



Systems Development Methods:
Element 3 Individual Element

Graduate



Directory

Harry Schilbach
Word Count: 2380



Contents

The Brief:.....	2
3. Discuss to what extent and in what ways you feel your chosen method above would be or would not be useful to develop a real system for The Graduate Directory. 10 marks.....	2
Explain the Roles of Life Cycles, Methods, Tools & Techniques in System Development.....	2
Crystal Clear Methodology	4
The ‘Three Pillars’	5
Frequent Delivery	5
Reflective Improvement	5
Osmotic Communication	6
Getting into the Safety Zone	7
Would Crystal Clear be Useful In the Development of The Graduate Directory?.....	8
Works Cited.....	10
System Request:.....	11
The Graduate Directory	11
Introduction	11
Project Sponsors.....	11
Business Practice	11
Inventory	11
Project Scope and Goals	11
Data Model	12



Tagging The World: An Investigation into Radio Frequency Identification (RFID) Technology by [Harry Schilbach](#) is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#).

Permissions beyond the scope of this license may be available at www.schilbach.co.uk/contact.

The Brief:

For element 3 you are to work individually to demonstrate an understanding of the role of life cycles, methods, tools and techniques in systems development and to research a method, its uses, limitations, tools and techniques. This element covers learning outcomes 1 and 4, comprises a 2,000 word report and counts for 50% of the overall mark for the module.

The first stage is to undertake research on systems development methods and to put forward a proposal as to which method you will use for your report. **This proposal must be agreed by the module leader before you start the ICA.**

According to Britton & Doake in their book „Software System Development: A Gentle Introduction“ : “A software development **method** is usually based on a life cycle model of system development and has a number of development phases with a set of steps and rules for each phase. Whereas a life cycle coarsely partitions the development of a system into stages, a development method takes a life cycle and further divides each of the stages into a number of steps. A development method will prescribe in great detail what tasks are involved in each step, the nature of each task, the order in which the tasks need to be done, what documents are produced at each stage and what documents are required as input to each stage. In fact, it provides a detailed plan for producing a system.”

Bearing this definition in mind and using the research you have carried out, write a 2,000 (+/-10%) word report that covers the following areas. State the total number of words at the end of the report. **Use examples from The Graduate Directory to illustrate your answer where appropriate.**

1. Explain the role of life cycles, methods, tools and techniques in systems development. **10 marks**
2. Explain the stages of your chosen method, the steps within those stages, the nature of the tasks, the order in which the tasks need to be done, what documents are produced at each stage and what documents are required as input to each stage. **20 marks**
3. Discuss to what extent and in what ways you feel your chosen method above would be or would not be useful to develop a real system for The Graduate Directory. **10 marks**

Explain the Roles of Life Cycles, Methods, Tools & Techniques in System Development

Life cycles, within software development, also known as software development life cycles (SDLC) were established out of necessity. From the beginnings of applied computing cracks began to appear when software solutions were scaled up for use on big projects. Even as early as 1972 Edsger Dijkstra (Dijkstr, 1987) talked of a 'software crisis' that was leading to the late delivery of over-budget and unreliable software solutions that did not solve the problems that were asked of them. As hardware became more powerful and allowed systems designers' greater scope and ambition the code underlying the software increased exponentially in both complexity and quantity. One of the causes of the software crisis was the inability in managing the development of this software; methodologies were needed.

SDLCs were considered part of the solution. In the past systems had been developed by computer engineers and scientists as a whole project from beginning to end, usually by a small team of dedicated programmers. This

worked well for small projects but struggled to deal with large scope projects; a more formal approach was needed.

The central tenant of SDLCs is to break down a whole project into defined stages. These should be followed by everyone within the team and "...guide your work and influence the quality of your final product..." (Hoffer, George, & Valacich, 2008). Development life-cycles become a template for the project to follow and allow a large project to become manageable, quantifiable and more successful.

