Pedagogical Evaluation of a Systemic Soft Domain-Driven Design Framework

Mohammed Salahat
Ajman University of Science and Technology, UAE
m.salahat@ajman.ac.ae &
*University of Huddersfield, UK,*u04223855@hud.ac.uk
Steve Wade
Informatics Department, School of Computing and Engineering, University of Huddersfield, UK
s.j.wade@hud.ac.uk

Abstract
This paper presents a pedagogical evaluation of the framework SDDD as a "soft systems" approach to Domain-Driven Design of computer-based information systems development. The framework combined techniques from Soft Systems Methodology (SSM), the Unified Modelling Language (UML), and an implementation pattern. Systems development and teaching evaluations are done to find better framework which can be used for teaching and developing information systems. More Feedback and reflections from the lecturers and Msc students of the module Methods and Modeling are presented. The results are supported our previous work of proposing the framework to enhance the understanding of the business process modeling and implementation into an integrated framework. This is an enhancement of Domain Driven Design approach because new "soft layer" is added and the framework used for teaching further than development as DDD. Comments received from all participants are used to enhance the framework development and for further evaluation in the future.

Keywords: Peer-Tutoring, SSM, UML, Multimethodology, Soft Domain-Driven Design, Modelling.

Pedagogical Evaluation of a Systemic Soft Domain-Driven Design Framework

Abstract

This paper presents a pedagogical evaluation of the framework SDDD as a "soft systems" approach to Domain-Driven Design of computer-based information systems development. The framework combined techniques from Soft Systems Methodology (SSM), the Unified Modelling Language (UML), and an implementation pattern. Systems development and teaching evaluations are done to find better framework which can be used for teaching and developing information systems. More Feedback and reflections from the lecturers and Msc students of the module Methods and Modeling are presented. The results are supported our previous work of proposing the framework to enhance the understanding of the business process modeling and implementation into an integrated framework. This is an enhancement of Domain Driven Design approach because new "soft layer" is added and the framework used for teaching further than development as DDD. Comments received from all participants are used to enhance the framework development and for further evaluation in the future.

Keywords: Peer-Tutoring, SSM, UML, Multimethodology, Soft Domain-Driven Design, Modelling.

1.0 Introduction

The failure of software support systems has been well documented over the years, and many of these failures have been attributed to poor business process modelling (Barjis, J., 2008). The systems failed because the business process model developed did not adequately support the process of designing and implementing the software support system. One of the main reasons for information systems failure is a tendency to concentrate on the technical aspects of design rather than understanding the business needs (Alter, S., 2007). There is a need for a systematic approach for capturing the information required by business processes (Barjis, J., 2008). This suggests a need to bridge the gap between business process modelling, information systems modelling, and implementation. Our previous work (Salahat et al, 2008), Salahat, M., Wade, S., 2009) proposed and evaluated a development "framework" to deal with soft and technical systems aspects with an emphasis on modelling workflow. The evaluation results guided us to modify the framework in a new direction in which the concept of "workflow" is less dominant. The new modified framework (Salahat et al, 2009), focuses on Domain-Driven Business Process Modelling (DDBPM) as an approach to modelling business processes in an object-oriented domain model. This approach was named SDDD (Soft Domain-Driven Design). SDDD aims to investigate, analyze and model a business domain so that we can implement it as a software support system. SDDD is a multimethodology systemic framework consisting of four phases with guiding procedures to steer the developer between the various compromises that need to be made throughout the development process. This paper gone further steps to evaluate the SDDD framework through teaching process. Peer-Tutoring System re-done as an Msc Project and a feedback from the lecturer of the module Methods and Modelling and the students done the module are presented and used to reflect on the framework as an approach for business domain modeling and implementation teaching and real software development approach. Section 2 reviews related work. Section 3 introduced the teaching of Methods and Modelling module. Section4 is briefed the research methodology used. Section 5 is introduced the framework as a multimethodology approach. Section 6 is a brief description of a practical case study in which the method has been applied. Section 7 presents feedback from the lecturer and students. Section 8 is a recommendations and conclusion.

2.0 Related Work

2.1 Domain Driven Modeling (DDM)

The business domain for any organization accommodates the organization business process that must be well defined and modelled for the implementation. Business domain comprises the business process can be defined as ‘the transformation of something from one state to another state through partially coordinated agents, with the purpose of achieving certain goals that are derived from the responsibility of the process owner’ (D., Platt,1994). There are many definitions of “business process”, and the most of these definitions are based on the idea of a business process as a deterministic system that receives inputs and transforms into outputs following a series of activities. For example (Daveport, T., 1993) defines business processes as “‘‘structured sets of activities designed to produce a specified output for a particular customer or market’’. Business processes are similar in different business domains running the same industry of business. To support the business domain, good information systems software used to support the organization work by handling the internal business process and control all aspects affecting the execution of the process. The business process must be supported with good business process modeling (domain modeling) and implementation techniques that can analyze, model, and implement the business process in a professional way to achieve the organizational goals (Warboys et al, 1999).

2.2 Domain-Driven Design

Domain-Driven Design can be used to model the business process as a business domain model (Evan, Eric, 2004). A Ubiquitous Language (UL) is generated first as a communication tool between different stakeholders and the domain model will be generated and implemented based on UL.

UML diagrams are sufficient tools for requirement modelling to support business process modelling in an object-oriented domain model (Svatopluk Štolfa, Ivo Vondrák, 2008). When it comes to implementing the system we have made use of the DDD implementation pattern (i.e. Naked Objects or True View) to reflect the system interface directly from the domain model. Naked Objects and TrueView Domain Modeller ar used for exploring Business Domains and creating rapid prototypes using Domain Driven Design. It helps you to work with your Domain Experts to understand business entities, relationships and the business' ubiquitous language and to write classes using .NET and the Naked Objects or TrueView framework.

2.3 Soft Domain-Driven Design

Soft Domain Driven Design (Salahat et al, 2009), is an approach that seeks to model the system processes as a domain model and develop a software support system based on it. In DDD Ubiquitous Language was used to create the domain model by the developers and domain experts (Evan, Eric, 2004) and to facilitate the communication between different stakeholders. UML, as a part of SDDD, defines a number of diagrams that can be used to model the business process (Al Humaidan, F.,2006) but lacks the ability to explore the soft issues related to the problematic situation which can be handled using Soft System Methodology. SSM ((Checkland, P., Poulter, J., 2006), (Checkland, P., 1999), and Checland, P., Howell, S.E,1998) is an established means of problem solving that focuses on the development of idealized models of relevant systems that can then be compared with real world counterparts. SSM is used in SDDD to model the business domain using rich pictures, root definition, and conceptual model. In our previous work (Salahat et al, 2009), we have adapted the idea of a Ubiquitous Language into a “Soft Language” which incorporate certain artifacts of a SSM analysis into the model. The first step of the SDDD approach is to develop a ‘Soft Language’ as result of the application of Soft System Methodology. This language is an a compliment of the Ubiquitous Language described in Domain-Driven Design (Eric Evan,2004) which consists of different concepts, diagrams, and documents to facilitate the communications between the developers and domain experts. Some researchers have explored the relationship between SSM and object oriented analysis and design techniques in general (Bustard, D. W et al, 1996) but less has been written about the application of these techniques in the context of the UML. An object-oriented domain model can be extracted from this Soft Language through a transition process from SSM Conceptual Model to UML Use Cases. We argue here that SSM helps the developer to gain a deep understanding of different stakeholders’ perspectives which will need to be represented in the Soft Language. In this paper we argue that this transition supported the students understanding of modeling the business domain and implementing the software support system based on that.

UML is considered by DDD and SDDD to model the business domain as a “Domain Model” with a difference between the two approaches. As described in our previous work (Salahat et al, 2009), SDDD framework guides the developer into creating a “Soft Language” which consists of the output of the SSM stage to deal with the soft aspects which are not handled explicitly by Domain Driven Design. The SSM Conceptual Primary task Model (CPTM) is used to map human activity to a UML use-case model using a new elaboration technique. Use-cases, as abstractions of business activities, are used to model the business process in a domain model using UML diagrams and based on the philosophy of DDD which employs the idea of “Knowledge Crunching” during the different stages. SDDD employs the same philosophy during its four stages as explained in later sections.

The SDDD framework combines SSM, UML techniques, and an implementation pattern either Naked Objects or True view, or others which satisfied the philosophy off DDD and SDDD. In this paper the application of SDDD by reusing the case study Peer-Tutoring System and investigating the lecturer and the Msc students done the module “Methods and Modelling” using SDDD framework. The implementation part focused on using True View and Naked Objects Implementation Patterns. To the best of our knowledge, this combination has not been applied in an intervention before, and an evaluation in teaching context and the application in business projects will be a contribution to this domain of research and software development.

2.4 Other related works:

Recent works (Wade, S., Hpkkins, 2002) and (Al Humaidan, F., Rossiter, N., 2004) consider the SSM conceptual model as a focal point for linking SSM and UML by mapping the activities of an SSM conceptual model into UML use-cases. Recent examples of this approach can be found in SWfM (Al Humaidan, F., 2006) and our previous works (Salahat et al, 2008), (Salahat, M., Wade, S., 2009), and Salahat, et al, 2009). Other researchers have made use of various extensions to the UML. For example (Sewchurran, K, Petkov, D., 2008) employed a systemic framework combining SSM and UML extensions proposed by (Erikksonn, H. E., & Penker, M., 2000) to model the business process of a manufacturing factory. Their framework is based on Mingers Multimethodology ideas (John Mingers, 2001) but does not encompass the software implementation phase of development.

3.0Teaching methods and modelling

Teaching modeling of business systems using a modeling language like UML will not lead to complete understanding that can help the students or developers to implement a software support system that can combine all the business experts' requirements. We argue that teaching business domain investigation and modelling using an integrated framework can enhance understanding of such problematic situation and may be lead to a substantial software system. Based on this, the module Methods and Modelling in Informatics Department in the University of Huddersfield has been taught to the Msc students using the SDDD framework which combines tools from SSM, UML, and implementation pattern. The approach can be applied in a wide range of situations including requirements analysis for information systems design. Some researchers have explored the relationship between SSM and object oriented analysis and design techniques in general (Bustard, D *et al,* 1996; Lai, L.S. 2000) but less have been written about the application of these techniques in the context of the UML. We argue that used alone UML models can encourage early design decisions before opportunities for improvement have been agreed and that SSM lacks the detailed information required by designers developing domain models. This leads to the conclusion that there could be some advantage in using the techniques together. We expected from using this integration, in teaching systems modeling, that the students will see the whole systematics picture of the business domain and the modeling will be understandable and will lead to a sufficient business domain model for coding the requires software system.

