whiteboardService  whiteboardComponentFactory |
| edgeController | whiteboardView, edgeService  whiteboardComponentFactory |
| MODEL | whiteboardModel, cursorModel  edgeModel | None |
| VIEW | whiteboardView, whiteboardViewport | whiteboardModel |
| SERVICE | whiteboardService | whiteboardController |
| chatService | chatController |
| spaceInfoService | spaceInfoController |

Table 3 - Some objects managed by Swiz Framework

# 3.6 SUMMARY (@TODO)

Designing for learning is a task that requires careful consideration. So so and so person remarks that we cannot design the way learning takes place, we can only design for it. Blab la bla.

# 4 Components Design

Having given a description of the system in chapter 3, this section describes the design of the components within the system. A lot of attention is given to the personal development planning component also referred to here as the Personal Development Planner. This is because of its uniqueness due to its pedagogic importance. Since personal development planning has already been discussed in [section 2.5](#_2.5_Personal_Development); we focus on the implementation of the component based on the theories of PDP and CPD already discussed. The design of other components such as the twitter search component; the audio component; the space painter; etc is also described. The next sub section gives a brief overview of the PDP component within the context of a real scenario.

## 4.1 Overview of the Personal Development Planning (PDP) Component

The PDP component enables the learner to take control of his learning. The process begins with a self audit (sometimes referred to as skills audit). Self audit is carried out by the learner with regards to the subject about to be learnt. After performing a self audit, the learner proceeds to state his learning goals and success criteria. Steps to be taken to achieve them within a specified time frame are also documented. The completed personal development plan therefore becomes a guide for the learner to organise and document his learning process. The learner can connect other relevant media components to the PDP component to document various materials linked to the learning goal. As the learner continues to attach other useful components to the PDP component, it evolves into a knowledge artefact that can be (re)used by other learners to achieve similar learning goals.

Take for example, a learner who wishes to learn how to play the piano. He will select a PDP component and fill in the necessary data such as the goal, success criteria, target time frame, action steps, etc. Having done that, it becomes quite easy for him to search for useful materials that will help him achieve his goal by following the listed steps. Assuming he finds a YouTube video, he can then select a video component and link it to the PDP component using one of the edges (directional or otherwise). The edge can be labelled to further clarify the relationship of the linked video with the PDP process. The label also indicates the purpose of the linked video. Other components such as links, text, images, etc can be linked to the PDP component thereby resulting in a knowledge artefact that can be (re)used by the learner (or others) to achieve the intended goal.

A PDP component can also be attached to another PDP component thereby forming a chain of goals and actualisation steps. For example, a PDP component whose goal is ‘Learn to program in PHP’ can be linked to another PDP component whose goal is ‘Learn to develop web applications’. The figure below gives a sketch of a PDP component and a screen shot of the implementation.

@TODO Sketch of PDP Component as well as screen shot of it?

4.1.2 Data Requirements for the PDP Component

The major data required for the PDP component are as follows:

* PDP Goal or Learning Goal
* Success criteria
* Target dates
* Steps to be taken
* Status of the process
* Reflection and evaluation notes

The PDP Goal is the main aim of the learning process. The learner sets a goal for himself and then specifies success criteria to be used to ascertain that the goal has been achieved. An example of a goal and a success criterion is given below:

* **PDP GOAL**: Learn to develop web applications in Rails
* **SUCCESS CRITERION**: I will develop a Couples-Matcher application for the School of Computer Science May Ball, 2012

The target dates include target start date and end date. For learning to be effective, the goals need to be **S**pecific, **M**easureable, **A**chievable, **R**ealistic, and **Time bound** (SMART). The last item is the essence of the target dates. Setting target dates ensure that the learner does not spend too much time trying to achieve a learning goal at the expense of other things. It also enables the learner to develop added skills such as time management, ability to learn things quickly, etc.

The steps to be taken are listed out carefully and can be used to adjudge the status of the process. Finally, reflection helps the learner to think about the process at each stage and also at the end. An evaluation can then be carried out by the learner to improve the process in the future.

Designing the PDP component began with an initial design based on the descriptions above and concepts from the literature discussed in [section 2.5](#_2.5_Personal_Development). The initial design was then refined until an acceptable design was gotten. This process is discussed next.

## 4.2 Initial Design of the PDP Component

The initial design of the PDP component comprised three multimedia items namely:

1. The main PDP component,
2. The PDP Action List Component, and
3. Other Media Components attached as needed (e.g. Text, Links, Audio, Video, etc)

The illustration below shows the initial conceptual design of the PDP component:

@TODO initial conceptual design (kind of domain model?)

The figure shows the relationship between the items that make up the PDP component. The following can be deduced from the diagram:

* The main component can be linked to one or more Action List component(s)
* The Action List component can be linked to zero or more media components
* Each item (the main component, action list component, media components) can be linked to a similar item to create a more complex learning plan.

The figure below shows a sketch of the implementation plan for the component described thus far:

@TODO Draw sketch of Implementation plan of the Initial Design

In order to use the PDP Component according to this design, the learner will pass through the following stages:

1. **The Main PDP Component**: This component contains the basic PDP data such as the goal, the success criteria, target dates, etc. The learner will fill out these data in the spaces provided. A button on this main PDP component (*Add Action List Component button*) enables the learner to create another sub-component called ‘Action List component’. This sub-component is automatically attached to the main PDP Component as shown in the figure above.
2. **The Action List Component**: This component enables the learner to set a target and then outlines plans to achieve the specified target. According to this design, it should be possible to connect multiple action list components to the main PDP component. Each connected action list component should contain a target that contributes to fulfilling the overall goal recorded in the main PDP component.
3. **The Media Components**: After preparing action list components, the learner can then connect other media components to the action list components. The type of media components connected to an action list component will be determined by the target of the learner as contained in the specified action list component. These could be audio, video, link, text, etc.

The initial design of the PDP component was collaboratively evaluated with my project supervisor during which certain challenges were discovered that led to this design being refined to produce a second design.

### 4.2.1 Evaluation of the Initial PDP Component Design

The initial design of the PDP component was indeed an attempt to capture all of the concepts in the literature on PDP. However, that came with a trade off. Doing so introduced a lot of complexity into the final product. The component was complex, the process of using it was complicated and non intuitive to the user. The following are some other deficiencies identified with the initial design:

* The component required a steep learning curve for the user to be able to use it as intended
* The component (or group of components) takes up too much space on the whiteboard which meant the user had to drag the hidden parts to view each time he needed access to them. This can become an exhaustive task and discourage the purpose which is to learn.
* The complex structure of the component requires a lot of data to be maintained not only about the individual components but also about the relationship among them. This introduces further complexity in the persistence layer as well as the network layer with regards to data storage, transfer and synchronisation between collaborative users.
* The complex structure also requires the user to keep track of too many components thus introducing an additional task to the learner. While trying to organise his learning materials, he is also trying to organise the PDP components. Hiding all other action list components and their sub components when one is selected could be helpful in this case but that means introducing additional controls to hide and un-hide them thus increasing the already steep learning curve.
* In terms of the learning goals and sub-objectives required for each action list component, it was noted that breaking down a learning goal to sub-objectives can be a difficult task. Removing the objectives from the action list control could be helpful in this case but that also means the component becomes just a list of steps to be taken to achieve the main goal. This was the first indication that both could be merged to form a single component.
* Breaking down a learning goal into multiple objectives can also be time consuming and will discourage the average learner from using the component frequently to support his learning.

These listed points led to the decision to refine the design so as to make the component more usable to the learner. The objective of the refinement was to “simplify the design on the basis that students deal best with very simple things” (van Harmelen, 2011b).

## 4.3 The Refined Design of the PDP Component

The points identified above ([section 4.2.1](#_4.2.1_Evaluation_of)) after carefully evaluating the initial design led to a series of refinements on the design of the PDP component until a final design was produced that satisfied the goal of the process. The figure below shows the refined design of the component.

@TODO Refined design of the PDP Component (Already sketched in my proj note book, dated 23-July-2011)

The refined design eliminated the action list component and merged the steps with the main PDP component. According to this design, all the learner is required to do is:

1. State his learning goal
2. State one or more success criteria
3. Specify target start and end dates where necessary
4. List steps to be taken in achieving the specified learning goal
5. Attach other media components as required to document the knowledge necessary to achieve the learning goal

The final design simplifies the concept such that the component can be used by the learner with minimal assistance. In order to achieve a neat user interface layout, a ‘tabbed navigator’ control was used to separate the PDP Goal from the PDP steps. Further details about the design of the component are discussed next.

### 4.3.1 Design Details for the Refined PDP Component

The illustration below shows the class diagram for the refined PDP component.

