**1 Project Proposal:**

**Automatic stock management system for a maintenance company**

The proposed project is to design and implement a stock management system for a maintenance company.

Client: Automobile spares ltd (fictional company)

Time Scale: Working system to be delivered 01/04/2014

**Current system:**

Currently the company uses a very manual process to manage spares. Current stock levels are recorded by hand in a stock book. When a user wants to take a part they must look through the stock book to find the location of the item, sign the item out and then manually order a replacement.

This system is laborious and prone to human error. If a person forgets to order the replacement part the stock level will remain low.

**Proposed system:**

The system proposed to replace the current system will add elements of automation and improve usability to avoid or remove the possibility of human error. The proposed system will be based on an android device, probably a tablet located in the store room. The new system will be an android application that give uses an SQLite database. It will have the following functionality / use cases:

1. Allow the user to add an item to stock
2. Allow the user to remove an item from stock and book it to a particular cost code / part of the company.
3. Allow the user to view the current level of stock
4. Automatically replace taken stock if the levels fall below a certain threshold
5. Do statistical analysis on the stock usage, ie mean time between failure, which part of the company is using the most stock
6. Interact with the users to inform about current stock levels
7. Export the database in SQL format for future system upgrades

**The market:**

This project is initially proposed to address a problem in a particular company, however there is also a place in the market for this. There are very few applications on the market that do what is set out in the proposal. There is one application that comes close call Stock Manager from Amalgame. This application doesn't do any statistical analysis and doesn't do any automated ordering. Therefore there is a possible market for this application beyond just the current client.

**Benefits:**

The proposed system has the following benefits over the current system:

1. Reduced labor costs due to time saving
   1. Stock levels can be quickly accessed
   2. Location of stock can be pin pointed instantly
   3. Orders are placed with the supplier automatically
2. Reduced running costs
   1. Cost of stationary eliminated
   2. required storage space reduced
3. Accuracy of the system improved
   1. Entries are always legible as they are typed in
   2. Stock levels are constantly maintained and replacement parts are never forgotten.
4. Additional information for management purposes
   1. Statistical analysis of the data can be perform to predict failures, future budgets, unreliable parts, unreliable usage of parts.

**Costs:**

The proposed project has the following costs to consider:

1. Development costs
   1. Software engineering costs
2. Setup costs
   1. Initial hardware costs for the android tablet
   2. Initial hardware costs for a machine to develop the software on. (this is just a pc, the android development kit is free and most companies already have development computers).
   3. Training the staff to use the new system.
   4. Installation costs for the android tablet (Secure enclosure and power supply)
3. Operational costs
   1. Negligible power consumption costs
   2. Software maintenance if clients want upgrades or find bugs.

**Relevant literature review:**

The following items of literature are relevant to this topic and help determine how technically complex the project may be and therefore potentially how feasible it may be in the give time constraint:

Android databases and content provider – Lars Vogel 22.12.2010

<http://www.vogella.com/articles/AndroidSQLite/article.html>

This article goes into details on how to implement an SQL database in android in the form of a content provider. By reading this article we can assess if the technical ability of the student to undertake this project is sufficient. The article is in the form of a tutorial and as such has no biased opinion, the author doesn't express any opinions and presents the article in a very formal way.

Inventory software top ten reviews – Tech media network 2013

[inventory-software-review.toptenreviews.com](file:///C:/Documents and Settings/user/My Documents/Dropbox/Uni/Sep 2013/Software Project Management/Assignment1/Complete/documents-export-2013-10-18/inventory-software-review.toptenreviews.com)

This article addresses the possible required functionality of current high end PC based systems. It presents ideas for additional functionality but probably isn't that useful. Unused functions mean wasted time and money and more maintenance costs. It would be advised to possibly mention some of these points at the requirement specification stage to the customer and maybe get some of them added at that stage.

Christine Connolly, (2008) "Warehouse management technologies", Sensor Review, Vol. 28 Iss: 2, pp.108 - 114   
<http://www.emeraldinsight.com/journals.htm?issn=0260-2288&volume=28&issue=2&articleid=1714557&show=html>

This is an article from a journal which comes highly regarded internationally shows the overview of different technologies used in warehouse stock rooms from radio-frequency product-labelling technologies to stock control databases.