4.0 Research Methodology

This research, as part of on-going research work, aims to answer the following research questions:

1- ***How to model and implement the Business Domain Processes into a Domain-Driven Design System?***
2- ***How the proposed approach, for modelling and implementation, can support the process of teaching the module "Methods and Modelling" for Msc students in Informatics Department?***

Both authors are involved in teaching in their Universities. This encouraged us to use the approach of **Action Research** since we are actors and part of any system in the education environment and for us it is the domain to apply any research. By teaching systems modelling and design for many years by both authors, we found from teaching and the literature review that many software systems failed and the reason of failures because of the tendency to focus more on the technical aspects rather than the Business Domain Processes modeling (Barjis, J., 2008). The majority of software development methodologies initiated from the software engineering science and not given sufficient attention to the business processes modeling for any given business domain. Involving the business experts with the technical people to investigate and model any domain needs a methodology or framework that can be used by different stockholders and facilitate the communication between them. Among this Domain Driven Design is dominant but still the communication depends on the technical system concepts which may be a problem for the business expert to understand. Soft System Methodology is well-established and known as an approach to explore problematic situation. Based on that, this ongoing research suggested the combination between SSM, UML as a modeling language, and an implementation pattern satisfied the philosophy of Domain-Driven Design as a dominant approach among others. The new approach is proposed and published in our previous work (Salahat et al,2009) and further evaluation from the development and teaching perspective taken place in this paper. To answer the above research question and to apply the research as it's designed, the following ***methodology*** followed:

1. Review the current situation of business domain processes modelling through teaching and literature review. A comparison between different approaches done based on that.

2. Formulate and propose a multimethodology framework considering soft and hard business domain aspects.

4. Evaluating the framework through different practical case studies as an undergraduate and Msc projects (Peer-Tutoring System, Work Placement Operations Mgt. System, and University Students Associations System).

5. Reflect on the implementation and record learning from the methodology application in order to guide further applications.

6. Evaluate the framework as an approach of teaching systems modelling for Msc students. This involves getting the feedback from students through interviews and questionnaire. The feedback of the lecturer also considered through an interview.

7- Reflection on the framework as an approach of teaching that support the module aim achievement

5.0 The SDDD Framework

The SDDD framework (Salahat et al, 2009) is briefed here to relate it with the evaluation in order to facilitate the understanding process of the reader. SDDD was developed into an action research intervention based on research of multimethodology, which justifies combining methods for the same business intervention (Minger, J., 2000). It is a multi-method framework which intended to guide the developer through an investigation of a problematic situation. The purpose here is to insure that a comprehensive understanding is achieved in order to facilitate the modelling and implementation of the domain-driven business processes as a software support system. The modeling will produce an object-oriented domain-driven model as the bases of developing the software support system. As mentioned in the previous work (Salahat et al, 2009), the framework was been developed through a series of “action research” case studies. Accordingly our case studies have involved development projects within our own school. In this paper, a peer-tutoring-system re-investigated as an Msc Project to evaluate the SDDD applicability in modeling the business domain processes into object-oriented domain model. The researchers are part of the school and they are participating in the daily activities related to the case studies. They supervised the students and guided them to the final stage of the projects and teaching courses related to business domain modeling and implementation.

The SDDDF Framework (Figure 1) is focused on modelling and implementation of the domain-driven business process as a software support system. SSM is used as a guiding and learning methodology with techniques including UML and implementation pattern (Naked Object or TrueView) embedded within it. The DDD philosophy is adapted to generate a “Soft Language” (SL) as a compliment of “Ubiquitous Language” (UL) and it used as an input to the next stages. The implementation pattern is used after the generation of the final refined change report which is an input to the implementation process.

Using (Minger, J., 2000) generic model which discussed in (Salahat et al, 2009), the SDDD framework consists of four phases and each phase consists of a group of activities. The framework satisfies the generic process of conducting an action research in the business intervention. SDDD represented in Figure1, Figure 2 represents the conceptualization of the framework, and Figure 3 represents the logical processes embedded in it. For more details about these phases refer to our previous work (Salahat et al,2009).

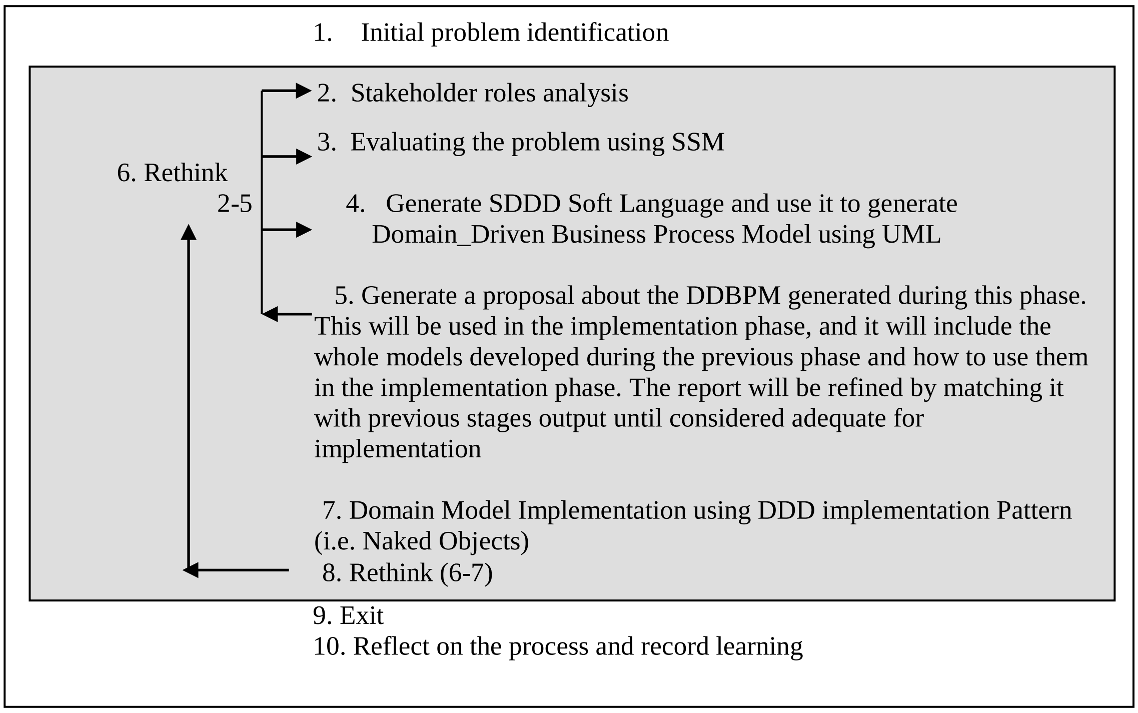

1. Initial problem identification
2. Stakeholder roles analysis
3. Evaluating the problem using SSM
6. Rethink 2-5
4. Generate SDDD Soft Language and use it to generate Domain_Driven Business Process Model using UML
5. Generate a proposal about the DDBPM generated during this phase. This will be used in the implementation phase, and it will include the whole models developed during the previous phase and how to use them in the implementation phase. The report will be refined by matching it with previous stages output until considered adequate for implementation
7. Domain Model Implementation using DDD implementation Pattern (i.e. Naked Objects)
8. Rethink (6-7)
9. Exit
10. Reflect on the process and record learning

Figure 1: A Systemic Soft Domain-Driven Design (SDDD)

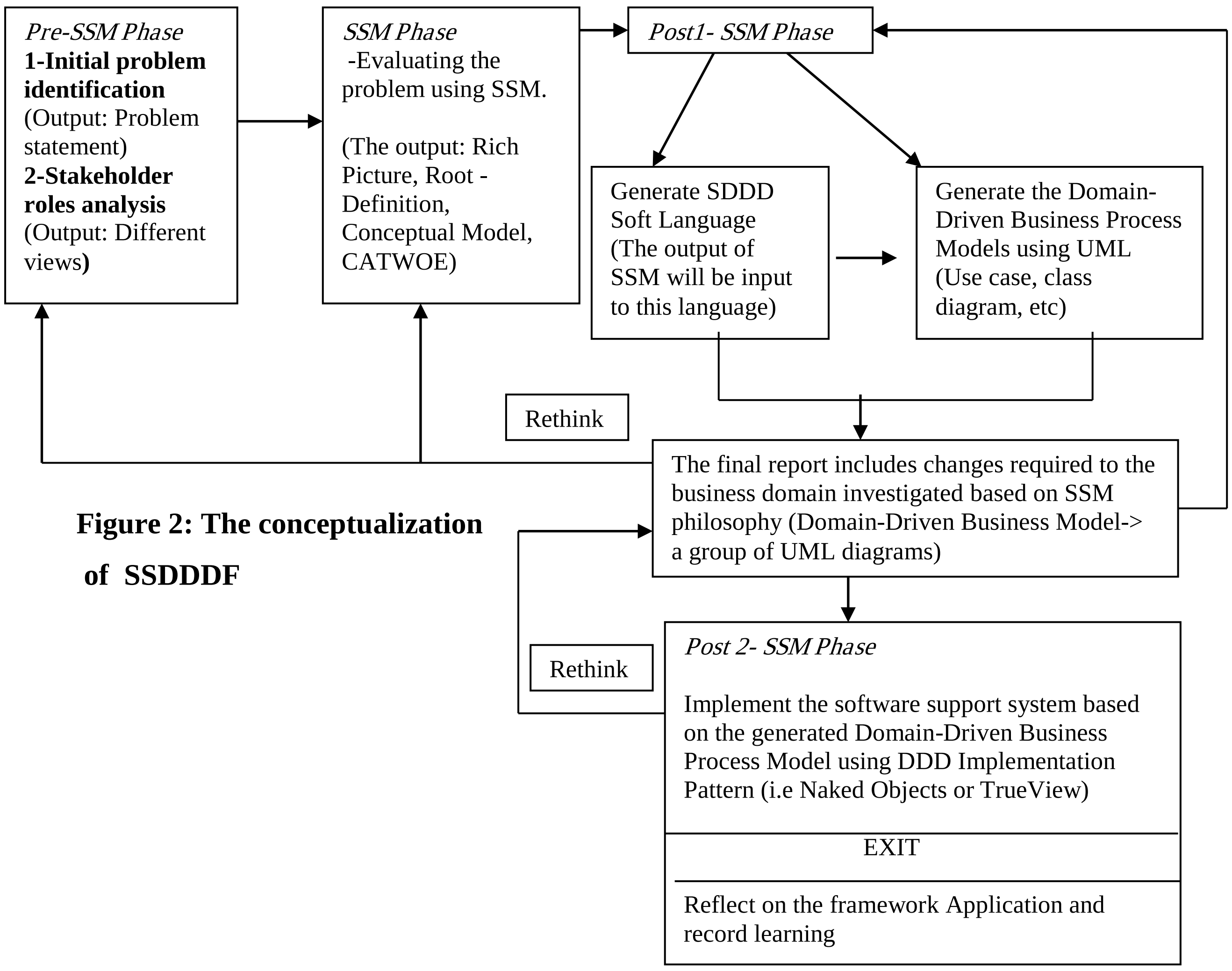

Pre-SSM Phase
1-Initial problem identification
(Output: Problem statement)
2-Stakeholder roles analysis
(Output: Different views)
SSM Phase
-Evaluating the problem using SSM.
(The output: Rich Picture, Root -Definition, Conceptual Model, CATWOE)
Post1- SSM Phase
Generate SDDD Soft Language (The output of SSM will be input to this language)
Generate the Domain-Driven Business Process Models using UML (Use case, class diagram, etc)
Rethink
The final report includes changes required to the business domain investigated based on SSM philosophy (Domain-Driven Business Model-> a group of UML diagrams)
Rethink
Post 2- SSM Phase
Implement the software support system based on the generated Domain-Driven Business Process Model using DDD Implementation Pattern (i.e Naked Objects or TrueView)
EXIT
Reflect on the framework Application and record learning