@TODO Draw class diagram for Refined PDP Component (Not showing parent base not or what ever, just the classes as PDP Goal Component class, steps class, steps item class, etc)

The responsibilities for the component are split among multiple classes. The PDP Goal class handles all responsibilities that have to do with the goal, success criteria and target dates. The PDPSteps class manages the data and functionalities associated with the steps. This class is composed of one or more PDPStepItems which is another class that encapsulates the functionality of a PDP step for example marking a step as completed or otherwise (step status). The PDPSteps class is responsible for adding and removing PDPStepItems. When it does this, it dispatches an event to notify all observers (objects listening to the event). The PDPComponent class captures this event and invokes a method on the controller to update the model.

The PDPComponent class does not function in isolation. It requires the services of other object in the system such as the controller, the service and the model objects. The diagram below shows the architecture of the PDP Component with regards to the other objects in the system.

@TODO Architetural overview of PDP Component

The architectural design shown above is based on the MVCS architecture described in section (@TODO you know what to do!). The view layer is further decomposed into sub layers to separate the styling and skinning of the component from the functionality and business logic of the component. Separating a component’s display skin from its business logic is considered best practice when building components in flex (@TODO Ref Adobe here). It allows for loose coupling between the component and its constituent parts thus giving the developer the ability to swap any part for an alternative when needed. This separation of skin parts is the major difference between Flex 3 components (aka Halo Components) and Flex 4 Components (aka Spark Components). Flex is further discussed in section (@TODO which?). The refined PDP component though simplified; maps well to the PDP learning approach discussed below.

### 4.3.2 Mapping the PDP Learning approach to the Refined PDP Component

The PDP learning approach is described in section (@TODO, you know what to do). This approach can be mapped to the PDP component as follows:

|  |  |
| --- | --- |
| PDP LEARNING PHASES | IMPLEMENTAION |
| Auditing | Not implemented. This is assumed to be done externally by the learner before proceeding to use the PDP |
| Planning | PDP Goals Tab for recording goals and success criteria  Target dates for time span estimation  PDP Steps Tab for brainstorming steps to be taken |
| Executing | Connecting relevant media components to the PDP component and populating them with useful content |
| Reflecting | There is no tab on the PDP component for this yet but a Text Component can be connected to the PDP Component and used for reflection notes |

### 4.3.3 Improvements to the Refined Design

Although the refined and accepted design of the PDP component was simplified enough, there still existed room for improvement. One prominent question was “how will learners prefer to manage the steps in the process”. This question is relatively important as it can determine which controls should be used in designing the PDPSteps components and how they should be used. If the learner prefers to list all the steps together in one place, then that might require a ‘TextArea’ control. On the other hand, if he prefers to manage each individual step separately, then several ‘Textbox’ controls might be the better option. To answer this question, two versions of the PDP component were built; one for each style of steps management. The screen shot below shows the implementation for both versions of the component.

@TODO screen shot of version 1 and 2 of PDP Component

The first version requires the user to add each step individually in the steps component provided. The user can add another step component by clicking the add step button. The second version however allows the user to manage the steps in a traditional way of typing them in a single list. Both versions were evaluated by user and the results of the evaluation are documented in section (@TODO you know what to do).

## 4.5 The Audio Component

The audio component allows the user to listen to audio files. These files can be added by specifying their URLs on the internet or alternatively uploading them from the user’s computer. The initial design of the component enabled the user to specify a single URL which the component stored and played whenever the user elected to listen to it. After a review of this design, it was however improved upon. The reason for this was because using the component does not involve any visual interaction between the user and the system apart from simply instantiating and specifying the audio URL for the component. Therefore, listening to more than one audio file means the user will have to instantiate multiple audio components. These components will take up space on the whiteboard and consume memory too. A better design is to enable a single audio component play audio files from different sources. The refined design is illustrated below:

@TODO illustrate of screen shot audio component.

The new design featured a ‘playlist’ which listed all the audio files added to the component. To play the audio file, the user either double clicks one of the listed sources or selects a source and click the play button. The sources of the audio files can be specified by activating the ‘add audio source button’ which brings up the interactive dialogue box pictured below.

@TODO Screen shot of add audio.

The dialogue box allows the user to either specify an audio URL from the internet or select an audio file from his system which is then uploaded and added to the playlist. The user can optionally specify a title for the audio source. Where this is not done, the component detects the title from the source and displays it in the playlist. Audio sources can also be deleted from the component by selecting them on the playlist and activating the delete source button.

Implementing the audio component required some special considerations on how audio files are used. Since the space is a collaborative environment, a change made by one user has to be updated across other users’ interfaces. For the audio component however, not every change had to be updated for all users. For instance, it should be possible for each user to listen to different audio files on the playlist. This particular scenario is very interesting because an audio source being listened to by one user can be deleted by another user and the deletion need to be updated across all users. This update requires that the audio file stops playing and is removed from the playlist. This was initially tricky to achieve as the audio file kept on playing even when the source was removed. It was later achieved however, by maintaining a record of the currently playing audio file and then using this record to search for and stop the audio file when deleted before removing the source.

Another interesting challenge was displaying a now playing icon for one user on a particular audio source on the playlist while displaying the icon for a different source on the playlist for another user. Initially, updating a field to specify the source now playing caused the updated change to display multiple ‘now playing icons’ for several audio sources in other users’ playlists. This functionality was later achieved by managing the currently playing source locally for each user. If an update occurred that changed the status of the currently playing source, the status is restored using the local data source of the user.

The audio component has some limitations. One of such limitation is the maximum file size for audio files uploaded from the user’s system. The size is limited by various factors such as the maximum upload size allowed by the web server processing the upload; the strength of the network and the file type that can be played by the component. Another limitation is that internet sources cannot be guaranteed to remain there for as long as the user needs it. The audio component however can inform the user when the source of the audio file is missing. The user can therefore delete the item from the playlist or change its source.

## 4.6 The Twitter Search Component

Twitter is a popular online social networking and micro-blogging service. Users of the service can post, read and re-post text based information of up to 140 characters. These posts are informally known as "tweets". The twitter component enables the user to search for keywords within tweets. The user specifies a search keyword and component retrieves the result of the search and displays them to the user. Twitter can be a very good source of information because users usually insert shortened URLs into tweets which when clicked points the reader to more details about the tweet. Additionally, the user can stay current about trends on a particular topic from the results returned since they are listed by date; the most recent ones first. The illustration below shows the design of the twitter search component and a screen shot of the implementation.

@TODO Design and screen shot Screen shot of twitter component.

## 4.7 The Space Painter Component

The space painter component is a drawing component. This component enables the user to use basic shapes and colours to draw graphics within the application. The component was adapted from the shapely demo application by Haase (2010) in his book ‘Flex for fun’. The component comprises 2 major sub components namely:

1. The toolbox and
2. The drawing canvas

An illustration of the graphical user interface (GUI) design of the component is shown below

@TODO draw a design or screen shot, whichever

To use the component, the user has to select a tool such as rectangle, ellipse, line, etc from the toolbox. The fill and stroke colour section of the tool box allows the user to specify line and fill colours for the shapes to be drawn. The user can also give the shape a gradient fill by specifying two colours as the fill property.

4.8 Summary

@TODO Write this later.

# 5. Implementation

In order to achieve the design goals, it is important that suitable tools be used to implement what has been designed. During the implementation phase, the architectural design of the system as well as the components was carefully translated to code using appropriate programming languages and platforms. Adobe® Flex 4.0 and ActionScript 3.0 were the programming languages chosen for the task. Also, carrying out any form of implementation required a good understanding of the existing code base. A period of ‘code reading’ and performing preliminary modifications to the code base provided the required knowledge to carry out the implementation. This section begins by discussing the tools selected for the implementation. An overview is given for each tool and justification is given for the choice of the tools. This is followed by a description of the implementation details as well as improvements made during this process. Challenges faced during the implementation and how they were solved is also discussed.

## 5.1 Implementation Platform, Languages and Tools

The multimedia learning space is design to be accessed from the internet via a link on the PLE. However, the application is designed to be a Rich Internet Applications (RIA). A Rich Internet Application is web application that feels, looks like and functions like a desktop application. A traditional web application is stateless. This means that the state of the connection with the server is not maintained; rather every new request from the user is treated as a new and independent transaction. Also, in a traditional web application, most of the processing tasks are performed on the server side. With RIAs, the reverse is the case. The connection is state is maintained and most of the processing is moved to the client side with the application contacting the web server only when it is necessary. This minimises the amount of round-trip to the web server and therefore increases the response time of the application. JavaFX, AJAX, Adobe® Flex and Microsoft® Silverlight are examples of some technologies used to build RIAs. Adobe® Flex was the technology of choice for implementing the multimedia learning spaces.