The article is now a few years old but these are still the technologies being used currently, we’re trying to advance with technology by making a stock system available on smart phones and tablets, which are widely used. First at a smaller scale for our client (local engineering company) but hopefully eventually advance through the industry.

**Outline Methodology:**

As this project is to be undertaken by a student in higher education it would be safe to assume that this project in itself will be a learning experience. With this in mind an iterative methodology would be the most suitable, in particular the spiral method. More detail on why this method is the most suitable can be found in section 6.

**Quality Assurance:**

It is important to implement some kind of quality assurance in any project. Please see section 4.

**Resources Required:**

The following resources will be required to complete the project:

1. A development PC
2. An android tablet

**4 Quality assurance**

Quality is hard to define and measure as quality can be subjective. There are two criteria to consider when addressing quality:

>Qualitative- Something which can’t really be measured and is somewhat down to individual opinion. e.g. looks good? Value for money etc (subjective properties)

> Quantitative this can help measure quality as it is usually a statistic/ measurement. By having measurements allows you to have an acceptable quality level? (Objective properties)

A common theme if the end product is ‘quality’ is if the software has turned out as expected, met specification, user requirements, legislation etc.

Looking at our project we looked at quality factors (external view) related to the example quality criteria (measurable criteria) to make sure our app for the engineering company will be to standards expected.

|  |  |
| --- | --- |
| **Factors** | **Quality criteria** |
| **Software Build** -Correctness | **Traceability-** The app should store who has made changes i.e. Added Stock, Made an order etc, so if errors occur or problems occur, the item record can be restored/ staff consulted. |
| **Consistency –** Item locking to make sure, if software is accessed at the same time, updates won’t be lost. |
| **Completeness –** have required fields for adding new items to stock system. |
| -Reliability | **Fault tolerance**- What happens if the app breaks ortablet system goes down? Have a backup so the stock system can carry on working without any issues. |
| **Accuracy-** Validation and error messages to allow the app to be run smoothly |
| **Simplicity-** Avoid overcrowding of information, clear and easy to read, simple GUI/interface, simple commands. |
| -Efficiency | **CPU, Disk and Memory efficiency –** Any temporary storage to wipe, when app is closed. Limit fields and only store necessary information. |
| -Integrity | **Access control-** As there is no personal data stored in the app, security isn’t necessary but you don’t want unauthorised staff changing information. A pass code would be required to edit or delete data. |
| -Usability | **Operability-** Simple and easy to use . |
| **User friendliness-** Simple GUIS/Interface, Help available through the app, Simple instructions and buttons. Allow change of font sizes. |
| **Late stages of software development -**Maintainability |  |
| **Self-descriptiveness-** Keep the system simple and easy to use, which anyone working in a engineering company would be able to use. Clear, Simple English. |
| -Testability | **Testing-** The app should be fully tested before being released and installed, to fix any bugs which may occur. Beta testing by proposed clients, to see the app working in the environment it is expected too. Test the system against the research, aims and objectives, requirement specification. Create test plan to see if apps gain expected results and navigation work as excepted etc. |
| -Flexibility | **Modularity –** The nature of android as a development platform is that it is very modular. The application must be designed to exploit the activity segregation that is inherent in the android development platform. Where possible implicit intents must be used for system calls. |
|  |
| **Expandability-** The app is currently for a local engineering company but eventually the aim is to expand so it will be in use in most warehouses which require stock control. Use on android smart phones and conversation to work on Apple products too. |
| **The apps future**  -Portability | **Machine independence**- The software can be installed on any tablet which runs android. And can be used anywhere in the companies warehouse as it will be installed on a tablet. |
| -Reusability | **Software system independence-** Can be installed on any android tablet. To make sure the system stays reliable when on more than one system, record lock will allow this. |
| -Interoperability | **Interface commonality-** In this case the database is in SQL so it is important that it can be exported in an SQL format for other applications to manipulate data |
| **Data commonality-** Consistency between stock and system**.** |

**5 Risk Analysis**

**There are many variations of risk management workflows but the key features will feature in all are: Identification, Priority, Planning and Monitoring**