They are a large variety of SDLCs to follow but all contain common elements and stages. Almost all contain the following:



Figure i - Common Elements of SDLCs.

Planning is the primary stage and involves identification of a problem that could be solved by new or updated software or information systems, in the case of the Graduate Directory this could be the recognition that the current system is not fit for purpose or that the Graduate Directory is expanding its scope. This could also involve prioritising of those problems or needs

Once planning has been undertaken the SDLC moves onto Analysis. The goal of this stage is to gain deep understanding of the current state of information systems within an organisation, be they computerised or not. This allows for analysis of the systems required, their scope and perhaps their structure. With our Graduate Directory this could involve talking to current end users like the

students, or guests, and also talking to administrators to discuss what functionality they would like in the system. It is vital to engage the people that are going to be using the system at this stage and indeed throughout the process.

Once thorough analysis has been done it's time to turn to design of the system. Several solutions maybe outlined at this stage, but not in great detail. Solutions should contain realistic estimates of cost, time to complete and deliverables of the system. They could also contain different hardware and software requirements. Within the Graduate Directory this could include solutions that require a dedicated website or to run only within an intranet, a version accessible on mobile devices may be considered

Once the design has been considered and perhaps revised and the client or users' are comfortable with the solution, implementation can begin. Here we build the system using the design and all the knowledge we've gained. Implementation spans the build of the system, the installation of the software and hardware, the supporting system, the transfer of any data from legacy systems, the training and the handover of documentation.

Job done? Not quite. One of the most crucial stages is the Maintenance of the system that has been implemented. This can often be neglected once the excitement of a successful implementation has passed. Maintenance can include bug fixes, compliance with changing legislation, version succession and compiling feedback from users' or the client. Indeed this stage could lead back to the design stage if major revisions are required. Maintenance should be seen as a business opportunity as it allows software developers to consolidate a relationship with a customer which could lead to further work in the future.

Let's take a more in-depth look at one of these methodologies; the Crystal Clear Methodology

Crystal Clear Methodology

Crystal Clear methodology comes from Alistair Cockburn summation of ten years research into what makes software development teams successful (Rusk, 2006). Cockburn studied software developments teams and how they worked, or indeed failed, and found through first-hand interviews that a lot of the aspects that the teams considered to be vital for success were overlooked by other methodologies.

Cockburn looked to exploit these aspects to create a more flexible methodology, he places a lot of importance on three central tenets, with a further four to really '...get into the safety zone...' (Cockburn, 2005) of his methodology. Crystal Clear gives the software development team the freedom to choose how they work within those tenets; after all these the software team is constructed out of software professionals, and it is the team that are best

suited to the project, which is why Crystal Clear leaves so many details undefined.

The 'Three Pillars'

Cockburn almost reverse engineered his methodology using interview techniques to establish what various successful software development teams themselves considered to be the vital elements to the projects' success. Cockburn was surprised by the simplicity of some of these and went on to use them as the main components of his methodology:

Frequent Delivery

Crystal Clear methodology prescribes that frequent delivery is key to project success, Cockburn believes this to be the 'single most important property of any project' (Cockburn, 2005). Crystal Clear talks about delivering iterations or segments of software at the very least every 3 months, but ideally between every 2 weeks to 2 months. Cockburn goes as far to say that if you are delivering code outside this period, you cannot be considered to be following this methodology.

The benefits of frequent delivery are manifold. They create a close relationship with the end user, allowing them 'hands-on' experience of the software-in-progress, this can ensure that the development team is fulfilling the brief, that any 'real-world scenario's that have been left out of the requirements documents are discovered - user feedback is invaluable. Frequent delivery also helps motivate the development team giving relatively short achievable goals that they can get quality feedback on; no developer wants to rewrite a years' worth of code because a vital step has been missed out. There are several ways of 'delivering frequently' it could be via software updates to the entire user base, it could be to the project sponsor or client; or even to a test environment with a single machine - just deliver!