### 5.1.1 Adobe Flex Framework and Flex SDK

Flex “is a highly productive, free, open source framework for building expressive mobile, web, and desktop applications” (Adobe, 2011). Web and mobile applications built using Flex share the same code base and can both be deployed as desktop applications. Applications are built in Flex using MXML tags and Action Script®. They can then be executed in the Adobe® Flash® runtime mostly for web environment and the Adobe® Air® runtime for desktop applications. The Flash runtime is widely available on all major internet browsers while, the Air runtime can be easily downloaded online. Flex applications running in web pages do not require a page reload to update information on the user interface. They connect to remote server side applications to retrieve data which are used to update the client interface as required. These features make Flex a very suitable candidate for building RIAs.

The Flex framework “provides the declarative language, application services, components, and data connectivity developers need to rapidly build rich Internet applications (RIAs) for mobile, web, or desktop” (Adobe, 2011d). Since it is an open source framework, it can be extended to suit the specific needs of the application being developed. The declarative language being referred to by Adobe above is ‘MXML’. MXML is a “declarative XML-based language, is used to describe user interface layout and behaviours” (Adobe, 2011d). As already stated, MXML is used to describe the visual aspects of the application while ActionScript is used for the more programmatic functionality of the application, like the business logic, the client logic, etc (Adobe, 2011d; Haase, 2010). This separation of responsibility in the Flex framework enables the developer to customise the look and feel of Flex components through their skins. Component skins are written in MXML files. They define the graphical elements that describe the visual appearance of the component (Haase, 2010). Additionally, the separation of concern encourages and supports parallel application development by developers and designers. The framework also comes with prebuilt application services and components that help developers build applications faster (Adobe, 2011d). DataGrids, Charts, Formatters, Validators, and other UI controls are some examples of components that come with the framework. Some prebuilt services include data binding service, drag-and-drop, the display system (for managing layout of the User Interface), the effects and animation system, the style system (for managing the look and feel of the controls and components) and the pop-up manager (Adobe, 2011d).

The Flex software development kit (SDK) is a developer toolkit for development in Flex. It includes a compiler, a debugger and a profiler. Each of these tools can be used from the command line (terminal, console) or from an integrated development environment (IDE). The current version of the SDK released by Adobe is version 4.5. It is important to note that components in Flex SDK 3.X (also known as Halo Components) are slightly different in implementation from the components in Flex SDK 4.X (also known as Spark Components).

### 5.1.2 Adobe ActionScript

Adobe® ActionScript® is an object-oriented language based on industry-standard ECMAScript. When developing applications using Flex, ActionScript is the language used to build client-side application logic (Adobe, 2011). Action script looks a lot like JavaScript in syntax and semantics. It was originally designed by Macromedia for website animation and released with Flash 4. The latest version of the language which is version 3.0 has been expanded to include a lot more functionality including database access. ActionScript is an event based language. This means that actions are triggered by event coming from either the framework or from user interaction with the application. ActionScript and MXML work together in a Flex application as described in the next section.

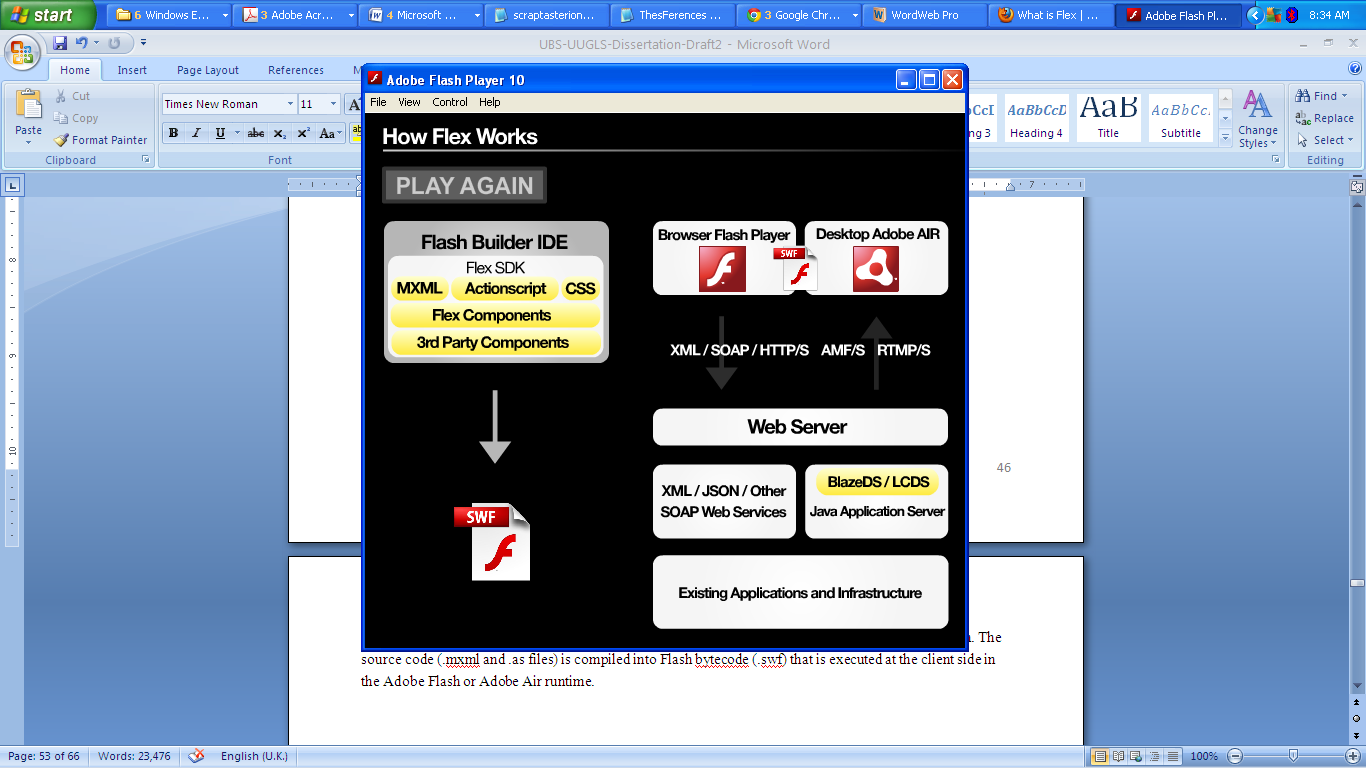
### 5.1.3 Adobe Flash Builder

Flash Builder is an integrated development environment for building Flex applications. It was formerly known as Flex Builder and is based on the Eclipse™ IDE Framework. A plug-in also exists that can be installed on any existing Eclipse™ based IDE. Flash Builder’s intelligent code assist function helps accelerate development in flex. The IDE also has an interactive debugger and profiler which can connect to Adobe Flash Player (debugger version) to assist the developer debug and profile the application. The premium version of the IDE also features a Network Monitor that can be used to monitor and analyses requests to the web server and responses. The latest version of Flash Builder is version 4.5 and it has support for editing MXML, ActionScript and Cascading Style Sheets (CSS).

### 5.1.4 How Flex Works

The figure below illustrates how a Flex application works.

@TODO, figure of how it works

Adapted from Adobe PET

MXML and ActionScript are compiled together into a single SWF file that makes up a Flex application. The source code (.mxml and .as files) is compiled into Flash bytecode (.swf) that is executed at the client side in the Adobe Flash or Adobe Air runtime. The SWF file communicates with the web server as necessary to perform the operations requested by the user.

### 5.1.5 Version Control System (VCS)

Version Control also known as Revision control or Source control is a way of managing changes to documents, program source codes or other files. Source control is very useful for tracking changes to files especially in a collaborative environment where more than person may make changes to a file at the same time. Changes are usually marked by a unique code. The file or group of files being managed can be “branched”. This involves making a duplicate copy of the original document and continuing work on the copy while the original remains untouched. The version control system used for this project is called **Git**. Git is a “free & open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency” (<http://git-scm.com>).

\*\*|\* Miscellaneous Tools (Astah, Github, etc)

## 5.2 Preparation for Implementation

Extending an existing code base to add new functionalities requires an understanding of the existing code. If the existing code was written using Test Driven Development (TDD), then ‘unit tests’ will exist and that can be very helpful. After making changes to the code, the tests can be executed to ascertain that nothing was broken by the change. Where there are no tests, one must do a bit of code reading to understand the structure of the code and how different parts work together in the application before attempting to make changes to the code.