**Identification-** Below we have identified the risks we believe create a threat in the development of our app, and listed them in a project register. Success/ By? Would be filled in IF the risk actually occurred and action was necessary. The greatest impact and probability have been put in **bold**.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Risk** | **Description** | **Impact** | **Probability of occurrence** | **Severity** | **Category** | **Risk- Action** | **Success? By?** |
| **1** | **Late changes to requirements.** | **System changes will need to be made accordingly otherwise the end product won’t be to the clients’ needs. Could be unworkable.**  **Delays to finish date and increase in cost.** | **Likely (7)** | **7** | **High** | **Contingency plan (REDUCE)- Incremental development and prototyping. This is why we will be using the spiral model. Gives the client the chance to adapt the system.**  **Build time in to schedule** |  |
| **2** | Delay to hardware availability to start coding | Delay to expected finish as implementation cannot take place till hardware is available. | Unlikely (3) | 4 | Medium | Contingency plan (REDUCE) - Build in time schedule. Hardware should be being prepared during the initial stages, given if there is still a delay it would be ready in time for implementation. |  |
| **3** | Android tablet unavailable | Delay to final finished time. As testing and installation couldn’t take place. | Very Unlikely (1) | 10 | Low | RETAIN- As there is a low chance of this happening as android tablets are widely available. Also this shouldn’t be an issue especially as installation could take place on a smart phone running android. |  |
| **4** | Budget overrun | Whole project could come to a standstill | Very unlikely (1) | 10 | Low | Contingency plan (REDUCE) - Where can costs be cut?  -Use similar projects to base cost and analysis (Cost benefit analysis) |  |
| **5** | **Part of software engineering team fall ill** | **Delay to project, depending on how serious.** | **Likely (8)** | **3** | **Medium** | **Contingency plan (REDUCE) - Can someone else take over there role?**  **Risk built into schedule** |  |
| **6** | **Development technically to hard (Poor project planning/ feasibility study)** | **Delay to overall project, dependant on how serious could bring project to a stop.** | **Unlikely (3)** | **8** | **Medium** | **Contingency plan (REDUCE)- Training?** |  |
| **7** | Rival company launch similar product | Reduces the need of the app. As the app will be already available. | Very unlikely (1) | 5 | Low | Contingency plan (REDUCE)- Monitor competitors i.e. Amalgame |  |
| **8** | New version of android released | Could change requirements and design if features have changed for development in new release | Very Likely (10) | 2 | Medium | Contingency plan (REDUCE)- Software engineering team should be equiped to deal with changes, training maybe required. |  |
| **9** | Finished app not as expected | App becomes unusable | Unlikely (2) | 10 | Medium | Retain- It has high consequences but as we’re using the spiral model this should never happen as prototypes are created for clients to review. |  |
| **11** | Software engineers never implemented an android app | Software production time could be extended | Very unlikely (1) | 8 | Low | Contingency plan (REDUCE)- Training during initial stages where coders are not required |  |
| **11** | Client wants app for Apple Products (ITunes) | Huge delays as software team could have never worked with Apps for Apple. May not be possible | Very unlikely (1) | 10 | Low | Avoid- If software team does not know how to implement a app for apple. Avoiding would be the last resort; first training would be built into a contingency plan. |  |

**Priority-** Above we look at the probability of occurrence, severity and category as this allows us to see which risks are likely to occur. To determine the chances we used a risk matrix.

(Yardley, D, 2002)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Severity of Impact | Probability of event occurring | | | |
|  | Very Unlikely < 10% (1) | Unlikely 10%-50% (2-5) | Likely 50%-90% (5-9) | Very Likely >90% (10) |
| Low (1-2) Minor Impact. No impact on benefits. | Low | Low | Low | Medium |
| Medium (3-5) Major impacts. Some impacts on benefits | Low | Medium | Medium | High |
| High (6-10) Major impact on benefits and project may longer be viable | Low | Medium | High | High |

Using the risk matrix we also worked out the risk exposure = (**Probability of occurrence X Severity).**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability Occurrence** | **Severity** | **Risk Exposure** |
| **1** | **7** | **7** | **49** |
| 2 | 3 | 4 | 12 |
| 3 | 1 | 10 | 10 |
| 4 | 1 | 10 | 10 |
| **5** | **8** | **3** | **24** |
| **6** | **3** | **8** | **24** |
| 7 | 1 | 5 | 5 |
| 8 | 10 | 2 | 20 |
| 9 | 2 | 10 | 20 |
| 10 | 1 | 8 | 8 |
| 11 | 1 | 10 | 10 |

**Planning-** In the risk register the Risk Action is the planning we have decided for each risk we have identified how to deal with them with Avoiding, Reducing (contingency plan), Retain or transfer. See project planning to see us taking risks in to account for our project.

