Contents

[Systems Development Methodologies 3](#_Toc386749731)

[1. 1 Introduction 3](#_Toc386749732)

[1.2 Structured Design 3](#_Toc386749733)

[1.2.1 Waterfall Development 3](#_Toc386749734)

[1.2.2 Parallel Development 3](#_Toc386749735)

[1.3 Structured Design Conclusion 4](#_Toc386749736)

[2.1 Rapid Application Development 4](#_Toc386749737)

[2.2 Approaches to RAD 5](#_Toc386749738)

[2.2.1 Phased Development 5](#_Toc386749739)

[2.2.2 Prototyping 5](#_Toc386749740)

[2.2.3 Throwaway Prototyping 6](#_Toc386749741)

[2.3 RAD Conclusion 6](#_Toc386749742)

[3.1 Agile Development 6](#_Toc386749743)

[3.2 Extreme Programming 7](#_Toc386749744)

[3.2.1 Continuous Testing 7](#_Toc386749745)

[3.2.2 Simple Coding 7](#_Toc386749746)

[3.2.3 Close Interaction 7](#_Toc386749747)

[3.3 Agile Development Conclusion 8](#_Toc386749748)

[System Analysis and Design Projects 9](#_Toc386749749)

[1.1 Project One Background 9](#_Toc386749750)

[1.2 Use Case 9](#_Toc386749751)

[1.3 Entity Relationship Diagram 9](#_Toc386749752)

[1.4 Evaluation 10](#_Toc386749753)

[2.1 Project Two Background 10](#_Toc386749754)

[2.2 Sequence Diagrams 10](#_Toc386749755)

[2.3 Evaluation 11](#_Toc386749756)

[3.1 Conclusion 11](#_Toc386749757)

[References 13](#_Toc386749758)

[Appendix 14](#_Toc386749759)

[Figure 1 14](#_Toc386749760)

[Figure 2 15](#_Toc386749761)

[Figure 3 15](#_Toc386749762)

[Figure 4 16](#_Toc386749763)

[Figure 5 16](#_Toc386749764)

[Figure 6 17](#_Toc386749765)

# Systems Development Methodologies

## 1. 1 Introduction

There are three main development methodologies; Structured Design, Rapid Application Development (RAD) and Agile Development. Each has different focus within the System Development Life Cycle (SDLC) and will progress through it at different speeds.

## 1.2 Structured Design

Structured Design is a step-by-step approach to the SDLC. There are two approaches that are used; Waterfall Development and Parallel Development which are used for process-centred and data-centred methodologies.

### 1.2.1 Waterfall Development

Waterfall Development is a way of progressing from one phase to the next similar to how a waterfall flows from top to bottom. There are usually five stages to a waterfall model; Requirements, Design, Implementation, Verification and Maintenance. This approach requires the full design to be completely specified before the implementation begins which can lead to long-time delays between the systems proposal and the system delivery. An area where this approach can be used in in industries such as construction or manufacturing where changes in the design can be costly.

### 1.2.2 Parallel Development

Parallel Development, unlike Waterfall Development which develops sequentially, seeks to reduce time by developing a series of projects in parallel producing possibly different versions of the systems rather than one. These will all be integrated back into one system before delivery to the client. An area that would use this development is software development. For example the company would release version 1, then whilst version 2 was being developed version 1 would branch off and work on critical fixes in parallel with the development of version 2. Once the fixes are done they are released as version 1.1 and the important fixes are merged from version 1.1 to version 2.

## 1.3 Structured Design Conclusion

Structured Design is a good methodology when the project is for a complex system. It is able to cover most types of time schedules with Waterfall Development for longer schedules and Parallel Development for short time schedules.

## 2.1 Rapid Application Development

Another approach is Rapid Application Development (RAD). This approach aims for minimal planning to develop a section of the system as fast as possible which is then delivered to the client so that they can offer amendments. This however can cause the clients to have increase expectation of the final system. There are four phases of RAD; Requirements Planning Phase, User Design Phase, Construction Phase and Cutover Phase.

Requirements Planning Phase combines the system planning and analysis phases from the SDLC. It is the phase that allows all those involved to discuss the requirements, constraints and that system aim. It ends when those involved have agreed on the key issues.