(@TODO Invite Code reading book here, talking about how one can get lost code reading) Code reading is an art in itself and can make the difference between this and that.

### 5.2.1 Understanding and Improving the Existing Implementation

Understanding the existing implementation involved code reading. However, code reading without an aim can lead to the process becoming an endless journey. In order to avoid this, the task of fixing existing bugs in the current implementation was set as the goal of the code reading process. This provided a direction and focus and ensured that the process was carefully monitored. Some improvements that were made to the existing implementation are discussed below.

1. Auto Save in Text Component: The existing text component required the user to click on the ‘save’ button after making changes for the changes to be updated. The improvement to this required the component to automatically save the changes as the user was typing. This was successfully implemented.
2. Title Disappearance Bug: The title bar of components contains a label that can be edited for each component. An existing bug in the implementation prevented the edited label from being persisted in some scenarios. In the first scenario, when the user double clicked on the title bar to edit the title, if nothing is entered into the textbox, the title disappears. This is not correct as the previous title should have been maintained. Secondly, while editing the title, if the textbox loses focus, the title disappears. Finally, even when the title was persisted, refreshing the browser to reload the application still led to the disappearance of the component title. All these bugs were fixed and the title now persists as required.
3. Fully Expandable Components: Existing components could only be minimised, restored or resized by dragging subject to minimum and maximum dimensions. The desired improvement was for components to be able to expand to fit the entire dimensions of the viewport. This was successfully implemented and tested. A base component that handles the functionality was then implemented such that other components that require such functionality can simply extend the base component by inheritance.
4. Resizing bug in Video Component: resizing the video component when a video is loaded made it lose its aspect ratio thereby making the video look stretched and out of proportion. This bug was due to the fact that the loaded video is an SWF file and therefore tries to adjust itself when its container’s dimension changes. Fixing this bug required that the resizing process be strictly controlled to be in synchronisation with the aspect ratio of the loaded video. Unfortunately, this bug is yet to be corrected as at the time of this report.

## 5.3 Adding New Components

The improvements and bug fixes made to the existing implementation provided the understanding necessary for adding new components to the application. The following new components were added:

1. The PDP Component
2. The Audio Component
3. The Twitter Search Component
4. The Space Painter Component

The design and functionalities of these components is described in section 4. The diagram below shows the generic class diagram for the components:

@TODO draws and explain UML implementation diagram for the components in general

Each class in the diagram is briefly discussed below. The properties and functionalities are described.

**The BaseWindowView Class**

The BaseWindowView is the base ActionScript class for all components in the application. This class extends the Adobe(r) spark panel component and provides the following functionalities for the sub classes.

* A titled GUI window which it inherits from the spark Panel class.
* Basic window controls such as the title bar icon; the window resizing icons; the close icon and the status bar
* Custom window events such as minimise and maximise events. These events are dispatched when the user triggers the corresponding actions.
* Custom ‘LOCKING’ events such as ‘Locked by current user’, ‘Locked by another user’, ‘Locked by no one’. Locking is used in collaborative mode to control access to shared resources. When one user is using a component that requires locking, it is obtained through the controller and the appropriate lock events are dispatched.
* Custom effects that get executed when the components are resized, minimised, maximised and moved.
* Implementation for basic functionalities such as ‘minimise’, ‘maximise’, ‘expand’, ‘resize’, ‘close’, etc which are later overridden and customised by sub-classes.
* A reference to the controller and model

This class is also responsible for initialising the basic properties of a component when it is instantiated. These properties include the component title, x-position, y-position, width, height, minimum width, minimum height, maximum width and maximum height. The initial values for these properties are requested from the model and stored locally as properties of the component.

The class also sets up a ‘Watcher’ for the basic properties of the component. A ‘Watcher’ is a function that is executed when the model notifies the component of changes to the property being watched. This is a classical implementation of the observer pattern. The components listen for notification events from the model about changes to specified properties. When this notification is received, corresponding functions are performed for each property that is being ‘watched’.

**The BaseNodeComponentView Class**

The BaseNodeComponentView class extends the BaseWindowView class. This class has the responsibility for the following functionalities:

* Setting up the edge controller which handles the linking of components to each other
* Setting up the title display label and text box which is used by the user to edit the title
* Basic functions for handling the linking of components using edges
* Getters for some basic component properties such as ‘x’, ‘y’, ‘width’ and ‘height’

The Swiz component which has already been discussed in section (@TODO) handles the creation of several objects required by the component such as the edge controller, edge view, and edge model. The BaseNodeComponentView also holds the responsibility of calling the Swiz component ‘autowire’ method which does the creation of the creation and setup of the required object through inversion of control principle.

**The Implemented Interfaces**

Three interfaces can be identified from the diagram namely:

1. IFocusManagerComponent
2. IWhiteboardComponentView
3. INodeComponentView

Two of these interfaces are implemented by the BaseWindowView class while one is implemented by the BaseNodeComponentView class.

The *INodeComponentView* interface defines four methods used to retrieve the basic properties of the component. These include the ‘x’ and ‘y’ position as well as the ‘width’ and ‘height’ of the component.

The *IFocusManagerComponent* interface is a Flex interface within the mx package. It “defines the interface that focusable components must implement in order to receive focus from the FocusManager” (ASDocs4, 2011). This interface is implemented by the BaseWindowView class and is used to capture focus for the media components. One practical use for this focus capturing is to control the layering of components. For example if there are four components on the whiteboard and a user selects one of the components that is slight covered by other ones, the selected component is move to the top and the other components arranged behind it. The selected component is thus said to have received focus.

The *IWhiteboardComponentView* interface defines the interface for setting the component controller and model. It also defines the interface for initialising the model; binding it to the component and retrieving a reference to it when required.

**The Media Components**

The media components are the final items in the UML diagram above. They are sub classes of the BaseNodeComponentView class. Functionalities specific to each component are implemented in the components separately while shared functionalities are inherited from the hierarchy. These shared functionalities can also be overridden to customise their implementation for each component. A good example of such overriding is during resizing of the components, the method can be overridden to ensure that the components do not get resized beyond a particular visual limit or to ensure the components are adjusted evenly based on a specified aspect ratio. Each component is made up at least two files:

* The View file (ActionScript class)
* The Skin file (MXML file)

Both files make up the ‘View’ layer in the MVCS architecture (see [section 3.4.3](#_3.4.3_The_MVCS)). The Flex controls or custom controls that make up the visual appearance of the component are defined using MXML in the skin files. Various display properties are set to arrange the controls according to the design required. Flex also provide containers for laying out these controls as desired. The logic for interaction is however moved to the view class file written in ActionScript. This class contains the codes that specifies which controls are displayed and when they are displayed to the user. When the user interacts with the controls, the response is also handled by this class. In Flex, the best practice is to implement a loosely coupled component model. Separating the view logic and content from their visual implementation enables this practice at the view layer. Other sub components required by the media components such as the media player for the audio component; the PDP Step Items for the PDP component, the ItemRenderer for the Twitter search component; etc are stored in separate packages which are named to identify the component they belong to. The components are carefully implemented according to the design to ensure a loosely coupled system that is amenable to protected variations.

## 5.4 Challenges Faced During Implementation

While there were many challenges faced during the implementation phase, one challenge was particularly stressful. Persisting objects proved to be a non-achievable feat. Initially, the data required to setup each component were meant to be stored in custom property classes for each component. However, the persistence layer posed challenges to this because after successfully persisting the object, they could not be recovered again to set up the component when the application is restarted. A careful investigation showed that this was caused by object references which got lost during model updates. This is due to the fact that ActionScript passes objects by reference only. To pass an object by value, one must find a way to clone the object. Cloning is also not well implemented in ActionScript so to clone properly; one must create a function that transfers the properties of one object to a newly created object so as to break all references to the former. However, this does not still solve the persistence problem because during such cloning, if a property of the former object is an object itself, copying the data still maintains a reference to the former object. Eventually, the concept of persisting data using objects had to be abandoned and a safer approach was taken. The alternative approach involved persisting data as arrays.

5.5 Chapter Summary

@TODO Write later

# 6 Evaluation and Analysis

This chapter describes the evaluation of the design of the system with focus on evaluating the PDP component. The purpose of the evaluation is to provide answers to the questions raised in section (@TODO Which section?). @TODO maybe talk about it more here....

## 6.1 Evaluation Method

Formative evaluation was used to evaluate the system at different stages. The main purpose of formative evaluation is to strengthen or improve the object being evaluated (RMKB, 2006). It is used in instructional design to assess ongoing projects in order to implement improvements. Formative evaluation can use a variety of techniques. In this case cooperative evaluation was used.