**Monitoring-** We will repeat the Identification and planning throughout the process of building the project this will allow us to check that original risks have been dealt with and that new ones have been observed/noted.

**6 Methodology**

As was mentioned earlier in the project proposal we have chosen to use the spiral model.

Spiral Model

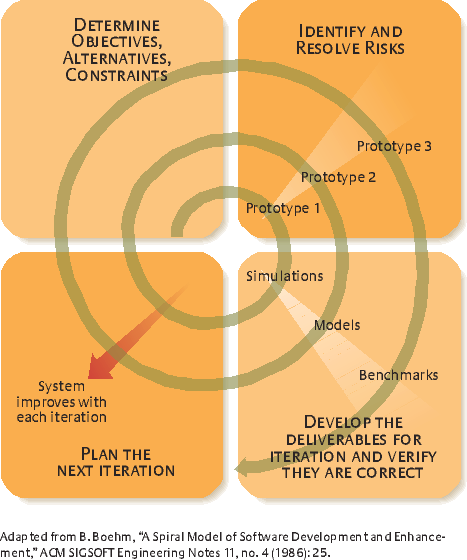


Figure 1 (Rudolf Kutina 26/08/2010 http://qualityguru.wordpress.com/2010/08/26/evolutionary-sw-development-spiral-models/)

The reason we chose the spiral model is because it aims to avoid abandoning the project half way through after realizing that the system benefits will be less than its cost. At each phase well-thought-out details are carried out and the possibility that the project will be successful can be justified (Hughes and Cotterell, 2006, p.77). Risks are taken into consideration and requirements and design are developed in a series of prototype (M. Younas 2013). Each loop in a spiral model follows 4 steps as indicated by the diagram above.

1. Determine goals, alternatives and constraints

2. Evaluate alternatives and risks

3. Develop and test

4. Plan for next phase

At the end of each loop an evaluation is done before the next iteration starts, this will be good because we get feedback from the user and anything that needs to be adjusted can be done at this stage hence another reason for our choice, client is involved and we do not have to wait until the real software is developed to realize that it is not what the user wanted. Once the user is satisfied the software can then be developed with phases following the waterfall model. So we can say the fact that risks are taken into consideration, the user is involved during development and with this model you can learn as you develop the system were our main drivers for choosing this model.

We did not choose the waterfall method because with this model you have to finish one phase before you move on to the next one, user requirements have to be well defined and the development methods have to be well understood (M. Younas 2013). Waterfall has its advantages which are it avoids rework as every phase is looked at thoroughly and it avoids reopening activities which were completed as this can mess with deadlines (Hughes and Cotterell, 2006, p.75).

The disadvantages of waterfall are that most likely users will change their requirements during development and it will be hard to go back to an earlier stage although it is possible to do it, this will cause the project to go over budget and most likely to be late (M Younas 2013). It is mostly used for projects that are low risk so in our high risk project it will not be appropriate. It is suitable for projects where requirements are clear, known and unlikely to change but in our case we are not sure that the requirements will not change so to be safe we chose the spiral where we will develop a prototype and find out if that’s what the user want.

We did not go with the Rapid Application Development (RAD) also known as prototype, because of some of its disadvantages. There are two types of RAD, throwaway and evolution. With the throwaway approach the final system can be delayed as the first prototypes that were developed get thrown away “when the true development of the operational system is commenced”, (M. Younas, 2013, Software project management, week 2). With the evolutionary approach the first prototype will be refined until it can be the operational system therefore “standards that are used has to develop the software has to be carefully considered”, (Hughes and Cotterell p.78). It is harder to control budgets and resources as the development can be repeated so many times. There is a risk that the final system can easily drift away from the original user requirements as so many changes are done.

Despite these disadvantages there are a lot of advantages with the RAD method, the user is heavily involved in the development process and the requirements are made clearer, usability of the system is improved as the system is tested and them modified if there is need to, there is improved communication between the client and the developers and maintenance costs are reduced (Hughes and Cotterell p.79). It is a learning experience just like the spiral because prototypes are developed first before the final system.