The vital thing is that the relationship with the user is encouraged and developed through regular exposure to the software and the development team. Cockburn believes that if at all possible a user should be seconded to the team allowing instant feedback and a 'sounding board' for the developers'.

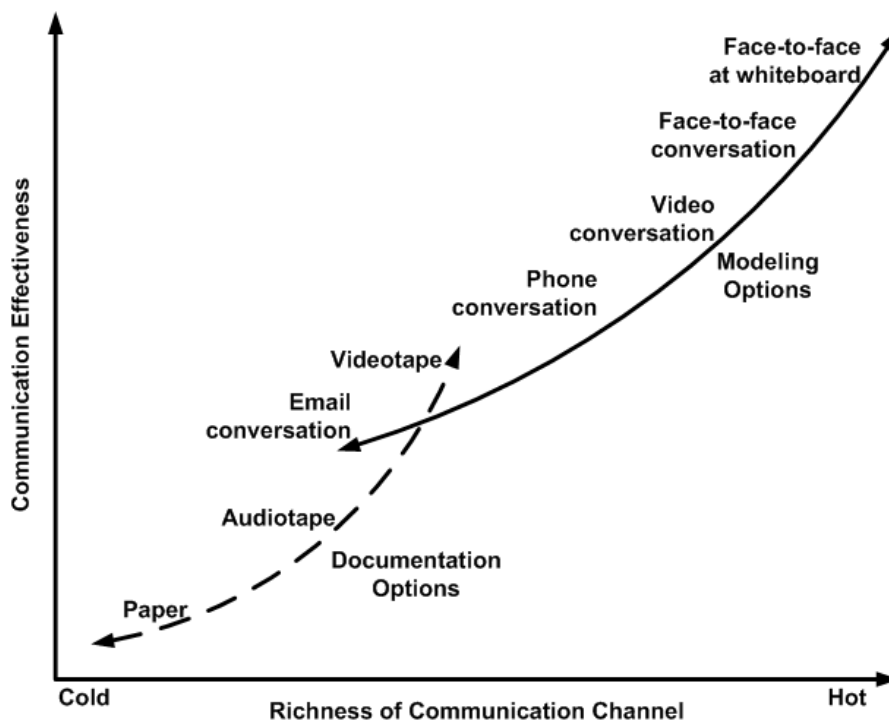
Reflective Improvement

Using the mantra of 'reflect and improve' (Cockburn, 2005) Crystal Clear insists that team get together frequently to discuss new ideas, perhaps, what went wrong, what they experienced and how they felt about these experiences, these get-togethers or workshops need to be frequent, maybe after delivery iterations, honest and free from repercussions. It allows the team to bond,

discuss differing aspects of the project and allows insight into the process various team members have gone through. ‘The specifics of the workshop aren’t nearly as important as the fact that the team are holding one’, however the general purpose of reflective improvement is to share ideas for better working and how to achieve the goals of the project more efficiently.

Osmotic Communication

One of the idiosyncrasies of Crystal Clear is the insistence that the team is located together. Crystal clear goes as far as to prescribe that ideally everyone should be situated in the same room, saying that rooms must be adjoining at the very least. Anything less is outside the scope of Crystal Clear. Cockburn recognises that for a team to succeed they must be a clear and instant channel of communication between, the more informal and immediate the better. In figure ii Cockburn shows us why he feels it so important for teams to physically work together.



Copyright 2002-2005 Scott W. Ambler
Original Diagram Copyright 2002 Alistair Cockburn

Figure ii - How communication falls off

He argues that picking up the telephone, emailing or leaving voicemails for other members of the team degrades the quality of the communication and places barriers in the flow of information.

Cockburn envisages an environment where the whole team is in one room working next to each other, questions shouted out, people commenting on code over another persons' shoulder and problems encountered discussed immediately around a whiteboard.

Getting into the Safety Zone

Frequent delivery, reflective improvement and osmotic communication are the main principles of crystal clear development, however there are four more properties that anchor this methodology, all building on what we've already discussed.