Cooperative Evaluation is a procedure for obtaining data about problems experienced when working with a software product, so that changes can be made to improve the product (Monk et al, 1993). Cooperative evaluation is a variant of think aloud evaluation. During a ‘think aloud’ evaluation session, the user performs a number of tasks and while doing that, ‘thinks aloud’ (i.e. talk while performing the actions) explaining what he is doing at each stage of the task and possibly why. The cooperative evaluation technique encourages design teams and users to collaboratively identify usability issues and their solutions. By encouraging the user to see himself as a collaborator in the evaluation rather than a subject, a more effective form of evaluation is achieved, thus increasing the utility of the data about problems experienced when working with the product. It is very important to note that in cooperative evaluation, the evaluation subject is the product not the user.

The main activities involved in preparing and running a cooperative evaluation session is as follows:

1. Recruit users
2. Prepare tasks
3. Interact and record

The process begins by recruiting users. The users evaluating the product should be representative of the target population that will eventually use the product. This ensures that useful information is gathered from the evaluation. The number of users that will evaluate the product is also determined. After this, tasks to be performed by the users are prepared. Selecting the right tasks is crucial for the success of the evaluation. Some points to consider when preparing the tasks are as follows (Monk et al, 1993):

* The tasks should be specific. For instance, ‘perform the normal tasks you usually do with this system’ is not a specific task. ‘Select a rectangle tool, drag across the screen to draw a box’ is a specific task.
* The tasks should be representative of the real tasks users of the product will perform
* The tasks should explore the prototype thoroughly
* The tasks should be doable by the users taking part in the session. This also implies the tasks should be clear enough to be understood by them.
* The tasks should be time bound i.e. there should be an estimated time of completion for each task.

The evaluator can also prepare extra tasks that can further explore a part of the product. These can come in handy if the user finishes the specified tasks before the allocated time. The last part of the session involves interacting and recording.

The ‘interact and record’ session id further divided into four parts:

1. Before the users arrive
2. When the users arrive (before starting the tasks)
3. While the users are using the system
4. Debriefing the users

Before the users arrive, everything should be put in place and fully functional. The prototype to be evaluated should be set up in a preferably quiet environment. The tasks to be performed should be ready on a sheet of paper. Some means of recording the session such as a video recorder should be set up and tested. In the absence of a video recorder, a notebook can be used. The questions to be asked during the debriefing session should be prepared. If questionnaires are to be used, they should be ready and kept close by.

When the users arrive, they should be put at ease before beginning the evaluation. The session is meant to be conducted in an informal manner; as such the users should be encouraged to see themselves as co-evaluators not as experimental subjects.

While the users are using the system, they should be kept talking. The evaluator must ensure he knows what is happening at every stage and should prompt the user to ‘think aloud’ if he is silent for a while. Questions such as the following can be used to prompt the user to speak up during the tasks (Monk et al, 1993):

* How do we do that?
* What do you want to do?
* What will happen if.....?
* What has the system done now?
* What is the system trying to tell you with this message?
* Why has the system done that?
* What were you expecting to happen then?
* What are you doing now?
* What were you expecting to happen then?

If the user gets stuck on a task and it seems he cannot perform the task, the evaluator should assist the user or encourage him to proceed to another task.

When the user has finished the tasks, the debriefing session begins. This session can also be recorded for analysis. The purpose of this session is to get more feedback from the user. Apart from discussing about the product that was evaluated, the evaluation process can also be discussed. The evaluator can ask the user questions such as:

* What do you think was the best and worst part of the product?
* What needs to be changed or improved upon urgently?
* How easy did you find the tasks?
* What questions do you have about the product?

Some very interesting comments usually emerge from the debriefing session hence the need to record the session.

Cooperative evaluation is inexpensive and can be done anywhere provided the required resources (the prototype to be tested, the evaluation tasks and recording materials) are present. The results of cooperative evaluation are always very useful when it is well performed. This is because the users do not see themselves as experimental subjects; rather, they consider themselves co-designers or developers. The feeling of having something to contribute to improve the product elicits useful responses from the users thus making the process a very productive one.

## 6.2 Participants, Tasks Requirements and Ethics

Cooperative Evaluation can be tasking and is not something that should be repeated. It is better to get it right the first time. Selecting participants and preparing the tasks are important parts of the process that must not be under-emphasised. Also, an evaluation that requires the participation of human subjects must conform to some ethics to ensure WHAT????

### 6.2.1 The Participants and Task Requirements

The participants for the evaluation were drawn from different areas of studies such as the Business school, Engineering, and the Humanities. The number of participants from computer science was minimised due to their technical background. A total of 10 participants participated in the cooperative evaluation of the software comprising 7 female and 3 male postgraduate students.

A set of representative tasks were prepared for them to perform after which a questionnaire was given to them to fill. The list of tasks as well as the questionnaires can be seen in Appendix A (@TODO which appendix really?). The evaluation generally lasted between 20 - 30 minutes per participant and was carried out in the PEVE lab in Kilburn Building, School of Computer Sciences which was appropriately set up to meet the requirements of the method being used.

The tasks that were set up for the participants to perform required some form of input from the participants. In order to ensure that the participants spend time evaluation the system rather than thinking of what to input, the required data was provided for each participant. Each task also had an estimated completion time to ensure that all the tasks fit in the target 20minutes estimated for the whole session per participant. The “time on task” (TOT) is the time required to complete a given scenario or task. The TOT was carefully set for each task to ensure 100% completion rate for the whole session. However, this turned out not to be so. Some participants had other non-anticipated difficulties during the process leading to some of the tasks extending beyond the estimated TOT.

### 6.2.2 Research Ethics

Research Ethics are “very simple rules of conduct, recognised by certain organisations, which can be applied to aspects of Computer Science evaluation” (Harper, 2010). Research ethics ensure that the evaluation is useful and not a waste of the participants’ time. It also ensures that certain ethical guidelines are followed in conducting the evaluation. The evaluation sessions were required to adhere to the following ethical guidelines:

* The performance of any participant must not be individually attributable.
* The focus of the evaluation must be on the software and not the knowledge or performance of the participant
* The participant's name should not be used in reference outside the evaluation session.
* The participant can leave at any time during the evaluation session if s/he deems it necessary to do so.
* The participant should willingly give his/her consent with regards to the method to be used in recording session.
* Where the session is recorded, the recording must be used only for the purpose of the research and nothing else.

A consent form was given to each participant at the beginning of the evaluation session detailing the ethical guidelines of the session. Each participant was also required to give consent to the form of recording allowed during the session. The forms of recording made available for the session included

* Pen and paper
* Computer screen capturing
* Audio only recording
* Video only recording
* Audio and Video recording

The ethical consent form can also be seen in appendix A (@TODO sure?)

## 6.3 Evaluation Results

The results of the evaluation sessions for the individual user tests as well as the collaborative user test are described below.