The User Design Phase allows the users work together with the systems analysts to develop models that represent all the system’s inputs, processes and outputs. The User Design phase is a continuous process of users modifying the system to one that meets their needs.

The Construction Phase is similar to the application development section of the SDLC however in RAD users will continue to have an input on the system development suggestion changes or improvements.

The Cutover Phase is similar to the final task of the SDLC implementation phase such as testing and user training. Compared with other approaches this phase is shorter allowing the system to be built, delivered and placed in operation sooner.

## 2.2 Approaches to RAD

### 2.2.1 Phased Development

Phased Development splits the systems into version that are developed sequentially similar to the Waterfall Development from Structured Design. The analysis phase identifies the overall system concept and the development team then collaborates with the users to develop the series of versions. They work together to identify the key requirements which version one will be developed based on those requirements. Once version one has been created it is evaluated and requirements are created for version two. This development cycle will continue until the system is complete or obsolete. This approach is beneficial to the speed of the project and allows for the user input on its creation however the users are evaluating an incomplete version of the final system.

### 2.2.2 Prototyping

*“System prototyping performs the analysis, design and implementation phases concurrently in order to quickly develop a simplified version of the system”* (Dennis et al. 2012). This means that user input about the system can be given and used to aid further development which continues to repeat until the client agrees the system is complete and it is installed into the organisation. This approach is allows for user involvement and as an information-gathering technique allowing developer to develop based on feedback. This gives the potential for changing the system early in its developments or sopping development on an unworkable system. There is however the difficulty in managing the process because of its rapid and iterative nature. It also requires feedback otherwise further development may not be possible

### 2.2.3 Throwaway Prototyping

Throwaway Prototyping has a focus on the analysis phase of SDLC and sections of the system are created as demonstration tools to help the user or client to understand how the system will work. This allows the user or client to have their input on changes or improvements to the system and will lead to a more refined system that meet the client’s needs. Because of this process the prototypes do not continue their development unlike in Prototyping as Throwaway Prototyping *“uses the prototypes primarily to explore design alternatives rather than as the actual new system”* (Dennis et al. 2012). This approach is similar to Spiral Development Methodology which is a *“risk-driven approach to the software process rather than a primarily document-driven or code-driven process”* (Boehm 1988). Each cycle of the spiral identifies the objectives of the section being produced, the alternative mean of implementation of that section and the constraints of the alternatives.

## 2.3 RAD Conclusion

The benefits from RAD are that it promotes collaboration and gathering of requirements from the client. It allows for the client to actively work with the developing team in prototyping and testing. There are however disadvantages from RAD as it requires collaboration between teams and the clients in order to produce a system for their needs. A client could also change the desired system after haven seen the prototype.

This style of development is most common in Games development which allows developers to produce an alpha or beta of their game to gather any changes or improvements from their audience.

## 3.1 Agile Development

The third approach is Agile Development. This approach is based on iterative and incremental development where requirements and solutions evolve through the collaboration between teams. It is an approach that promotes adaptive planning with a rapid response to change. As Agile Development can be used for programming-focused methodologies an example could be Extreme Programming.

## 3.2 Extreme Programming

Extreme Programming is a software-focused development methodology which aims to improve the quality of software and the responsiveness to any changes in requirements. The analysis, design and implementation phases are all performed interactively with the planning and feedback looping.  
There are three key Extreme Programming principles; Continuous Testing, Simple Coding and Close Interaction.

### 3.2.1 Continuous Testing

Extreme Programming’s approach believes that if testing can improve on the systems and remove some of its imperfection therefor a lot of testing can remove lots of imperfections. There are two types of test that are used; Unit Tests which tests if a feature work an intended and Acceptance Tests which verify that the requirements gathered meet the clients actual requirements.

### 3.2.2 Simple Coding

This technique is one done in pairs, one programmer writes the code whilst the other programmer reviews each line of code written. The programmer that is reviewing will think about the strategic aspect of the work, how the current code can be improved, predicting future problems and how to address them. This allows the programmer writing the code to focus on the tactical aspect of completing the tasks at hand and using the reviewer as a guide. To allow for different ideas and improvements the pair will switch roles frequently.

### 3.2.3 Close Interaction

In order to produce a system that meets the client’s requirements the development team will work closely with them for communication and to gather feedback allowing the tram to build the system quickly and as close to the actual requirements as possible.