Figure 2: The conceptualization of SSDDDF

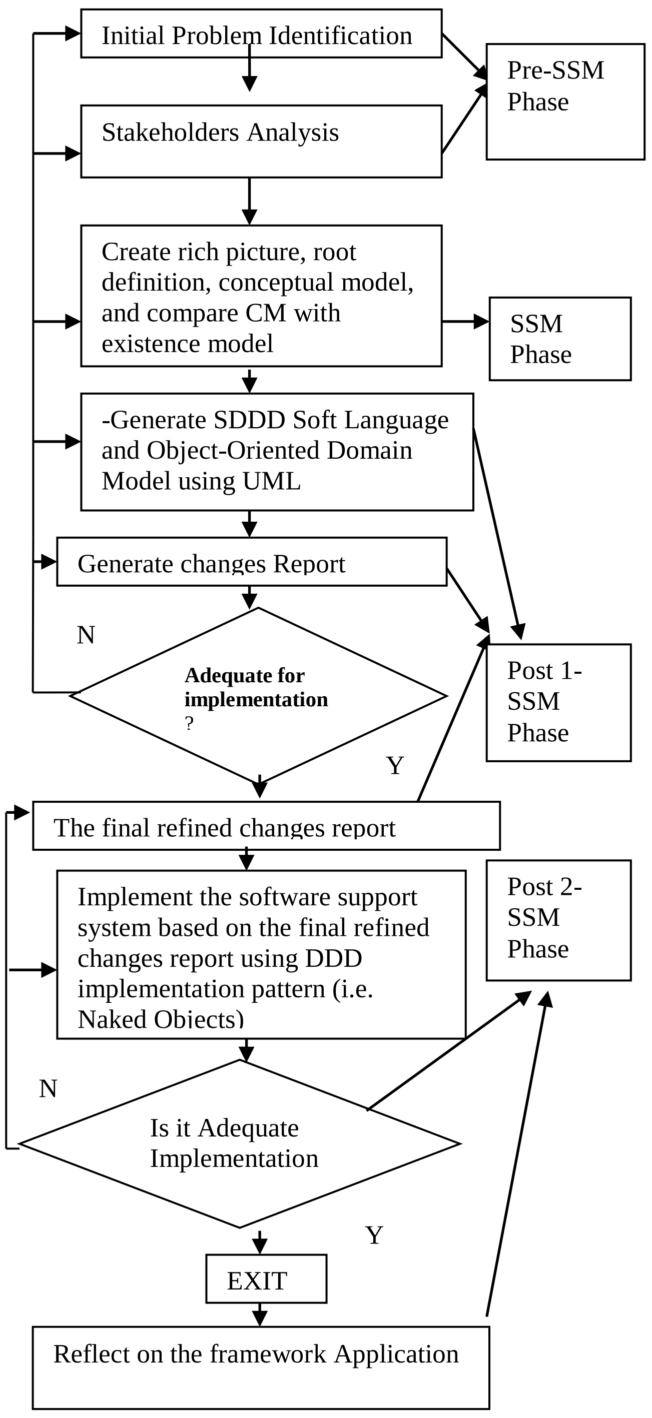

Initial Problem Identification
Stakeholders Analysis
Pre-SSM Phase
Create rich picture, root definition, conceptual model, and compare CM with existence model
SSM Phase
-Generate SDDD Soft Language and Object-Oriented Domain Model using UML
Generate changes Report
N
Adequate for implementation ?
Y
Post 1-SSM Phase
The final refined changes report
Implement the software support system based on the final refined changes report using DDD implementation pattern (i.e. Naked Objects)
Post 2-SSM Phase
N
Is it Adequate Implementation
Y
EXIT
Reflect on the framework Application

Figure 3: The embedded logic in SSDDDF

6.0 The development of Peer-Tutoring-System using SDDD Framework

6.1Peer-Tutoring briefing:

Since we have been engaged in an information systems development project using SSM and UML techniques within an agile framework to make recommendations about the development of an intranet for the academic school in which we are employed. At the beginning of the project the department had an operational intranet but this was not widely used. An information system strategy was initiated to investigate ways in which the intranet could be developed to support the university mission and departmental goals. Initially we used use cases as the primary fact-gathering technique but certain limitations in this approach led us to a more thorough SSM-based analysis of the situation. We argue that the techniques of SSM can help the developer to identify a richer set of use cases than would otherwise be possible but developers with a full use case model still have many challenges ahead of them. We are interested in object oriented design and the view that all business behavior identified in the use case model should be encapsulated as methods on domain objects. Thus, a Student object should not just be a collection of data about the Student; it should encapsulate all the behaviors that we need to apply to a student. In Domain-Driven Design these are often referred to as 'behaviorally-rich' domain objects. A number of software systems required supporting the department and one of these is Peer-Tutoring System at the Undergraduate level proving programming modules. It aims to design and implement peer-tutoring system for introductory programming unit in the department of informatics to support the students and reduce number of failures. One of the current problems facing students and lecturers in university is the difficulty of understanding and mastering the skills required to write and run computer programs successfully. The system has been suggested to be introduced with the reason of improving the pass rate at the University and also increasing the confidence and knowledge in students when teaching each other during the sessions. On the other hand, this system will reduce workload on lectures as time they spend clarifying a point to a single student can be reduce since they will discuss such points at the tutoring session amongst themselves thus leaving the lecture to concentrate on preparing lessons for the next classes. A number of researchers have suggested that peer tutoring can be particularly useful to support this type of learning because it allows learners to learn and support each other (Goodlad, S.,Hirst, B., 1989) and it is beneficial to help students learn and practice the required skills more actively in a setting that encourages them to be more active and intellectually engaged (Gardner, H., 1993). Other researchers (Miliszewska, Tan Grace, 2007) reported about the problems of teaching programming course at Victoria University in Australia and they proposed an approach to enhance the delivery of this module. (Hu Xiaohui, 2006) Raised the difficulties of teaching programming course in Chinese universities and discussed different modern incorporating strategies, to solve this problem, which includes "Concept Mapping", "Peer-learning" and "E-learning" methods. The implementation part aims to build an application that will be used to manage the PTS by allowing Students make bookings for sessions and allow lectures to select tutors and tutees from the results that the students got in their previous year, previous semester or Blackboard quiz results. The tutors will be students in the final year with good grades while the tutees will be in the first or second year and need support to improve their skills. The lectures will also be able to load room availability to enhance the booking process and monitor the progress of the system by monitoring the pass rate if it has increased compared to the previous year without Peer Tutoring System (PTS). The pass mark which determines a final student can qualify for the tutor position will be determined by the management and set as a business rule. The proposed solutions by other researches show how to recap the difficulties of teaching programming unit by concentrating on the delivery methods only without investigating all soft and hard systems issues that can cause such a problem (Miliszewska, Tan Grace, 2007), and (Hu Xiaohui, 2006). SDDD aims to model hard and soft system aspects. In this paper, we developed a Peer-Tutoring System using SDDD framework to support and improve the teaching process. This solution aims to enhance the students understanding which may be reduce the percentage of failures in this module. The development of PTS is presented the next section.

6.2 Peer-Tutoring System Development:

The SDDD framework used to build an application to help managing the system in the sense of scheduling, confirming and cancelling tutoring sessions for undergraduate programming modules. The application developed aims to help the administrator of the PTS in a way that it allows:

- Tutors book tutoring sessions without the help of a lecture, see how much rewards has been allocated to them and also update their diaries to allow tutees see if the tutors are available before making a booking.

- Tutees are able to book tutoring sessions without aid of a lecturer. They can also mark attendance for the sessions they attended to allow lectures and management judge progress of the system as a whole.
- Lecturers are able to load tutee, tutor and room information onto the system. The Lectures are also able to calculate rewards due to a tutor as per sessions they have delivered. Should there be a system failure; the lectures will also need to report them to an engineer to attend to the problem.
- The management is also to see the rewards allocated to a tutor by a lecturer so that they can be redeemed. Policies and Procedures will also be applied to the PTS by the management. Detailed application of the framework presented in the following sub-sections.

6.2.1 *Pre-SSM Phase*

6.2.1.1 The problem identification

It is mentioned in the previous work (Salahat et al, 2009) that the Department of Informatics in the School of Computing and Engineering at the University of Huddersfield in UK and Information Technology College at Ajman University of Science and Technology in UAE both offer introductory programming modules for their first year computing students. These modules focus on Java programming; lecturers face certain difficulties related to students understanding of the subject because of the nature of the required problem-solving skills. Students require more tutoring and practical sessions to help them practice different exercises in order to enhance their understanding and practical skills. Both Universities expect that implementing a peer-tutoring system will reduce the failure rate. The departments want to know how to select tutors among good students and how to reward them. The exact problem identified by working with the students as lecturers and interviewing them about the difficulties. The interviews were conducted with students studying programming modules in the Informatics Department at the University of Huddersfield as these will be the people using the system and students in the IT College in Ajman University in UAE. Feedback of authors located in both mentioned universities were recorded also. Also we interviewed some members of staff at the School of Computer Engineering as these are the people that will allow the system to be used in the department, reward the tutors and apply policy and regulations in the system.

As an action researchers, we conducted the interviews in an informal way and done face to face so that the participants will be feeling comfortable during the interview as they can see who is interviewing them hence also giving their ideas and suggestions comfortably. With some ideas, participants were better explained by face to face interviews for example expressions which may not be fully explained in writing or over the phone. These actions were being noted throughout the interviews and appreciated unlike over the phone that these cannot be appreciated.

The interviews were targeted to collect the following information:

- What is the current system offering?
- How far their lecturers can go in supporting them with their works outside classroom hours?
- Would they need more support on the work outside their lecture hours to increase their comfort in the module and increase skills?
- What do they think of PTS?
- Would they understand better if they were learning from a fellow student who had outstanding grades in the previous year and learn from their experience and achievements?

6.2.1.2 Stakeholder Determinations

The stakeholders defined in this case are the people that will be using the system, and who will benefit from it. The stakeholders of the required PTS system were determined to be peer tutor, peer tutee, lecturer, and management. The stakeholders have different expectations of the system. The different stakeholders of this system expected that they can achieve the following from using it:

- Peer tutors are generally looking for teaching experience to be added to their CVs.

- Peer tutees are looking for extra help.

-Lecturers are looking to reduce their workload, and to determine which students most require tutoring sessions.

-Management looks to reduce the number of failures on programming modules.

6.2.2 *SSM Phase*

6.2.2.1 Investigating the problem situation using a rich picture

A rich picture is a drawing that graphically illustrates the issues expressed by people, processes involved in the transformation, people involved with or affected by the change (stakeholders), working climate, conflicts and structures within the change process (Williams, B.,2005). Rich pictures were used as a tool used in this investigation to express the views of stakeholders and their expectations from the being developed system.