Cockburn talks about Personal Safety, which comes out of osmotic communication, the trust that allows direct, sometimes critical, communication between manager and team. This breeds honest and shorter lines of communication.

Focus is the fifth property that Crystal Clear dictates. The time to spend, uninterrupted, on working on a project Again this may sound self-explanatory but belies the fact that developers are often assigned to multiple projects and endure many interruptions for 'quick fixes'.

Cockburn then moves onto Access to Expert Users', we've already talked about how important it is to have access to an expert user, researchers Keil (Keil & Carmel , 1995) summarised 'get real access to real users'.

Finally Crystal Clear advocates the need for a technical automated test environment that can be used as a robust test environment for the regular iterations that the team is producing. This allows the team to have a central depository for the coded produced, allows integrity and regression to previous versions easily.

Crystal Clear is perhaps unique in that it does not specify exactly how the team works, there is a lot of what Cockburn calls 'tolerance' within the methodology, he even goes as far as to suggest the use of other methodologies such as SCRUM or XP to framework the day to day working. He believes that the

professionalism of the team members coupled with the direct lines of communication that have been fostered will allow the team to reach its goal; As Highsmith puts it ‘process is important but secondary...to communication’ (Highsmith, 2002).

Documentation is also something that Cockburn applies his ‘tolerance’ too. Crystal Clear answers the question of just what documentation is needed by stating “whatever the sponsor and the team decide” (Cockburn, 2005). Crystal Clear believes that the documentation should be produced at the latter stages of the development to ensure that it is up to date and does not include previous iterations. Cockburn goes on to provide pragmatics tips such as converting hand drawn sketches and photographs of whiteboard planning sessions into digital images for use in the final documentation, whilst he doesn’t underestimate the use of modelling languages such as UML he feels that it is not necessarily needed for Crystal Clears ‘light touch’.

Would Crystal Clear be Useful In the Development of The Graduate Directory?

Applying what we have learnt about Crystal Clear, how could we use it in the development of The Graduate Directory?

One of the cornerstones of Crystal Clear is the frequent deliver. I think this would be especially attractive to the project managers of the Graduate Directory, whom all come from a technical background, they would get ‘hands on’ useable code to test, and evaluate, and would not be frightened by early buggy code that was a little ‘rough around the edges’. This would also provide the essential user feedback that is central to the Crystal Clear methodology.

Whilst frequent access to the Graduate Directory Staff may be achievable, we may struggle to get feedback and input from the student users’ who’ll end up using the system and some sort of incentive to encourage participation maybe needed.

Looking at the dynamics of the team and how they work may also provide a challenge for using this methodology. If the software was developed by students or staff, then they may struggle to fulfil Crystal Clears’ strict conditions around how and where the team work together. It is unlikely that a group of students or staff could co-ordinate their timetables enough together to create the osmotic communication that this methodology demands.

Another disadvantage maybe the documentation - or to be specific it’s lack of formality. This may not prove sufficient for a higher-education organisations ‘due diligence’, or dare I say rather bureaucratic approach to, documentation!

Overall I believe one of the biggest advantages that this methodology would have for the Graduate Directory is its high level of tolerance for the actual working methods that can be employed and still be considered to be working within Crystal Clear; this would suit a team of developers that came from different backgrounds and had used differing methodologies. I do think it would require leadership from a project manager to ensure it adhered to Cockburn's principles of reflective improvement and to provide a safe working environment.

Works Cited

Cockburn, A. (2005). *Crystal Clear: A Human Powered Methodolgy for Small Teams*. Addison-Wesley.

Dijkstr, E. (1987). *The Humble Programmer, "ACM Turing Award Lectures: The First 25 Years*. Addison-Wesley.

Highsmith, J. (2002). *Agile software development ecosystems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co.

Hoffer, J. A., George, J. A., & Valacich, J. S. (2008). *Modern Systems Analysis & Design*. Pearson International Edition.