Despite RAD having some similarities with the spiral model we chose the spiral model because once the client is satisfied it then follows the waterfall model which involves coding, integration and testing, acceptance test and implementation and to us these steps are important.

**7 Teamwork report**

Individual Roles:

Lee Hudson - The Proposal, Work breakdown structure, research on the decided topic, compile the whole document.

Vishane Perera - Quality Assurance, Risk Analysis, Work breakdown structure, research on the decided topic.

Agnes Chinyoka - Methodology, Work breakdown structure, research on the decided topic, Write up the group report.

Josh Wong - PERT and Gantt charts, Work breakdown structure, research on the decided topic.

After group allocation we emailed each other and decided to meet during seminar time so that we can introduce ourselves and map a way forward with the assignment. All four group members were present. After introductions we started to make suggestions on a suitable topic, but nothing seemed interesting so we decided to go and think of more topics. We were going to use emails as our main communication source but we also exchanged phone numbers. As one of our team member’s works all week and can only meet on Mondays’ we decided to use Google docs for collaboration. Before we left we scheduled our next meeting for the 7th of October. The meeting lasted 30 minutes and later minutes were uploaded on Google docs.

During the week we decided on a topic and tasks were going to be allocated on our next meeting. We agreed that we all do some research on the agreed topic prior to meeting on the 7th. We met on Monday the 7th, 3 group members attended and 1 group member send his apologies. The agenda for this meeting was general discussion on the topic, discuss our findings and allocate tasks. Tasks were allocated and two tasks were left over so we were going to discuss them on the next meeting. We agreed that after completing your task you had to upload it onto Google docs and by our next meeting all allocated tasks should be uploaded. The meeting lasted 30 minutes and our next meeting was scheduled for the 14th of October. Minutes were uploaded on docs. We agreed that Lee, who was doing the proposal should do his part first as this was going to guide other team members.

When we met on the 14th, the proposal, risk analysis and methodology were done. Agenda for this meeting was to discuss the work breakdown structure so that Josh can go ahead and work on the PERT and Gantt chart. All group members worked on the work breakdown sketch and Lee was going to write it up. Vishane was going to do the quality assurance and Agnes was going to put together the group report. We had a group deadline to upload the all the documents on Google docs by Saturday the 19th of October, Lee was going to compile the document and all team members were going to proof read it and make an adjustments. This meeting lasted 1 hour. Besides our online meeting during the week our next physical meeting was going to on the 21st of October. Agenda for this meeting was to make sure everything was in place, proof read our document and print it ready for submission.

References:

Hughes, B. and Cotterell, M. (2006). *Software project management.*4th ed

M. Younas (2013) Software project management Module U08784 Session materials Week 2, Semester 1. Oxford: Oxford Brookes University.

Rudolf Kutina 26/08/2010 http://qualityguru.wordpress.com/2010/08/26/evolutionary-sw-development-spiral-models/

**References:**

***Yardley, D.*** *(2002) Successful IT Project Delivery: Learning the Lessons of Project Failure. London: Addison Wesley. p.p.148*

***Hughes, B. and Cotterell, M.*** *(2006). Software project management.4th e**d*

***M. Younas*** *(2013) Software project management Module U08784 Session materials Week 2, Semester 1.*

***Rudolf Kutina*** *26/08/2010* [*http://qualityguru.wordpress.com/2010/08/26/evolutionary-sw-development-spiral-models/*](http://qualityguru.wordpress.com/2010/08/26/evolutionary-sw-development-spiral-models/)

Android databases and content provider – Lars Vogel 22.12.2010

<http://www.vogella.com/articles/AndroidSQLite/article.html>

Inventory software top ten reviews – Tech media network 2013

[inventory-software-review.toptenreviews.com](file:///C:/Documents and Settings/user/My Documents/Dropbox/Uni/Sep 2013/Software Project Management/Assignment1/Complete/documents-export-2013-10-18/inventory-software-review.toptenreviews.com)

*Christine Connolly, (2008) "Warehouse management technologies", Sensor Review, Vol. 28 Iss: 2, pp.108 - 114*[*http://www.emeraldinsight.com/journals.htm?issn=0260-2288&volume=28&issue=2&articleid=1714557&show=html*](http://www.emeraldinsight.com/journals.htm?issn=0260-2288&volume=28&issue=2&articleid=1714557&show=html)