In order to develop a rich picture of the situation under study, a number of information sources were used to capture views of the introductory programming unit from the perspective of the management (the school & the college in both universities), lecturers, and students. Interviews with the school (or college) administration and groups of students were conducted to understand the problematic situation of teaching introductory programming course and set out suggestions to solve the problems. The following Figure represent the Rich Picture of PTS.

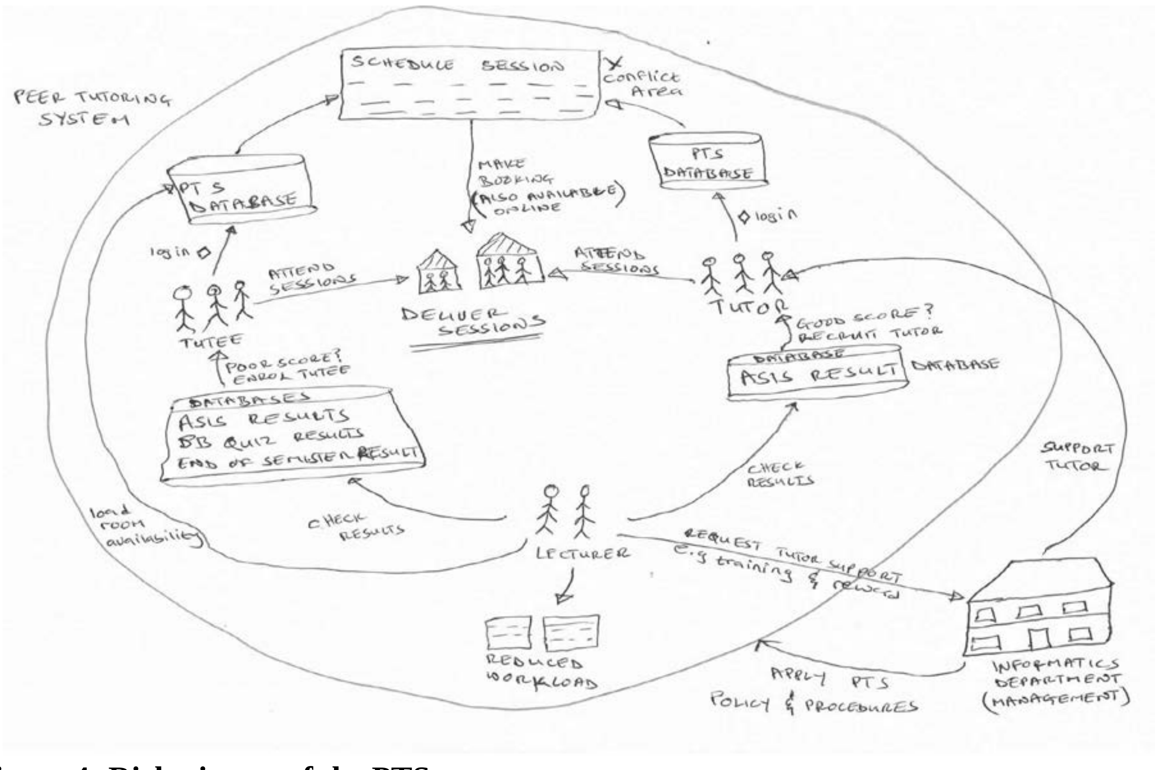


Figure 4: Rich picture of the PTS.

6.2.2.2 Modelling the relevant system using SSM

Modelling the system using Root Definition is described by (Checkland, P., Jim, S., 1990) as a movement from the real world to systems thinking about the real world.

(Williams, B., 2005) mentioned that during root definition stage, points of views from the different stakeholders are drawn out from the rich picture and have them in a structured development process. Root Definition (RD) of the PTS is as follows:

"To propose a peer-tutoring system for the informatics department to help in the selection of peer-tutees and peer-tutors, the scheduling of tutoring sessions based on the availability of rooms, tutors, and tutees, monitoring the perceived benefit to tutors and the progress of tutees in increased self-confidence as well as measure the impact on failure rates."

Root definition has been used to extract the conceptual model which represents the different stakeholder views. So the conceptual model describes activities that might take place if the relevant root definition was to be an accurate representation of the work of a system. The following Conceptual Models (CM) represents the different stakeholders' views, the actions that must be taken based on their views, and also meeting their particular cultural, political and social requirements of the system. All of these issues are expressed in the rich picture and modelled using the conceptual models. The proven issues between different stakeholders presented in a diagram called Conscious Primary Task Model.

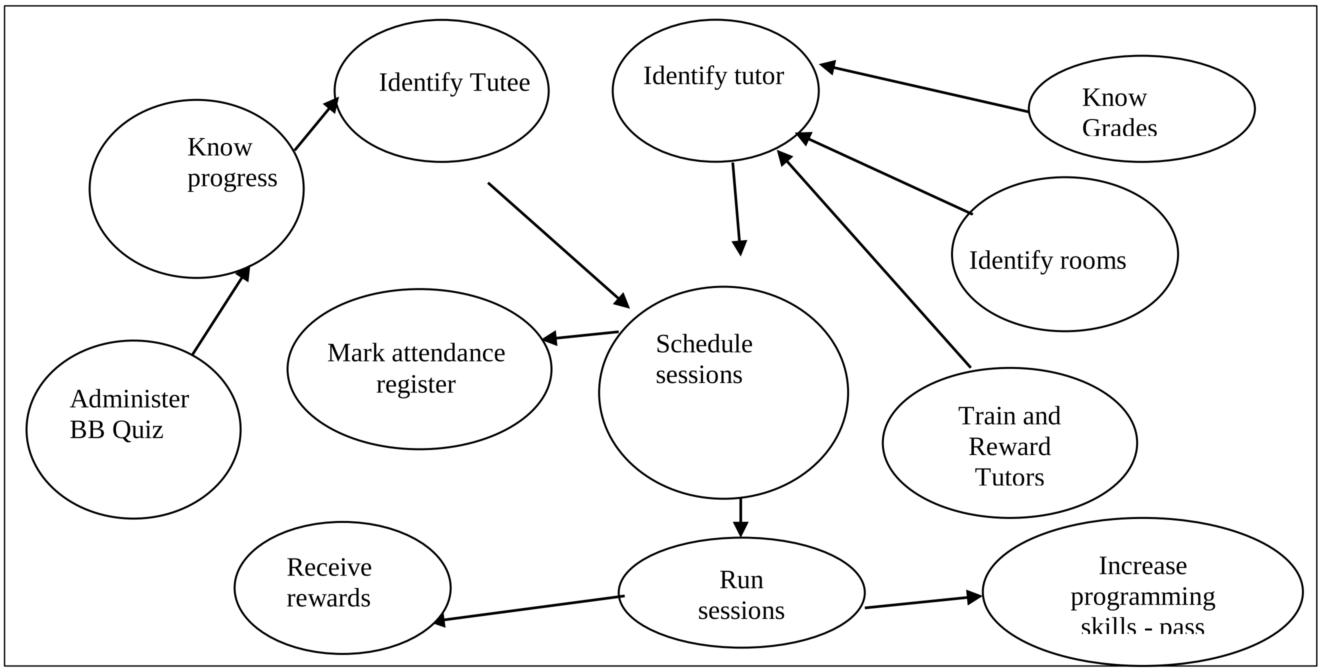


Figure 5: Conscious Primary Task Model (CPTM)

6.2.2.3 Compare the conceptual model to the real world:

SSM required the investigator to compare the produced conceptual model with the actual real life work. There is no real life PTS available to be compared with the developed conceptual model. In this case, the conceptual model will be considered the base to model the PTS system as a domain model. The CPTM, as a combination of all conceptual models, and considering the other components of SL, all will be used in the next phase to generate the domain model as stated in the beginning.

6.2.3 Post1-SSM Phase: Moving from Soft language (SSM Phase) to Domain Model

Domain Model will be represented using UML. Domain modeling starts with the conversion from the Conceptual Model into Use Cases and Use Case modeling. The extracted Use cases will be used to develop UML Sequence Diagram, Class diagram, and Activity Diagrams development. The next subsection will show the conversion from CM to Use Cases.

6.2.3.1 Moving from SSM Conceptual Model UML Use Cases

The conversion process is presented in Figure 9. Any activity required software support will be selected as a use case. The stage of moving from an SSM conceptual model to a use cases l is not as straightforward as this high-level discussion would suggest. In thinking this through we have been pushed towards making a clear distinction between stakeholder goals, business activities and use cases. The following model (Figure 9) shows the relationship between these key abstractions.

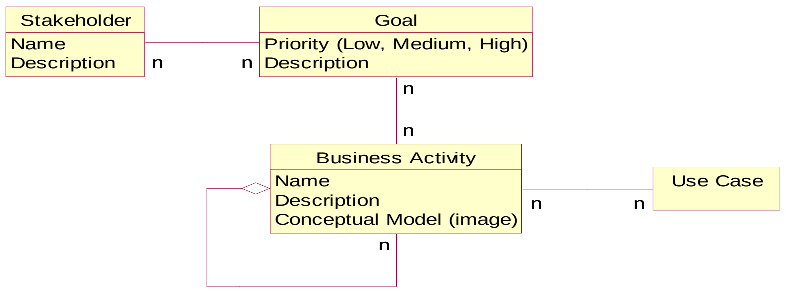


Figure (6): Moving from an SSM to use case diagram

Using the above conversion algorithm, the conceptual models presented above converted into different use case. The following Use Case diagram (Figure 10) presented as a result of the conversion process.