General Results

* All the participants agreed that it was easy for them to find their way around the application. Three participants said “the application was self explanatory”. When asked to expatiate, they explained that since the application is consistent with the way components are represented, hence, it is easy to deduce the usage of other components after using one component. One participant remarked that she is up to 75% confident that she can use the system without any guidance.
* Four participants agreed that using the system was fun to use unlike the way it should be when using a system for the first time. One participant commended the software noting that much effort has been put to producing a good system. The participant however noted that there’s still room for improvement.
* Five participants found it difficult to locate the resize button for the components. Three participants tried resizing the components from the borders. The general consensus was that they would have preferred resizing from any of the borders as it is done in the windows environment they are conversant with. Resizing via the borders should be implemented for the components as the present method of resizing is counterintuitive.
* All the participants found double clicking on the title bar to change the title somehow difficult. 4 participants double clicked on the title bar icon instead. This suggests that an alternative method of changing the component title should be implemented.
* All the participants found it very easy to locate the basic window parts such as the buttons (minimise, restore, close) and the title bar.
* All the participants found minimising and restoring the component very easy. According to them, the icons are similar to what they are used to. This shows the importance of using familiar representations in the application. One participant however tried to restore a minimised component by dragging.
* Three participants complained that clicking the close button popped up a dialog asking them about deleting the component. This they found confusing as their intention was to close, not delete the component. Either the dialog box prompt or the functionality of that button should be changed to avoid confusing the users.
* It was easy for all the participants to find the icons representing the components. About 50% of the required components were located using the pictures, others were found using the tooltip. This shows the importance of using the right icons and the need to maintain the tooltips.
* Five participants were able to locate the edge icon based on the description given. One participant wasn’t able to do this. When asked if they would have been able to locate the edge icon in the absence of the description given, they all answered in the negative. This shows the need for a help facility that describes the component parts in details.
* Four participants frowned at the name “whiteboard” for the area where the components are displayed. They suggested the name be change as the area in question was not even white. One of the participants suggested the name “media workspace”.
* Adding components to the whiteboard which was expected to be a very simple task turned out to be a bit tricky. Four participants complained that the system did not make it obvious to the user that a component has been added to the whiteboard when the icon is clicked. General observations also showed that all the participants added a component more than once on at least one occasion. This shows the need to make it more obvious to the user that the selected component has been successfully added to the whiteboard. One of the participants also tried dragging the component icon from the toolbox to the whiteboard. This method of adding a component to the whiteboard is currently not supported but may be implemented as an alternative method.
* Two participants complained that the usual help facility was missing from the system. One of them even attempted bringing up the help feature by pressing the ‘F1’ functional key. Both users suggested that such facility be made available on the system.
* Two participants commented that the tooltips (help texts that appear when the user places the mouse icon over an item briefly) are very useful. They suggested that the feature be maintained and more information added to make it more useful.
* One participant complained that ‘*right click*’ doesn’t bring up the usual menu he is familiar with. This is a limitation in Flash run time but should be looked into. A way of enabling the basic functionalities users require when they right click should be implemented.
* The task sheet had an instruction that expected the user to add a video component. The participants complaint about the fact that the toolbox said YouTube not video component. This is an indicator that the video functionality has to be merged into one component that plays from different sources.
* Three participants agreed that the system is memorable while one participant disagreed to this.
* One participant also commented that the toolbox seems to be cluttered with icons and as such another way of managing the icons should be investigated.
* stated add video, the component is YouTube video which means the video functionality has to be merged to play different kinds of videos
* GEN: System was complex for the first use (x1). No (x1)
* GEN: Toolbox seems cluttered x1
* GEN: System is memorable (x1). No (x1)
* (x1)
* Close button easy to find (x2)Edge icon difficult to locate (x1)

Suggestion section:

One way to make it obvious that a component has been added to the whiteboard is to highlight the component by its borders using a bright colour such as orange. A flashing border might appear on the component until the user activates focus on the component after which this blinking border disappears. An alternative way is to use a speech-icon message close to the toolbox to inform the user of the added component. A much simpler way might just be to add the component very close to the toolbox where the user might immediately notice it.

PDP Component Results

When performing the step ‘minimise component’, xx users did not require help to identify the minimise button. This shows that the button is intuitive and should be kept. Xx users however pointed the button and waited a while to see the tooltip before clicking it. This shows that the tooltip is useful and should be kept. It also underscores the need to make the tooltips more explanatory.

Edge Link = Hold down and drag to link this window should be to link this component.

Edge Arrow when pointed is highlighted -> Good, it should also show a tooltip message about what can be done to it.

Copyright to uploaded audio should be included in the dialog box.

Use the Evaluation **SUS** to score the system for Usability. See downloaded template

Subjective evaluations regarding ease of use and satisfaction will be collected via questionnaires, and during debriefing at the conclusion of the session. The questionnaires will utilize free-form responses and rating scales.

6.2.4 Limitations

6.4 Critical Analysis (I think it should be analysis b4 evaluation bro!)

6.5 UI and UX Analysis

6.6 Summary

7 Reflections and Conclusion

Learning spaces did this did that?

Designing learning spaces needs theory and understanding.

This section reviews the research questions and describes the achievements made. Areas of future work are also identified. Finally, conclusion is given.

7 Research Questions

7.2 Summary of Achievement

7.4 Areas of Further Research Future Work

7.3 Reflection and Approach Evaluation

7.5 Conclusion

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

THINK ABOUT IT, MAYBE INCLUDE IT SOMEWHERE:

RTMP AND SECURITY or SECURITY IMPLEMENTATION IN THE SPACES. Scalability too? Maybe

<http://livedocs.adobe.com/flashmediaserver/3.0/docs/help.html?content=04_overview_security_15.html>

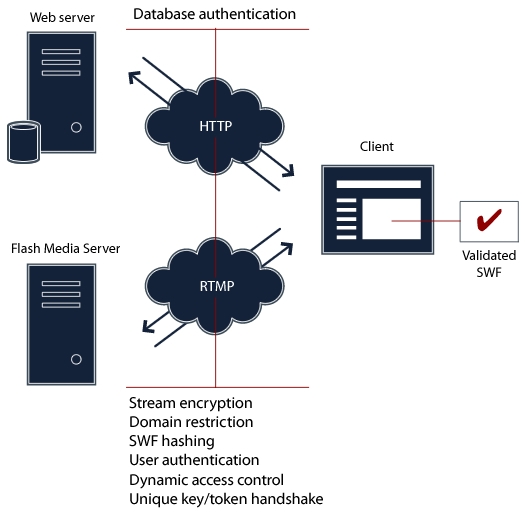
<http://www.adobe.com/devnet/flashmediaserver/articles/scaling_securing_fms3.html>

Future Work:

Consider using RTMFP to do direct peer to peer connection

<http://flashrealtime.com/tuts/p2p-in-flash.html>

<http://labs.adobe.com/technologies/cirrus/> nice images but search adobe more for details



## 3.1 Project Description, Delivery Strategy & Evaluation

This section identifies what the project involves, what will be done, how it will be done, what tools will be used and methodologies that will be applied. Expected deliverables and evaluation criteria are also discussed.

### 3.1.1 Project Description

As discussed in section 1.1 and 1.2, this project involves “improving” the User Interface (UI) for a multimedia, multi-user, collaborative environment. Here, the product to be improved is the “learning space” in a Personal Learning Environment (PLE). The approach will involve taking the existing system and evaluating it by gathering user/expert feedback through cooperative evaluation technique. This feedback will be used to improve the system. Additionally, the feedback gathered will be merged with:

* The Product Owner’s vision for the system
* My own visions for the system and
* Ideas from other domain experts

This process will then be repeated in a series of iterations that will lead to an improved and usable product to be delivered at the end of the project.

Improving the usability of any system is not a one off task; rather, it is a process. It involves using a Usability Evaluation Method (UEM) to assess the usability of the system then applying usability design principles and feedback to improve the system. In this case, technical/functional usability will measure aspects such as navigation, error handling and recovery, user interface layout, accessibility, etc. Good Pedagogical usability is also required.

Since learning spaces foster private construction of knowledge as well as collaborative learning, some interesting questions pop up with regards to making the system more usable. Some of which include:

* What is the best way to manage a user’s private space and shared space?
* In a shared space, how do we accommodate changes and undo requests given that several users may be working on the same learning resource.
* How do we place emphasis on relevant materials so that the user can immediately locate it on first visit?
* When a user visits a shared or private space and returns to it later, there’s a possibility that certain aspects or resources on the space have changed. How do we highlight these changes to the user?
* How do we make it possible for users to invent new components for use in the learning spaces?
* How do we improve the flexibility and adaptability of the learning space?

Another key area to consider is re-configurability and adaptability. Technically, a learning space in the PLE will simply be a canvas with UI components for user generated content. However, the layout of the interface must enable the learner to interact with the space in ways native to his/her learning style. For instance, the learner should be able to use the space as a private classroom, a presentation wall, a resource aggregation platform, a mind map board, etc.

One area I have been considering is transferring some desirable classroom characteristics to the learning spaces. One good example of this is the principle of a “dimly lit” room during presentation. This can be achieved for instance by applying a soft gradient overlay on the page, excluding the area where the presentation is being made. This gives the impression of a “dimly lighted” room and can be very effective. This has been applied on movie and other multimedia streaming websites such as seesaw (<http://seesaw.com>).

Another characteristic is the principle of “immersion”, the ambience and other environmental properties engender learner immersion, thus making the learner relaxed in the learning space. This can be achieved by applying careful layout design principles and colour blending. The use of “full screen mode” also enhances this concept as it removes competing on screen distractions thereby engendering immersion.

Instant feedback is another learning principle that enhances learning and is strongly tied to the motivation theory. Often when using the web, the user has to wait for a page to load and may lose interest and motivation. The use of Flex4 to design the spaces handles this limitation (see development tools in section 3.2.1).

In social or collaborative environments, violation of “Personal Space” is always a consideration worth taking into account. In traditional classroom design, “designers are usually advised to pay attention to the degree to which students feel crowded in a classroom” (Graetz, 2006). Similar notion can be transferred to the design of the learning space by paying careful attention to the crowding of the user interface with widgets. The guiding principle should be, “if it is not needed at the moment, hide it!” Also, being a virtual space, there is no limitation to the amount of learners that can collaboratively share this kind of social space. However, rather than place a hard limit on the number of learners that can share a space at a time, some other interface design principles can be applied. For example, in chat room comprising many people, it is needless to display the list of participants, instead a collapsible container can be used such that the user can expand it when he wishes to see the list and collapse it again thereafter.