## 3.3 Agile Development Conclusion

A benefit of Agile Development is that it develops in short intervals which produces miniature software projects and allows the system to be released in mini-increments however short iteration could result in too little functionality leading to delays. Another disadvantage of using this approach is that it emphasizes real-time communication which can become problematic in larger team. Agile methods also produce very little written documentation throughout the development of the system and instead require a large amount of post-project documentation.

# System Analysis and Design Projects

## 1.1 Project One Background

For the first team project a dance studio owner required a new website designing for their company. The team was to come up with a solution for the requirements given.

## 1.2 Use Case

One of the contributions made for the project was Use Case designs. These documents would show how a process was carried out. An example of a Use Case is figure 1. This Use Case shows the customer registration form the customers point of view. The document helps to show how information in gathered from the user and where the information will then be stored internally. Another example was shown in figure 2 which shows how the payments from booking a class will be handled through the internet using PayPal. All Use Cases that were created in the project were done using the table creation in Microsoft Word which satisfied the need for creation a Use Case form. The purpose of using Use Cases in the project was to show and help the client understand how the system would process the features that were created based on the functional requirements which they do well.

## 1.3 Entity Relationship Diagram

Another contribution to the project was an Entity Relationship Diagram (ERD). ERD’s are used to show a database in an abstract way. ERD’s are a good way to represent what information stored and how one class can be used by another class. Figure 3 is an ERD to show the different database classes that would be used and how they interacted with each other. This diagram was created in Microsoft Visio which was a good tool for creating diagrams. As the project was to design a website system that would be used to book classes, handle payments and hold information creating an ERD was a good contribution to the give understanding of how the system would work and how the information was being used across the system.

## 1.4 Evaluation

Overall the project was as successful project as the team was able to create a solution for the requirements. Project one followed the Waterfall Development from the Structured Design approach. The team after receiving the requirements created the diagrams and documents sequentially each showing how part of the system worked which was put into a single Microsoft Word document. This approach meant that work could be produce well but the pace of it was slow with some pieces not being collaborated until the last few class sessions.

## 2.1 Project Two Background

For the second team project a community centre required a new system design to automate rota creation and distribution, event planning and to create and to store the members and volunteers data. The team was to come up with a solution for the requirements given.

## 2.2 Sequence Diagrams

The main contribution to this project was Sequence Diagrams. These diagrams are used to show the processes operate with one another and in what order. One example is figure 4 shows a user registering their details and verifying their email to create an account. The diagram shows the flow of processes in order to complete the registration. A more detailed process is shown in figure 5 which a user trying to log in to their account. A final example is figure 6 showing how the clerical assistant would create an event and distributed it to the volunteers and members. These diagrams were created in Microsoft Visio which way a great tool to use for their creation. The Sequence Diagrams that were created for the project are a good representation of the systems processes that are carried out when a user interacts with the website.

## 2.3 Evaluation

Overall the project was as successful project as the team was able to create a solution for the requirements. Project two followed the Parallel Development from the Structured Design approach. The team after receiving the requirements chose which diagrams and documents they wanted to create and put into a shared Google Document. The approach allowed the work to be produced at a much faster pace than in project one.

## 3.1 Conclusion

The first improvement from project one to project two was how the work was stored and collaborated. Project one was a word document that one person kept and everyone stored their work in it whereas in project two the work was kept in a Google document that allowed anyone to alter and allowed people to work simultaneously on it.

The second improvement was the team’s communication. In the first project the team had poor communication which was done through the class sessions depending on whether the team members attended, in project two the team still had poor communication as it continued to be done through class session however towards the end of the project the some team members began to communicate through the Google document stating which topics or diagrams they would be doing.

The third improvement was in work produced and the diagrams that were created. At the start of project one the Waterfall Development approach was taken as the team would produce the work in class together however towards the end and due to lack of communication work was not being produce which meant that as the project’s deadline approached and the work that had not been done needed to be produced fast. For this a RAD approach, Phased Development, was taken in order to create the diagrams and pieces of work to meet the project requirements. This approach allowed for work to be produced fast. In the second project, having learnt from the previous projects mistakes over time constraint, a Parallel Development approach was used so that team members select an area they wanted to develop and produce work which was then stored in the Google Document. A benefit of this approach meant that team members were not waiting to see what others were doing allowing for more quantity and quality of work produced.