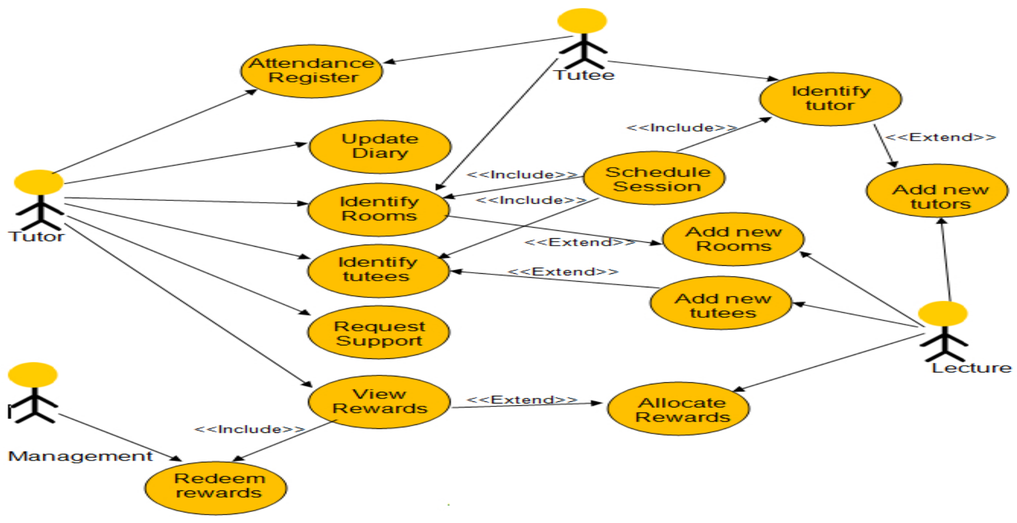


Figure 7: Peer-Tutoring System Use Case Diagram

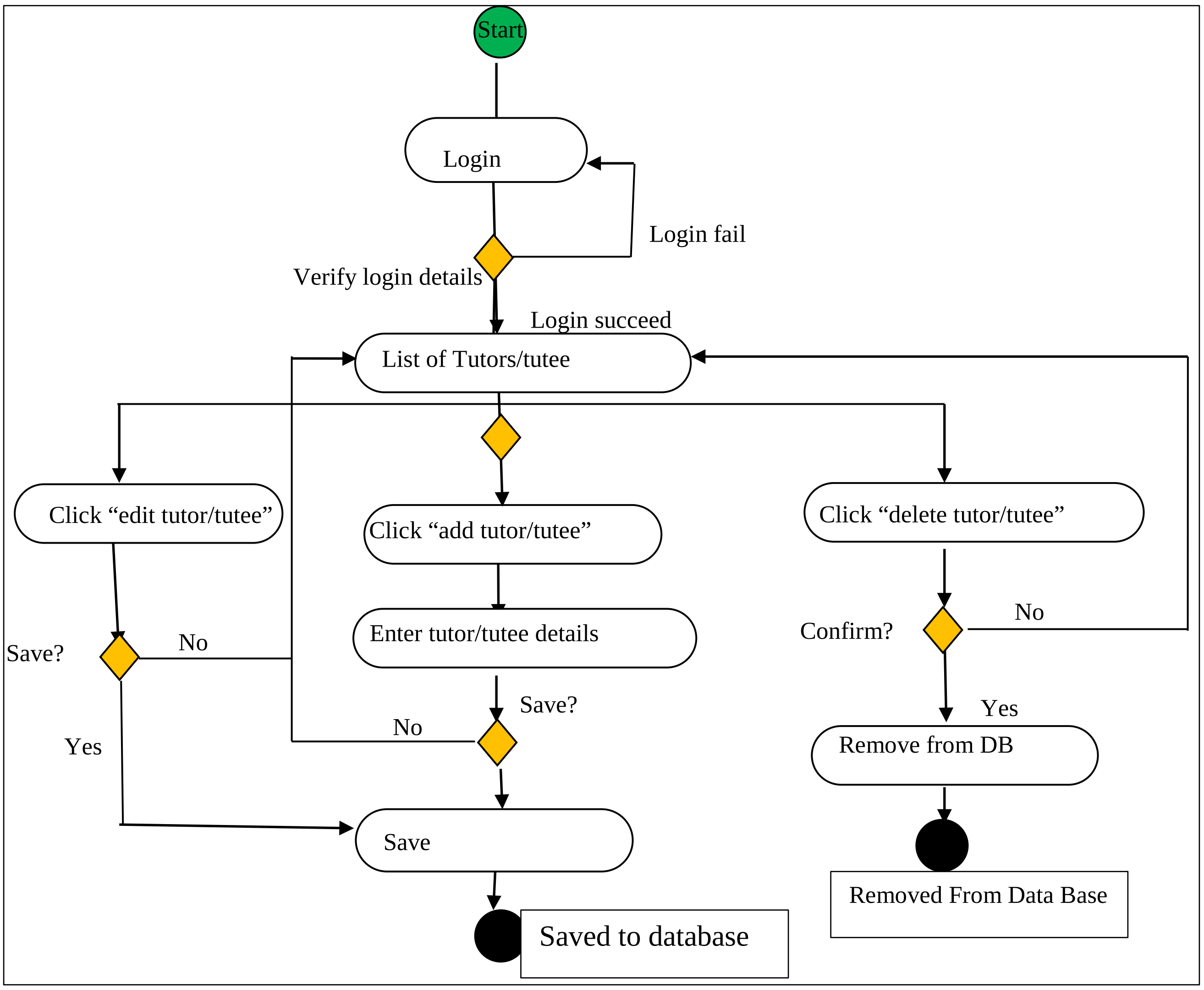


Figure 8: Activity Diagram to add, edit or remove a tutor or tutee

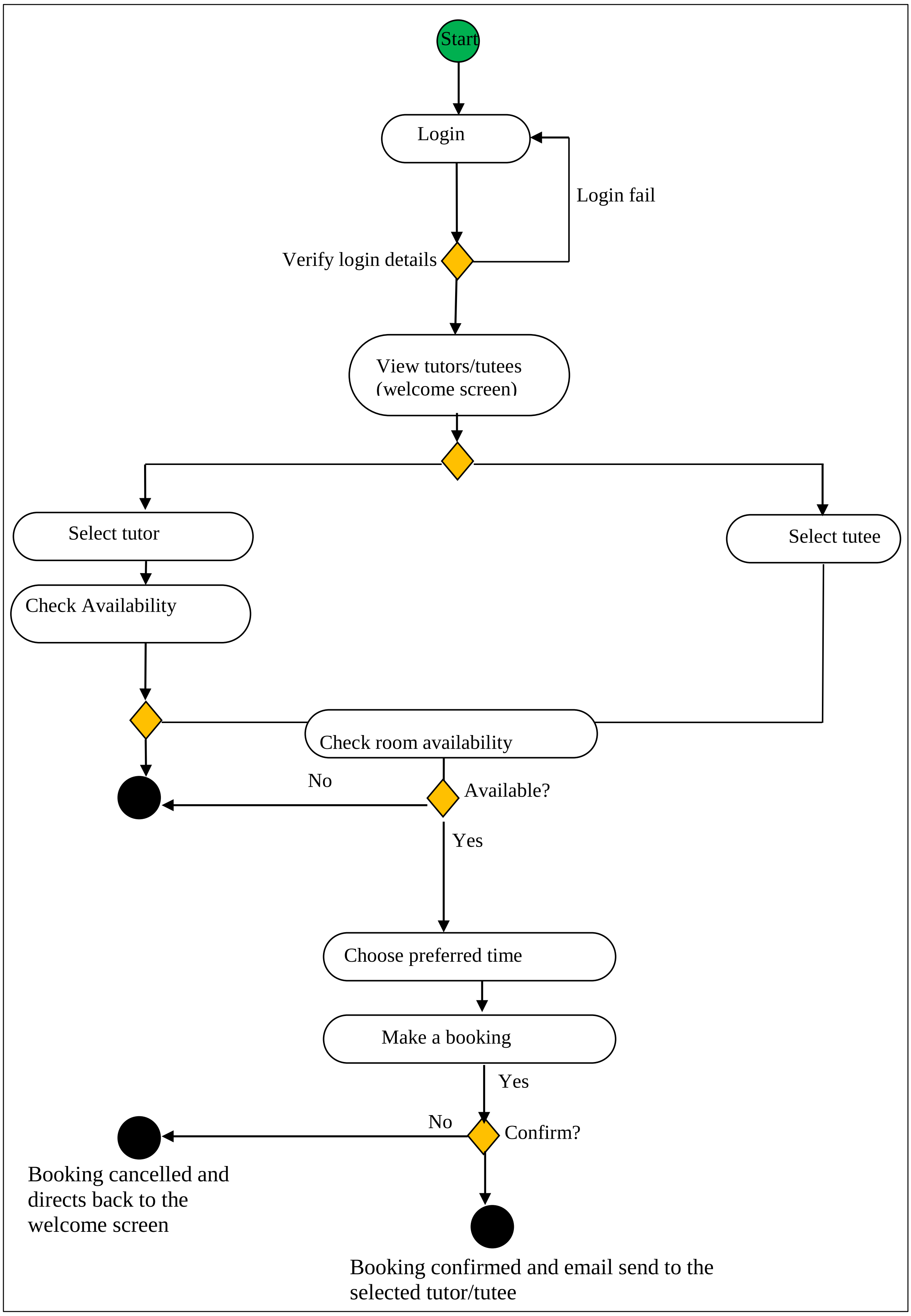


Figure 9: Activity diagram showing the scheduling of a session

6.2.3.2 Generating the Activity Diagram

The following diagrams presented part of the business domain processes as can be implemented later on or may some activities performed with a need of software system. Figure 8 represents the process to add, remove, or edit a tutor or tutee and Figure 9 represent session scheduling process.

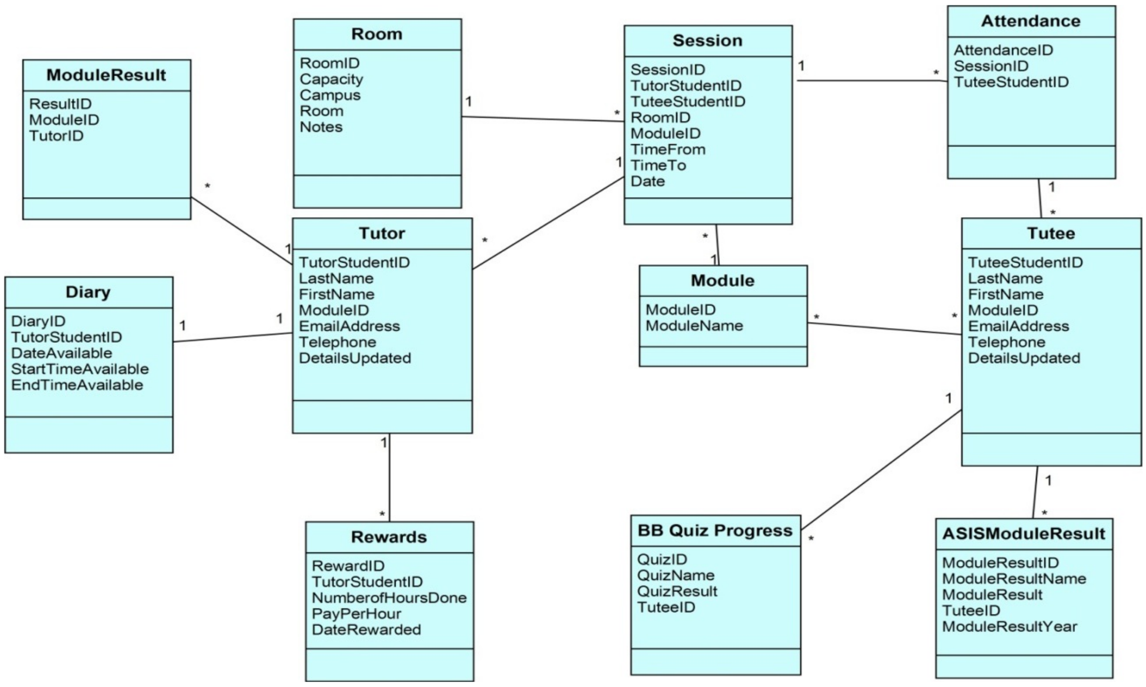


Figure 10: Class diagram

6.2.3.3 Generate the Class Diagram

A class diagram is a representation of a basic structure of a system, it shows what classes will be present in the system, how the classes are going to link between themselves and how many links to one class there would be from another. It's a presentation of the system to more detail (Oliver, I., Kent, S. 2009). Each use case presented using textual template, activity diagram, sequence diagram, and all use cases are combined in a use case diagram. The next step in the process is to take the business logic identified in the use cases and associate it with classes in a class diagram. We have followed the guideline that all important business logic must be implemented in classes in the domain model. Class diagram is the major part of the Domain Model that can be used to generate the programming code through the implementation pattern. Class diagram of PTS is presented in Figure 10.