### 3.2.2 Communication & Collaboration Tools

Communication is vital to the success of any project. Interaction and Collaboration are core values of agile and nothing fosters these more than communication. Communication tools have therefore been setup to enable uninterrupted flow of communication among all parties involved. The communication and collaboration media are as follows:

**Google Group**

A Google group (see <http://groups.google.com/>) called “newple” has been set up to enable discussion among all parties involved in the design and development of the PLE. This group ensures that everyone gets access to crucial information on time.

**Skype®**

Skype is a software application that allows users to make voice calls and chats over the Internet (see <http://www.skype.com/intl/en-gb/home>). Calls made between Skype® users are free. I can communicate with my supervisor and other people involved via Skype.

**Scratch Wiki**

The scratch wiki (<http://nymph.cs.man.ac.uk/scratch>) was used to produce my Project summary report, project website and a good part of this background report. The wiki enables me and my supervisor to collaboratively document the project as it progresses. Reports in the wiki can be viewed periodically and used as the basis for defining new pathways in the project.

### 3.1.2 Research Methodology

I will follow a 3 phase methodology approach to ensure I achieve the goals of the project. These phases are as follows:

**PHASE 1**:

This is the “Preliminary Preparation” phase. It includes initial research leading to a broader understanding of the project; identification of related work and tools that can be used to achieve the goals of the project. This process has been ongoing and the submission of this background report brings this phase to a logical conclusion. Next, I will go into a brief period of reflection and evaluation to ascertain that the objectives at this stage have been met. Further discussions with my supervisor will also confirm this and together, we will set the path for progress. The most important artifacts produced in this phase are the project time-plan and the background report. The plan will help me manage the most limited resource which is time while the background report has enabled me to situate my project within a wider research theme thus giving me a broad familiarization with the topic.

**PHASE 2**:

This phase will officially begin after my exams, though some aspects of it are already in progress. I refer to it as the “Design, Development and Testing” phase. The understanding derived from phase 1 will be used as the basis for design and implementation of the specified software. In keeping up with the project time-plan, tools for the design have already been identified. Learning to use the tool is also in progress. I have already developed a very deep understanding of the proposed tools. The pending tasks in this phase include the system design, which will be done iteratively together with iterative implementation, testing and evaluation. To ensure a robust system is developed, Test Driven Development (TDD) will be applied to the development process. The Agile Project Management method will be used to ensure that this part of the project is delivered successfully as described in the next sections.

**PHASE 3**:

The third and final phase of the project which I refer to as the “Report, Review and Submission” phase will cover the writing of the dissertation report. The dissertation report will cover all aspects of the project including some aspects already covered in this background report. The report will also highlight the conclusions drawn from the project, contributions made, deliveries and areas for further research. The figure below presents a graphical overview of the phases and their relationship.

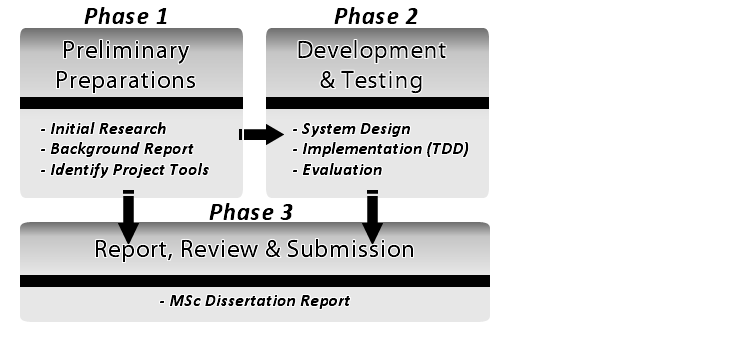


Figure 17 - Research Methodology Flow

Each phase is further split into “**Tasks**”. “**Milestones**” are delivered after a set of tasks. The successful delivery of these milestones on or before the deadlines is a clear indication that the project is on course. A list of the milestones is presented below while the project Gantt chart can be found in the appendices (Appendix A).

**LIST OF MILESTONES**

* Project summary document (due February 17th, 2011)
* Project website (due April 7th, 2011)
* Background report (due May 9th, 2011)
* Periodic evaluation reports (see formative evaluation in section 3.1.4)
* Dissertation report (due September 9th, 2011)

The total time available for this project is 8 months starting from February 1st 2011 and ending September 9, 2011 including the examinations between May 19th and June 8th. As at the time of submission of this report, 50% of this period would have already been gone. It is therefore very important that the tasks in each phase described above progress as outlined in order to ensure I meet the deadline and also deliver a good product fit for a Masters’ Project.

### 3.1.3 Evaluation Plan

This project will be considered successful if all the phases outlined in the research methodology section are completed within the specified time allocated to them. Generally, this will mean that the target goals have been achieved and the expected deliverables delivered.

The type of evaluation being used is formative evaluation. The main purpose of formative evaluation is to strengthen or improve the object being evaluated (RMKB, 2006). It is used in instructional design to assess ongoing projects in order to implement improvements. Formative evaluation and refinement has been ongoing and will continue throughout the process. Formative evaluation can use a variety of techniques. In this case cooperative evaluation (Monk et al, 1993) is being used.

Cooperative evaluation is a variant of think aloud evaluation. The technique encourages design teams and users to collaboratively identify usability issues and their solutions. By encouraging the user to see himself as a collaborator in the evaluation rather than a subject, a more effective form of evaluation is achieved, thus increasing the utility of the data about problems experienced when working with the product. It is very important to note that in cooperative evaluation, the evaluation subject is the product not the user. Cooperative Evaluation will ensure early feedback which will be used to iteratively improve the design and usability of the learning spaces.

The final dissertation report will be evaluated by applying the university research guidelines and regulations to ensure it meets the high standards of the university.

## 3.2 Project Tools

The project has been described, plans have been made but without the availability of the necessary tools, the expected deliverables cannot be met. This section takes a look at the tools that will be used in the project.

## 3.3 Project Management

This section briefly describes plans put in place to properly manage the project in order to ensure its successful completion. A project management plan is relevant to ensure that the project is successful. The Agile Project Management Plan discussed below will be used.

### 3.3.1 Agile Project Management

Agile project management adapts the ideas from “agile software development” into project management. According to the Agile Manifesto and as seen in practice during one of my course modules, agile methods generally promote collaborative development process that encourages interaction, individuality, stakeholder involvement, feedback, incremental and evolutionary development, periodic delivery of working products, effective control and quick response to change. Agile project management will be used to ensure that the project is managed in an incremental and evolutionary manner until it is successfully completed.

### 3.3.2 Risk Management & Issue Resolution Plan

Every project is prone to risk. Identifying, analyzing, prioritizing, and controlling project risks are important factors that must not be overlooked. Since I will be using the iterative development method, at the end of each iteration, I will be able to identify risks during the iteration review. I will record the risks identified and develop plans to handle them. Risk control however, is more desirable than risk management; it is better to prevent risks than manage them. Agile methodologies such as scrum are designed to control risk as much as possible. Daily stand up and sprint planning meetings ensure that risks are reduced to the barest minimum.

# Chapter 4: Conclusion

In designing learning spaces, the architect uses the landscape as his canvas while the technologist uses electronic devices. What they do not seem to realise is that to the learner, the learning spaces so designed, becomes his own canvas upon which he constructs his learning. What makes a learning space useful and usable? This is the question to bear in mind as I progress in this research project.

The “Net Generation Learners” are evolving learning in very unpredictable ways. When creating learning spaces for this breed of learners, “you can’t be sure how these spaces will be used. You are just creating the opportunities for things to happen” (Tom Finnigan cited in JISC, 2006). Their fidelity to technology has become the norm rather than the exception. These learners “appear to have no fear of technology” (Oblinger, 2006a). “Mobile phones, digital cameras, and MP3 players constitute today’s backpack. Browsing, downloading, and messaging” is their way of life (Oblinger, 2006a).

According to Oblinger (2006b), “the Internet has changed our notion of place, time, and space” (Oblinger, 2006b). Nevertheless, the “students’ comfort with the Internet means it isn’t ‘technology’ to them—it may be a way of life” (Oblinger, 2006a). Consequently, learning spaces are no longer just physical, virtual spaces have come of age. Trends in Information and Communication Technology will continuously redefine the meaning, boundaries and styles of learning. But whereas technology come and goes, the psychology of how people learn is more persistent (Brown & Long, 2006). Therefore, we must return to the pedagogical roots (learning theories and styles) in order to be able to blend technology, learning goals and today’s learners’ in the right learning spaces.