Overall both projects were a success as they were able to create a solution for the clients. In project one there were many faults such as poor communication that were addressed in project two allowing for a higher quality project two overall. Project two allowed for the development of skills and the creation of diagrams that were not used in the first project for example sequence diagrams (Figure 4). It also allowed for an improved approach to how the work was produce and an increase in communication between team members.

## References

Boehm, B. (1988) A Spiral Model of Software Development and Enhancement, IEEE Computer

Dennis, A., Wixom, B., H., & Roth, R., M. (2012) Systems Analysis & Design, Wiley, 5th Edition

## Appendix

### Figure 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Use Case Name: Customer Registration | | | ID: 1 | Importance Level: 1 | |
| Primary Actor: Customer | | | | | |
| Short Description: Customer registers to a website | | | | | |
| Trigger:  Type: External Temporal | | | | | |
| Major Inputs | | Major Outputs | | | |
| Description  First Name Surname Address 1 Address 2 City Postcode Email Phone Number Mobile Number Username Password | Source  Registration Form Registration Form Registration Form Registration Form Registration Form Registration Form Registration Form Registration Form Registration Form Registration Form | Description  Customer Account | | | Destination  Account Database |
| Major Steps Performed:   1. New user register detail into a registration form 2. New user submit form 3. Data entered is validated 4. Email is checked if unique 5. Email confirmation sent 6. Customer account created | | | | Information for Steps:  First Name Surname Address 1 Address 2 City Postcode Email Phone Number Mobile Number Username Password | |

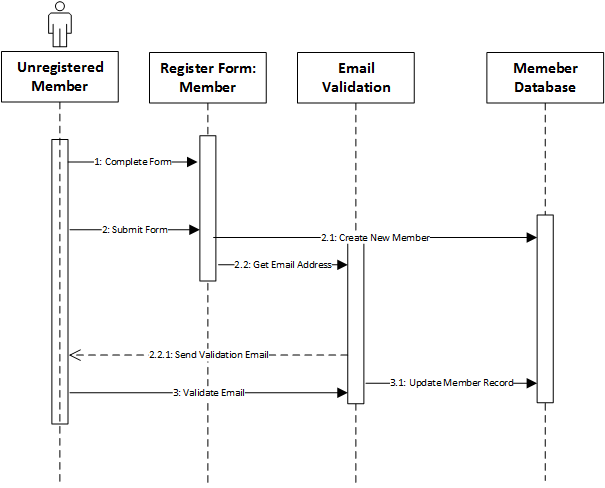
### Figure 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Use Case Name: Payment for Booking | | | ID: 4 | Importance Level: 2 | |
| Primary Actor: Customer | | | | | |
| Short Description: Customer pays for a booking | | | | | |
| Trigger:  Type: External Temporal | | | | | |
| Major Inputs | | Major Outputs | | | |
| Description  PayPal Account email PayPal password | Source  PayPal PayPal | Description  Customer Account | | | Destination  Account Database |
| Major Steps Performed:   1. Customer input PayPal email 2. Customer input PayPal password 3. Customer confirms payment for booking | | | | Information for Steps:  PayPal Account email  PayPal password | |

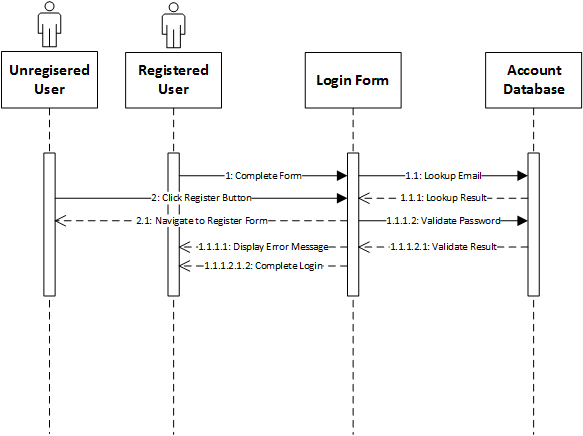
### Figure 3



### Figure 4



### Figure 5



### Figure 6