6.2.3.4 Change report generation and refinement

As shown in the framework (SDDD), there is a draw back to the previous stages to refine what's done during Pre-SSM, SSM, and Post1-SSM. This refinement is essential to be sure that the exact changes required already modelled well as a domain model. As a guiding methodology, SSM focus on the generation of the required change report as a result to be recommended for the management actions ((Checkland, P., Poulter, J., 2006), (Checkland, P., 1999), and Checland, P., Howell, S.E,1998).

So, before leaving this stage, Domain model supposed to be refined and ready for implementation.

6.2.4 Post2-SSM Phase: Software Implementation

SDDD framework considered the domain model as the base to extract the programming code using the implementation pattern. Naked objects and Trueview are recommended as implementation patterns. Brief description and implementation of PTS using both patterns are presented in the following sections.

6.2.4.1 Naked Objects Pattern:

Naked objects was originally a framework written in Java only and uses Java reflection capabilities, it is supported by any development environment that supports Java but it is not a development environment itself, however though it was written in Java, it can still allow the business objects to be written in C# and VB.NET (Pawson, R, 2004). He defines Naked Objects framework as *"A set of Java classes that can be instantiated or sub-classed by an application"*. However, the java framework suffered criticism for its usability and interface presentation, another .NET version called Naked Objects Models, Views and Controllers (MVC) has been released since; this new version combines the original Naked Objects pattern thus having all business behaviour put in the domain objects and having the behaviours exposed to the user with Microsoft's ASP.NET MVC 2 framework (Naked Objects, 2010). The difference between these two Naked Objects versions is not anything to do with their philosophies but the language they are written in, the user Interfaces, authorisations just to mention a few. The reason MVC has been chosen for this projects instead of the Java framework is because the MVC application gives a user interface that is easier for the users to find their way around, more information on their usability. Most business systems today even in 'thin layer' architectures have adopted the architectural pattern of having four generic logical layers with every new business concepts having being implemented in all the four layers in different forms of course (Pawson, R., 2004). The layers are called presentation, controller, domain and Data management layers. Pawson cites (Brown, K., (995) who first recorded the four layer architectural pattern in 1995 but mentioned that the methods was practiced even before the recording (Pawson R, 2004). Pawson argues that “the relationships between the elements in those four layers often require a complex, many-to-many mapping. Although this generic architectural pattern has evolved over the years to meet certain needs, and although each of the layers may be object-oriented in some sense, this is a far cry from the original principle of behaviourally-complete objects”

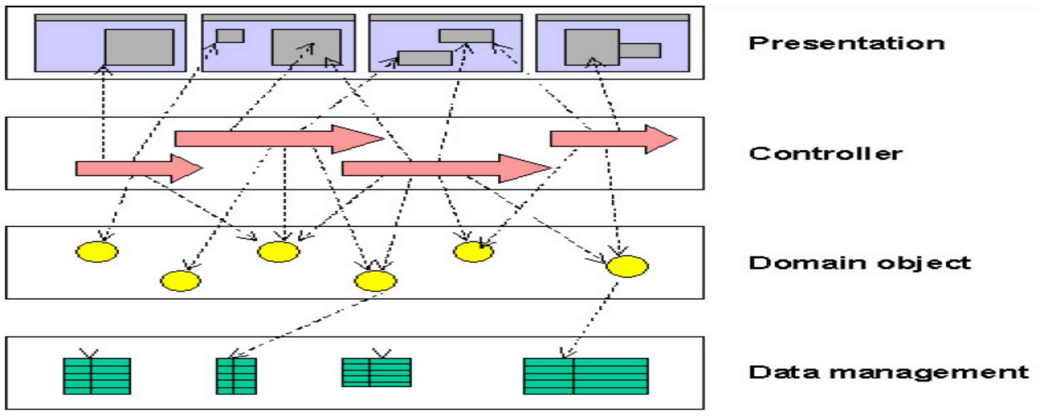


Figure 11: The four layer architectural pattern adopted by most business systems (Pawson R, 2004)

(Pawson R, 2004) suggested a solution to the problem above as requiring view and controller roles completely generic just as the original idea of MVC which writes the business application in terms of domain entities. Naked objects therefore gets rid of the effort to develop the controller and the presentation layers from the four layer architecture allowing users to interact directly with the domain objects since the controller layer is resided in the presentation layer and the presentation layer is provided automatically by the software thus reducing the work load on the developers.

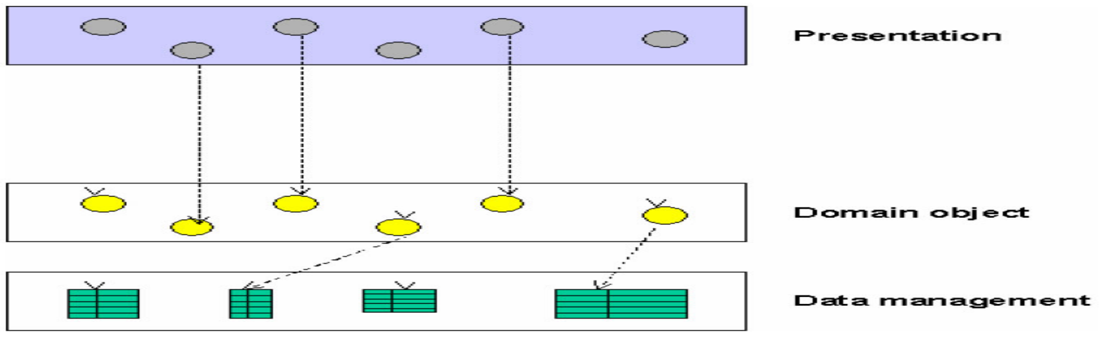


Figure12: Domain objects rendered visible to the user in a Naked Objects implementation with the required business functionalities encapsulated on the objects domain (Pawson, R. 2004)

Naked Objects operates an Object Oriented User Interface (OOUI) that allows the user to see and manipulate the domain objects' behaviours to do anything. Pawson mentions that the easier way to allow this is by presenting the domain objects as user icons and all behaviours required to action are presented as options for the icons.

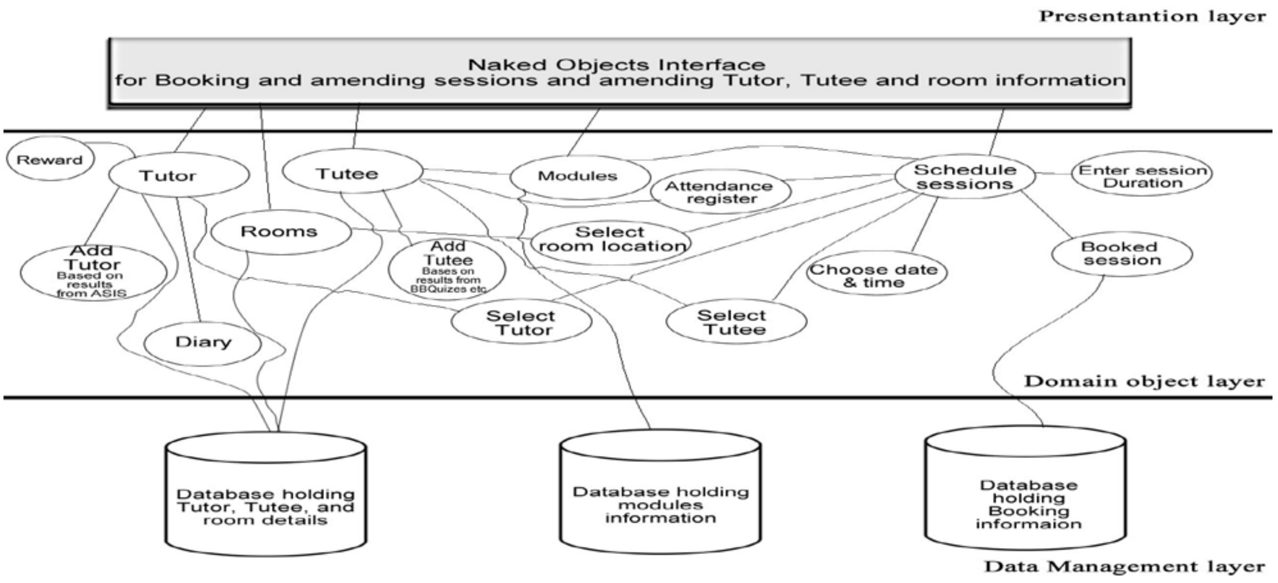

Presentantion layer
Naked Objects Interface
for Booking and amending sessions and amending Tutor, Tutee and room information
Reward
Tutor
Tutee
Modules
Attendance register
Schedule sessions
Enter session Duration
Rooms
Add Tutor Based on results from ASIS
Add Tutee Bases on results from BBQuizes etc
Select room location
Choose date & time
Booked session
Select Tutor
Select Tutee
Diary
Domain object layer
Database holding Tutor, Tutee, and room details
Database holding modules information
Database holding Booking informaion
Data Management layer

Figure 13: PTS architectural model implemented with naked objects.

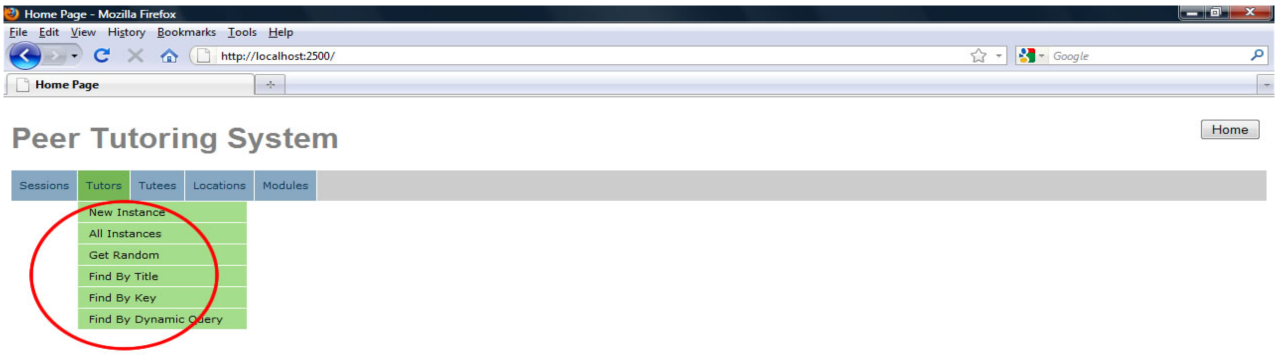


Figure 14: Naked Objects MVC application with a user's mouse hovering over an object making Object behaviours directly accessible to the user.