Oblinger (2006b) recommends involving the users in learning space design. This includes “students, faculty, and staff”. This is a vital point because while architects and developers see the complexities involved in realising a design, learners do not. What they see is the kind of learning they wish to have in the environment of their choice. Thus it becomes a challenge to the architect and developer to bring to reality the wishes of the learners. In the case of this project, that architect is the software developer who creates virtual spaces for learners. Exactly how we bring together space, technology, and pedagogy to create usable learning spaces is the focal point of this project. Through this project, I hope to be able to contribute significantly to the support of students who learn in the Manchester PLE’s virtual learning spaces, and to generalize from this to contribute to knowledge about “usability in user generated learning spaces”.

# List of References

Adobe® Systems Incorporated, 2011. *What is Flex*? [online]. Available at: <http://www.adobe.com/products/flex/>. [Accessed 25 April, 2011].

Brown M., 2005. "Learning Spaces". In: D. Oblinger and J. Oblinger, eds. *Educating the Net Generation*. Boulder, Colo.: EDUCAUSE, 2005. Ch. 12.

Brown, M. and Long, P., 2006. "Trends in Learning Space Design". In: D. Oblinger, ed. 2006. *Learning Spaces.* Boulder, Colo.: EDUCAUSE. Ch. 9.

Chism N. V. N., 2006. "Challenging Traditional Assumptions and Rethinking Learning Spaces". In: D. Oblinger, ed. 2006. *Learning Spaces.* Boulder, Colo.: EDUCAUSE. Ch.2.

Chowdhury, M. S., 2006. Human Behavior In The Context of Training: An Overview Of The Role of Learning Theories as Applied to Training and Development. *Journal of Knowledge Management Practice*, 7(2).

Crawford, K., 1996. Vygotskian approaches to human development in the information era. *Educational Studies in Mathematics*, (31), pp. 43-62.

DeVries, R., 2002. *Developing Constructivist Early Childhood Curriculum: Practical Principles and Activities*. New York, Teachers College Press.

Dillenbourg, P., Schneider, D., and Synteta, V., 2002. “Virtual Learning Environments”. *Proceedings of the 3rd Congress on Information and Communication Technologies in Education*, Rhodes, Kastaniotis Editions, Greece, pp.3-18.

Domjan, M., and Burkhard, B., 1993. *Domjan and Burkhard's The principles of learning and behavior*. Pacific Grove, Calif, Brooks/Cole Pub. Co.

Fortin, C., & Rousseau, R. (1989). *Psychologie cognitive: une approche de traitement de l'information*. Sainte-Foy, Québec, Télé-université.

Fritscher, L., 2011. *Cognitive Theory - Definition of Cognitive Theory* (Reviewed by The Medical Review Board). [online]Available at: <http://phobias.about.com/od/glossary/g/cognitivethedef.htm> [Accessed 16 March 2010].

Fruchter, R. and Emery, K., 1999. Teamwork: assessing cross-disciplinary learning. In: C. M. Hoadley and J. Roschelle, eds. *Proceedings of the 1999 conference on Computer Support for Collaborative Learning (CSCL '99), International Society of the Learning Sciences,* Article 19.

Graetz K. A., 2006. "The Psychology of Learning Environments". In: D. Oblinger, ed. 2006. *Learning Spaces.* Boulder, Colo.: EDUCAUSE. Ch. 6.

Hertzum, M., and Jacobsen, N. E., 2001. The Evaluator Effect: A Chilling Fact about Usability Evaluation Methods. *International Journal of Human-Computer Interaction*, 13(4), pp. 421-443.

Illeris, K., 2007. *How we learn: learning and non-learning in school and beyond*. London, Routledge.

International Organization for Standardization, 2008. *ISO 9241-151: Ergonomics of human-system interaction - Part 151: Guidance on World Wide Web user interfaces*. Geneva: ISO.

Joint Information Systems Committee (JISC), 2006. *Designing Space for Effective Learning: A Guide to 21st Century Learning Space Design* (Bristol, U.K.: Higher Education Funding Council for England, 2006), Available at: <http://www.jisc.ac.uk/uploaded\_documents/JISClearningspaces.pdf> [Accessed 15 March, 2011]

Jonassen, D. H., and Land, S. M., 2000. *Theoretical foundations of learning environments*. Mahwah, N.J., L. Erlbaum Associates.

Kim, T. and Axelrod, S., 2005. Direct Instruction: An Educators’ Guide and a Plea for Action. *The Behavior Analyst Today*, 6(2), p. 111.

Learning Theories Knowledgebase (LTKB), 2011. *Behaviorism.* [online] Avaliable at: <<http://www.learning-theories.com/behaviorism.html>> [Accessed 14 April 2011].

Learning Theories Knowledgebase (LTKB), 2011a. *Social Development Theory (Vygotsky).* [online] Avaliable at: <<http://www.learning-theories.com/vygotskys-social-learning-theory.html>> [Accessed 14 April 2011].

Leonard, K., Noh, E.K., and Orey, M., 2007. “Introduction to Emerging Perspectives on Learning, Teaching, and Technology”. In: M. Orey, ed. *Emerging Perspectives on Learning, Teaching, and Technology*. [online] Available at: <http://projects.coe.uga.edu/epltt/> [Accessed 15 March 2011].

Marx, M. H., 1970. *Learning: Theories*. New York, Macmillan Co.

Monk, A., Davenport, L., Haber, J., & Wright, P., 1993. *Improving your human-computer interface: a practical technique*. New York, Prentice Hall.

Neill, S., and Etheridge, R., 2008. Flexible Learning Spaces: The Integration of Pedagogy, Physical Design, and Instructional Technology. *Marketing Education Review*, 18(1), pp.47-53., Available via Business Source Premier, EBSCOhost, [Accessed 17 April 2011].

Nielsen, J., 1993. *Usability engineering*. New York, Academic Press, Inc.

Nokelainen, P., 2006. An empirical assessment of pedagogical usability criteria for digital learning material with elementary school students. *Journal of Educational Technology & Society*, 9(2), pp.178–197.

Norman, D. A., 2000. *The Design of Everyday Things*. London, The MIT Press.

Oblinger D., 2006a. "Space as a Change Agent". In: D. Oblinger, ed. 2006. *Learning Spaces.* Boulder, Colo.: EDUCAUSE. Ch. 1.

Oblinger D., 2006b. "Learning How to See". In: D. Oblinger, ed. 2006. *Learning Spaces.* Boulder, Colo.: EDUCAUSE. Ch. 14.

Oblinger D., and Oblinger J., 2005. "Introduction". In: D. Oblinger and J. Oblinger, eds. *Educating the Net Generation*. Boulder, Colo.: EDUCAUSE, 2005. Ch. 1.

Research Methods Knowledge Base (RMKB), 2006. *Introduction to Evaluation*. [online]. Available at: <<http://www.socialresearchmethods.net/kb/intreval.htm>> [Accessed 04 May, 2011]

Richardson, V., 2003. Constructivist Pedagogy. *Teachers College Record*, 105(9), pp.1623 – 1640.

Shneiderman, B., 1980. *Software psychology: human factors in computer and information systems*. Cambridge, Mass, Winthrop Publishers.

Standridge, M., 2002. “Behaviorism”. In: M. Orey, ed. *Emerging Perspectives on Learning, Teaching, and Technology*. [online] Available at: <http://projects.coe.uga.edu/epltt/> [Accessed 15 March 2011].

Ughade A., Raffin E., Bouteiller G., Bell I., Cocon P., 2007.*Personal Learning Environment: Design Factors and Implementation*. Msc. The University of Manchester.

van Harmelen, M., 2010. *Short introduction: the Manchester PLE*. [online]. Available at: <http://www.youtube.com/watch?v=Ooy2xR9YcBk> [Accessed 04 May, 2011]

van Harmelen, M., 2011. *Discussion on Usability in User Generated Learning Spaces*. [email] (Personal communication, 27 April 2011).

Vygotsky, L. S., and Cole, M., 1978. *Mind in society the development of higher psychological processes*. Cambridge, Mass. [u.a.], Harvard Univ. Press

Wortham, S., 2003. Learning in Education. *Encyclopedia of Cognitive Science*, Volume 1, Article 563, pp.1079-1082.

Zaharias, P, and Poylymenakou, A., 2009. Developing a Usability Evaluation Method for e-Learning Applications: Beyond Functional Usability. *International Journal of Human-Computer Interaction*, 25(1), pp. 75-98. Available via Business Source Premier, EBSCOhost, [Accessed 11 April 2011].

# Appendix A: Project Plan Gantt Chart