Keil, M., & Carmel , E. (1995, May). *Customer Developer Links*. Retrieved May 18, 2011, from Communications of the ACM:
<http://portal.acm.org/citation.cfm?id=203356.203363&coll=DL&dl=GUIDE&CFID=21989772&CFTOKEN=32770402>

Rusk, J. (2006, Jan 14). *Crysta Clear Methodology*. Retrieved May 01, 2011, from AgileWiki:
<http://www.agilekiwi.com/other/agile/crystal-clear-methodology/>

System Request: The Graduate Directory

Introduction

This document will outline the content of the project in terms of the expected deliverables and business objectives. An inventory of relevant documentation and digital content will be undertaken with the client and presented within this document to define the scope for The Graduate Directory.

Project Sponsors

School of Computing's EXPOTEES Team – represented on the project by Exhibition Director.

Business Practice

EXPOTEES is an annual showcase of students' projects which demonstrate innovation, research and development in a wide range of computing subjects.

Our students undertake an in-depth exploration of a chosen subject area and demonstrate the ability to research, analyse, synthesise, and creatively apply what they have studied. The project is often in an area they have gained an interest in either via a work placement or through their studies. Some students undertake projects which have external clients and require project managing to industry standard.

Exhibition content represents a cross section of our final year work and as such, it demonstrates the full spectrum of subjects taught at the School of Computing.

The event attracts guests from industry, education, business and the public service sector, and provides an amazing opportunity for students to discuss their work with industry professionals and showcase it to potential employers.

Employability is high on the University agenda and EXPOTEES is one of the many ways that the School of Computing endeavors to raise our students' profiles. By exhibiting their work to potential employers many students are successful in securing employment at or through the event.

Some students are proactive in generating their own CVs to give to employers on the day of the exhibition. The employer will also receive a brochure which contains every exhibitors name, photo, project image and project synopsis. If the employer is not able to meet the student on the day then they will contact the University and request a meeting with them. This brochure is also available online on the EXPOTEES website www.tees.ac.uk/expotees.

Inventory

There is already in existence an administration system for final year projects which contains information relating to the aforementioned project and the student. We also have a central registry system which holds details relating to a student's academic record. A separate administration system for EXPOTEES contains only information relating to exhibitors. The Placements Office will hold records of each student's employment during their placement year, if a placement has been undertaken. No live data can be provided due to the Data Protection Act during the development of this system.

A copy of the EXPOTEES brochure can be found on the Downloads section of the EXPOTEES website.

Project Scope and Goals

To improve current practices by resolving the following issues:

- Not all students at the exhibition will create a CV.
- CVs are not always well formatted and cannot include portfolio work.
- Paper CVs are more likely to be misplaced during the event.
- Students who do not take part in the exhibition are not included in the brochure.
- Limited information about the student can be entered into the brochure.
- We cannot include university e-mail addresses in the brochure as they will no longer be active after graduation and personal e-mail addresses are not appropriate as they may change.

Successful completion of this project will resolve the issues outlined above. Specifically it will:

- Provide an online repository to hold up to date information regarding students.
- Capture information about the students' skills list.
- Capture information about any certification/qualifications that the student may possess e.g. PRINCE2 Practitioner, Microsoft Certified Professional etc.
- Provide a facility for holding a student's employment history, focussing upon the placement year if undertaken by the student.
- Link to a student's portfolio of work, e.g. samples of animations, websites or applications generated by the student.
- All entry of endorsements (references) by teaching staff or previous employers.
- List the students' interests in terms of future work e.g. Is interested in.....consultancy/freelance work/web development etc
- Provide a mechanism for all students to access and update their own skills list. Students will not have access to anyone else's skills profile.
- Provide a search facility for use by employers.

This online repository The Graduate Directory will be launched at the next EXPOTEES event.

Data Model

The following data model has been provided, but this data model needs further development. Identity **3 more** entities and add to this data model.