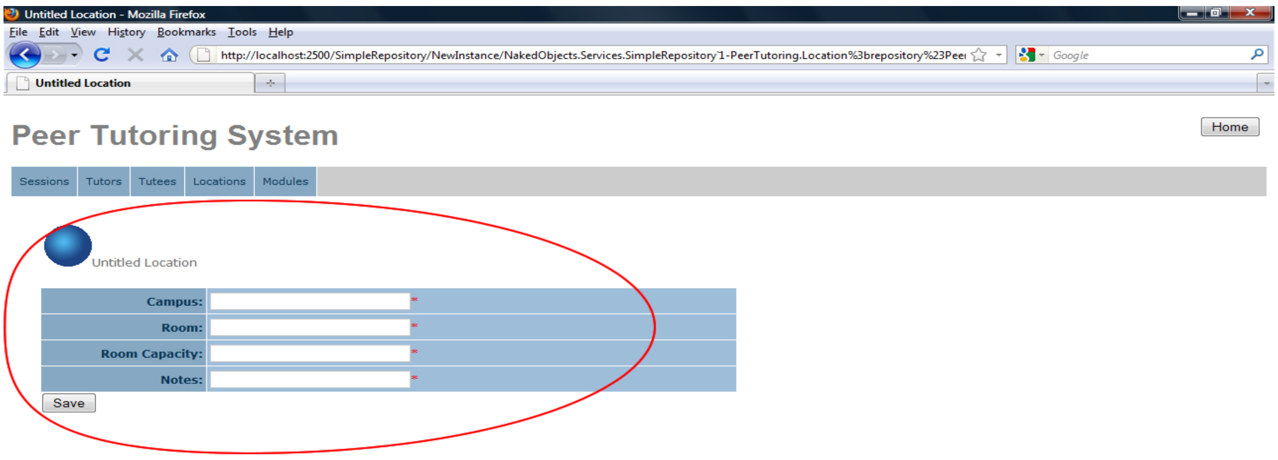
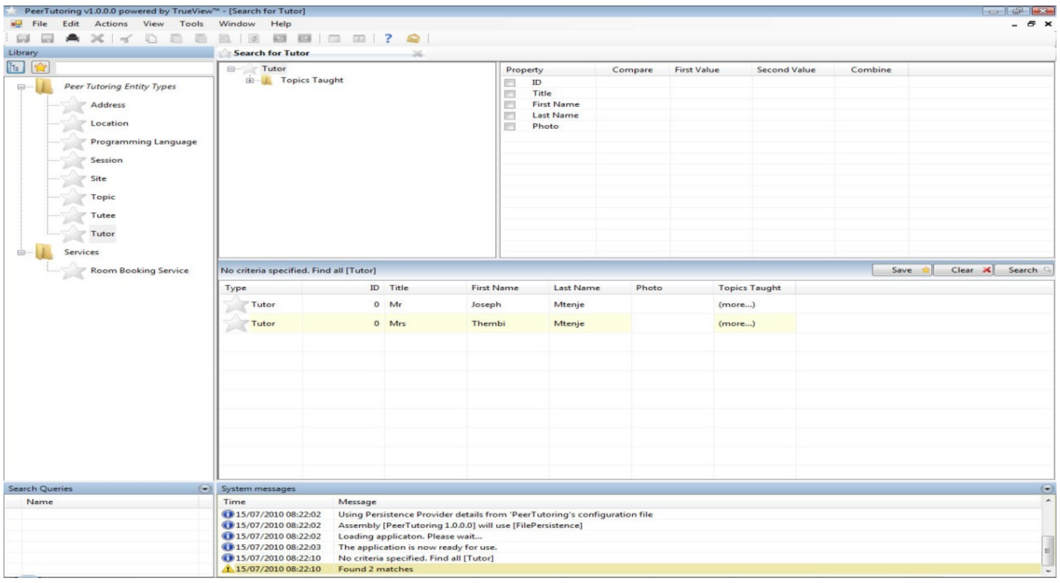

Untitled Location - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:2500/SimpleRepository/NewInstance/NakedObjects.Services.SimpleRepository`1-PeerTutoring.Location%3brepository%23Peer
Untitled Location
Peer Tutoring System
Home
Sessions Tutors Tutees Locations Modules
Untitled Location
Campus:
Room:
Room Capacity:
Notes:
Save

Figure 15: Naked Objects MVC application allowing a user to update the object's properties.

6.2.4.2 TrueView Implementation Pattern:

TrueView is software provided by Evolving software which is a company registered in England and Wales in 2006. TrueView creates applications based directly on .NET entities that are the classes developed in the UML stages as TrueView is used for "exploring business Domains and creating rapid prototypes using Domain Driven Design" and its applications focus on Domain models. Evolving software states that TrueView helps keeping Business logic clean, concise and focussed by having an Object Relational Mapping facility for data persistence. They also mention that the application was designed to suit problem solvers that is why it is being used in this project as it gives freedom and flexibility in DDD implementation as the interfaces can be customised, adds security capabilities and data persistence to an application. As TrueView's behaviours are controlled through attributes, it creates entity classes and relationships between them that help in having the whole system working to deliver efficiency in the booking system. We will also use the software to build an interface that users will use to access the system to do all activities and arrange for sessions. The following figures show the User Interfaces of TrueView as implemented in the PTS.



Property	Compare	First Value	Second Value	Combine
ID				
Title				
First Name				
Last Name				
Photo				

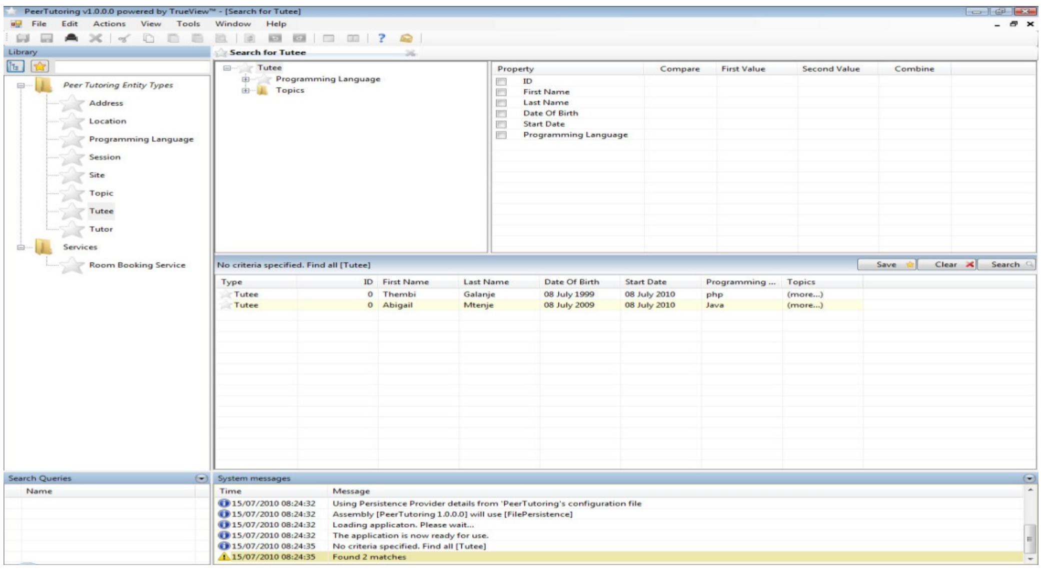
No criteria specified. Find all [Tutor]

Type	ID	Title	First Name	Last Name	Photo	Topics Taught
Tutor	0	Mr	Joseph	Mtenje		(more...)
Tutor	0	Mrs	Thembi	Mtenje		(more...)

System messages

```
Time                 Message
15/07/2010 08:22:02  Using Persistence Provider details from 'PeerTutoring's configuration file
15/07/2010 08:22:02  Assembly [PeerTutoring 1.0.0.0] will use [FilePersistence]
15/07/2010 08:22:02  Loading applicaton. Please wait...
15/07/2010 08:22:03  The application is now ready for use.
15/07/2010 08:22:10  No criteria specified. Find all [Tutor]
15/07/2010 08:22:10  Found 2 matches
```

Figure 16: Showing Tutors availability and also showing that a single tutor can teach more than one topics



Property	Compare	First Value	Second Value	Combine
ID				
First Name				
Last Name				
Date Of Birth				
Start Date				
Programming Language				

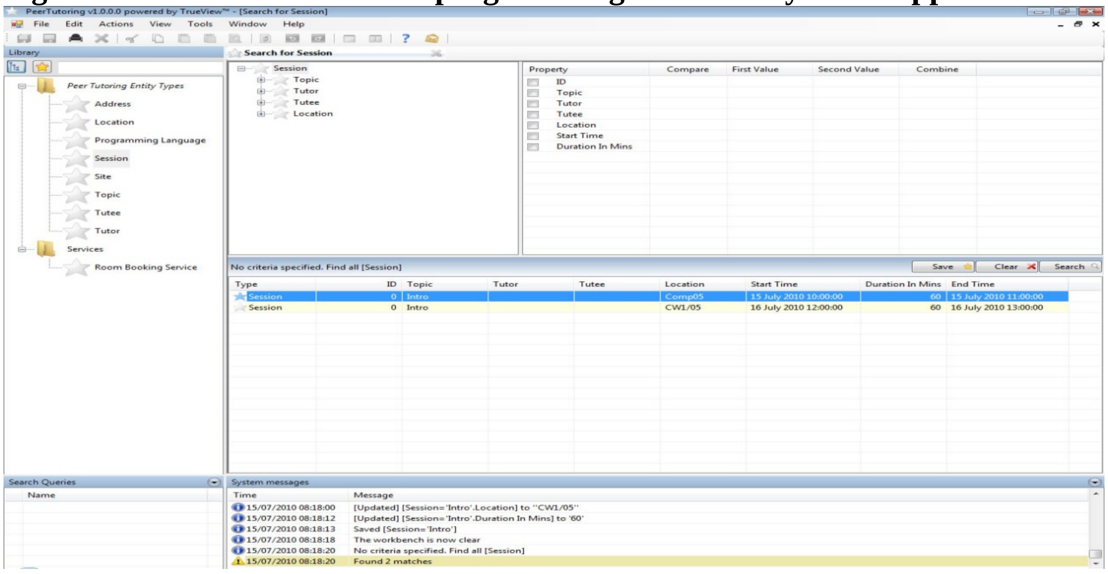
No criteria specified. Find all [Tutee]

Type	ID	First Name	Last Name	Date Of Birth	Start Date	Programming ...	Topics
Tutee	0	Thembi	Galanje	08 July 1999	08 July 2010	php	(more...)
Tutee	0	Abigail	Mtenje	08 July 2009	08 July 2010	Java	(more...)

System messages

```
Time                 Message
15/07/2010 08:24:32  Using Persistence Provider details from 'PeerTutoring's configuration file
15/07/2010 08:24:32  Assembly [PeerTutoring 1.0.0.0] will use [FilePersistence]
15/07/2010 08:24:32  Loading applicaton. Please wait...
15/07/2010 08:24:32  The application is now ready for use.
15/07/2010 08:24:35  No criteria specified. Find all [Tutee]
15/07/2010 08:24:35  Found 2 matches
```

Figure 17: list of tutees and the programming module they need support



Property	Compare	First Value	Second Value	Combine
ID				
Topic				
Tutor				
Tutee				
Location				
Start Time				
Duration In Mins				

No criteria specified. Find all [Session]

Type	ID	Topic	Tutor	Tutee	Location	Start Time	Duration In Mins	End Time
Session	0	Intro			Comp05	15 July 2010 10:00:00	60	15 July 2010 11:00:00
Session	0	Intro			CW1/05	16 July 2010 12:00:00	60	16 July 2010 13:00:00

System messages

```
Time                 Message
15/07/2010 08:18:00  [Updated] [Session='Intro'.Location] to "CW1/05"
15/07/2010 08:18:12  [Updated] [Session='Intro'.Duration In Mins] to '60'
15/07/2010 08:18:13  Saved [Session='Intro']
15/07/2010 08:18:18  The workbench is now clear
15/07/2010 08:18:20  No criteria specified. Find all [Session]
15/07/2010 08:18:20  Found 2 matches
```

Figure 18: sessions booked in TrueView

We tried the implementation with both patterns, and it up to the client to decide and evaluate which is better to his environment. Both patterns support the philosophy of DDD and then SDDD. But difference available in the user interface and usability. Some of the students asked to use both implementation and many of them preferred Naked Objects implementation. This paper will not deal with the comparative issue between Naked Objects and TrueView as an implementation patterns, and this will be for further work to investigate the usability of the implemented software using both patterns.

7.0 Feedback about using SDDD Framework in developing the PTS and teaching the module "Methods and Modelling".

7.1 Feedback from the Msc student who applied the framework to the PTS

In the evaluation part, the postgraduate student mentioned that he has not come across any combination as this, the closest one he has come across is the one used by (Lane and Galvin ,1999) where they combined and transited from SSM to Object Oriented Analysis. In this they moved from SSM use cases and developed use cases but did not proceed to build an application using DDD implementation software, while in SDDD framework, the application is built allowing users to access the business objects without using controllers which Lane and Galvin did not mention about.

Also, SDDDF has so many advantages but the main one is that is enables and equips the researcher to understand the problem situation better through SSM as it tends to get different views of the situation from different stakeholders at the root definition stage and as well as at DDD stage when you have to understand the business objectives and how activities are done. This enables one to build a better application that would suit the user requirements and also build a system that has better requirements as studies in the UML stage. The application is even easier to use as it gives a user direct access to business objects and can manipulate them easier than through controllers like in conventional MVC applications.

But he said that the point that he found difficult in the framework was the point of conversion from SSM to UML, as this is not a one to one conversion, but involves combination and decomposition of Conceptual models. He advised to do more research on this area to get a smoother and easier transition to ensure other researchers don't spend more time on I as he did.

About the implementation pattern he preferred Naked Objects on TrueView based on the students asked to use both implementations. The important issue he raised is the usability of the system developed using Naked Objects is better than the one with TrueView.

7.2 Feedback about using SDDD Framework in teaching Methods and Modelling

7.2.1 Students Background

This module is for Msc students in the Informatics Department in the University of Huddersfield. There are about 38 students joined the module in November 2011. A background questionnaire is distributed to them on the first class to find out information about their majors and experiences before doing this module. The purpose from doing that to see how we can deal with students if their backgrounds different. The analysis of the questionnaire show that two types of student taking the module:

- 18 students of MSc Advanced Computer Science. These students have a strong background in programming. Some experience of modelling but not with the UML. None of them were familiar with the idea of multimethodology. None of them had heard of SSM. Whilst studying methods and modelling these students are also studying advance software development modules in areas such as internet application development.
- 20 students of MSc Information Systems Management. These students do not have a strong back ground in programming. Most of them were unfamiliar with the principals of object oriented programming. Some experience of modelling but not with the UML. Most of them had heard of SSM but were not aware of the literature on multimethodology. Whilst studying methods and modelling these students are also studying information systems modules in areas such as competing in a digital economy.

The module is finished and the students initial feedback through discussion encouraged us to continue using this framework in the future and to do further refinement on it. We have received their practical work and marking is going on. Based on marking scheme and the module aim, we designed a questionnaire to be distributed to them this month to find out how much this framework as a teaching methodology contributes to the achievement of module aim. Certain hypotheses covering the entire framework are identified and will be the bases of the statistical analysis of the filled feedback questionnaire. This will be the target of next Journal publication.

7.2.2 Feedback from the lecturer of the module:

My modelling module is taught to students of MSc Information Systems Management and students of MSc Advanced Computer Science. The ISM students have difficulty relating the modeling to programming this is because they do very little programming on the modules they are studying. But the ISM students are comfortable with SSM techniques. The ACS students have a different problem. They do plenty of programming – lots of object oriented programming in Java but often can’t see the point of studying business analysis. I struggle to keep the two different audiences on board because of this difference in the contexts of their studies. All the students find sequence diagrams very difficult. We spend a lot of time on this topic but they still find it very confusing. In general the framework satisfied the objective of using it as a teaching approach for business domain modelling and implementation with little variety because of the major and background of the students.

8.0 Recommendations and conclusion

The work done in this paper focused more and highlighted the pedagogical evaluation of the multimethodology framework developed before that can handle both soft and hard issues of domain business process modeling and implementation as a software support system. The evaluated framework SDDD previously developed based on the combination of tools from UML as an approach for Domain-Driven Design modeling and Soft Systems Methodology for understanding the problematic situation of business domain. We have added a "soft" perspective on DDD to form "Soft Domain-Driven Design". The framework is being evaluated as an approach for business information systems development using Peer-Tutoring-System" (PTS) case study as an Msc project to show how the proposed framework can be applied to a real problem situation. The evaluation work is done in educational perspective to show how this framework can be used to teach Methods and Modelling module for Msc students in Informatics department. The feedback and from the lecturer and students supported our previous and current evaluation that this framework and can be used as an approach for business domain modelling and implementation as a software support systems. Further investigation is going on and a questionnaire is designed to investigate the students done this module to detailed reflection about the learning and practice for different tools used for teaching the module.

REFERENCES

Al Humaidan, F.(2006) *Evaluation and Development Models for Business Processes*, PhD thesis, University of Newcastle, UK.

Al-Humaidan, F.,Rossiter, N.(2004) *Business Process Modelling with OBPM combining soft and hard approaches*, in Proceeding of 1st Workshop on Computer Supported Activity Coordination (CSAC), 6th International Conference on Enterprise Information Systems, Porto, , pp 253-260.

Alter, S., (2007) *The work system method: Connecting people, processes and IT for business results*, Work System Press, Larkspur, CA.

Barjis Joseph (2008) *The importance of business process modelling in software* *systems design*, Science of Computer Programming Journal, vol 71 ,pp 73–87.

Bustard, D. W., Dobbin, T. J., Carey, B. N.(1996) *Integrating Soft Systems and Object-Oriented Analysis*, IEEE International Conference on Requirements Engineering, Colorado Springs, Colorado, pp. 52-59.

Checkland, P., Poulter J.(2006) *Learning for Action. A short Definitive Account of Soft Systems Methodology and its use for Practitioners, Teachers and Students*, John Wiley and Sons Ltd, West Sussex, England.

Checkland, P.(1999) *Systems Thinking, Systems Practice*", John Wiley and Sons Ltd, West Sussex, England.

Checkland, P., Holwell, S.E.(1998) *Information, Systems and Information Systems, Making sense of the field*, John Wiley and Sons Ltd, West Sussex, England.

Checkland, P., Jim, S. (1990) *Soft Systems Methodology in Action*, Toronto: John Wiley and Sons.

Daveport, T., (1993) *Process innovation: Reengineering work through information technology*, Harvard Business School Press, Boston, Mass.

D. Platt, (1994) *Process Modelling and Process Support Environment to Design Management*, Department of Civil Engineering, Faculty of Engineering, University of Bristol, UK.

Eric Evan , (2004) *Domain-Driven Design –Tackling Complexity in the Heart of Software*, Addisson Wesley.

Erikksonn, H. E., Penker, M.(2000) *UML business process modelling at work*, John Wiley and Sons, New York, 2000.

Gardner, H. (1993) *Multiple intelligences: the theory in practice*, New York, NY:Basic Books.

Goodlad, S., Hirst, B.(1989) *Peer Tutoring: A Guide to Learning by Teaching*, London: Kogan Page; New York: Nickols Publishing.

Hu Xiaohui.(2006) *Improving teaching in Computer Programming by adopting student-centred learning strategies*, China papers, issue 6. 46-51.

Lai, L.S. (2000) *An integration of systems science methods and object oriented analysis for determining organisational information requirements*, Systems Research and Behavioural Science 17, 205-228.

Lane, C., Galvin, K. (1999) *Methods for Transitioning from Soft Systems Methodology (SSM) Models to Object Oriented Analysis (OOA*), developed to support the Army Operational Architecture (AOA) and an Example of its Application pp12-13.

Mingers J.(2001) *Combining IS Research Methods: Towards a Pluralist Methodology*, Information Systems Research, 12, 3, Institute for Operations Research and the Management Sciences (INFORMS), pp. 240-259.

Miliszewska Iwona , Tan Grace.(2007) Befriending Computer Programming: A Proposed Approach to Teaching Introductory Programming. Issues in Information Science and Information Technology, volume 4, 277-289.

Naked Objects (2010) *Naked Objects MVC - Product description* [online] available at: <http://nakedobjects.net/product/product_intro.shtml> [accessed on 15th August 2010].

Oliver I., Kent, S. (2009) *Validation of Object Oriented Models using Animation*, [online] available at: <http://kar.kent.ac.uk/21768/1/validation_of_object-oriented_oliver.pdf> accessed on 24th June 2010.

Pawson R. Mathews R.(2002) Naked Objects, John Wiley and Sons Ltd, West Sussex, England.

Pawson R. Mathews R.(2002) Naked Objects, John Wiley and Sons Ltd, West Sussex, England.

Salahat , M., Wade, S., Lu, J.(2008) *A systemic Framework for Business Process Modelling and Implementation*, In the proceeding of 5th International Conference on Innovations of Information Technology (Innovations'08), UAE University, Al Ain, UAE, in IEEE xplore 978-1-4244-3397-1/08..

Salahat, M., Wade S.(2009) *A Systems Thinking Approach to Domain-Driven Design*, In the proceeding of UKAIS2009 conference, Oxford University, Oxford, UK.

Salahat , M., Wade, S., Ul-Haq, I.(2009) *The Application of A systemic Soft Domain Driven Design Framework*, WASET online Journal, Issue57,pp476-486.

Sewchurran, K. & Petkov D.(2007) *A systemic Framework for Business Process Modelling Combining Soft Systems Methodology and UML*, Information Resources Mnagement Journal, 20, 3, IGI Publishing, PA,USA, P. 46-62.

Svatopluk Štolfa, Ivo Vondrák,(2006) *Mapping from Business Processes to Requirements Specification*, Retrieved on 7th Aug, 2008 from 85.255.195.219/conf/esm/esm2006/abstract.pdf.

Wade, S., Hopkins, J.(2002) *A Framework for Incorporating Systems Thinking into Object Oriented Design*, Seventh CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'02), Toronto, Canada, May ,27-28.

Warboys, Brian, Kawalek, Peter, Robertson, Ian, and Greenwood, Mark, (1999) *Business Information Systems-A process approach*, McGraw-Hill, UK.

Williams B. (2005) *Soft Systems Methodology* [online] available at: <http://users.actrix.co.nz/bobwill/ssm.pdf> [accessed on 18th June 2010].