

**E-HEALTH SYSTEM FOR
PRENATAL SCREENING
(PrenatalLink)
Courage Chibiya
BSc Computer Science (Industry)**

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material that is obtained from another source may be considered as plagiarism.

(Signature of student) _____

Summary

Prenatal Screening covers the range of procedures that are available to pregnant women in order to find out those at risk of having a child that is affected by congenital disorders. Prenatal screening software serves to calculate the likelihood of such a risk based on clinical tests results and pregnancy related factors.

Although it is possible to do such calculations manually, the calculations involved are too complex to be done routinely by a physician. Advances in prenatal screening have now reached a level where physicians would like to carry out prenatal screening using highly efficient, internet enabled software systems. Up until now, prenatal screening has not been fully available on the web (Shaffiq, 2004) and current systems do not meet user needs such as being:

- Configurable and incorporating the latest research findings on biochemical and ultrasound markers.
- Scalable for networked applications and internet enabled.
- Able to give multi-stage results and interpret them for the user.

This project aims to develop a web system for calculating the risk of a pregnancy being affected by a fetal anomaly. The system will use clinical test results and pregnancy related information to calculate the risk of a pregnancy being affected by a selection of chromosomal disorders. Based on the calculated risk value, the system will interpret whether a pregnancy is affected or not using population risk cut-off parameters. This project aims to produce a tool that automates the risk calculation process for physicians.

Derived objectives of the project are to produce a solution that fulfils the current needs of physicians to have a system that is internet enabled, highly configurable and that produces multi-stage results. The final solution would also relieve physicians from carrying out of complex analysis and interpretation of results.

Acknowledgements

This project was proposed by Media Innovations Ltd, a company that specialises in developing medical diagnosis software. Without their involvement and support, this project would not have been a success. I would like to express my utmost gratitude to the company and their representatives for their support and for enabling me to develop the final solution.

I would also like to give thanks to the following people for their suggestions and input during the compilation of the system requirements and the project lifecycle.

- First and foremost I would like to thank Mrs Carol Wilson, this project's supervisor at Media Innovations Ltd. She was crucial to the success of this project and provided invaluable help, advice and encouragement.
- Many thanks to Dr S R Roberts, my University of Leeds project supervisor for his constructive advice and helpful suggestions.
- And finally thanks to my friends Marion Braun, Mathew Bosquet and Carmen Gonzalez for their support throughout this project.

A considerable number of people were involved in requirements analysis, testing and evaluation of this project. I would like to express my deepest gratitude to all those who have not been mentioned here. It is with great appreciation that I acknowledge the information, counsel and support you provided.

Contents

Summary	ii
Acknowledgements	iii
1 Introduction.....	1
1.1 The Problem Domain	1
1.2 Media Innovations Ltd	2
1.3 Project Aim	3
1.4 Project Objectives	3
1.5 Minimum Requirements	3
1.6 Deliverables	4
1.7 Schedule	4
1.8 Relevance to Degree	5
1.9 Evaluation Criteria	5
1.10 Report Structure	6
2 Background	7
2.1 Screening Theory	7
2.2 Congenital Disorders.....	7
2.3 Screening and Diagnostic Methods.....	8
2.4 Reasons for Prenatal Screening.....	8
2.5 Prenatal Risk Calculation.....	9
2.6 Project Methodology.....	11
3 Analysis	13
3.1 Considerations.....	13
3.2 Constraints	14
3.3 Stakeholders	14
3.4 Feasibility Analysis.....	14
3.5 Risk analysis	16
3.6 Existing systems.....	17

4	System Requirements	18
4.1	Elicitation Techniques.....	18
4.2	Requirements Gathering	19
4.3	Functional and Data Requirements	20
4.4	Non-functional Requirements	20
4.5	External Interface Requirements.....	21
4.6	Requirements Evaluation	22
5	Methodology and Development Technologies	23
5.1	Development Technologies.....	23
5.2	Programming Languages	25
5.3	Scripting languages.....	26
6	Design.....	27
6.1	Design Pattern.....	27
6.2	Architectural Design	28
6.3	Elipse Object.....	30
6.4	User Interface (View) Design	30
6.5	Controller Design.....	31
6.6	Security	31
6.7	Database Design.....	32
6.8	Development Plan	34
6.9	Design Review	35
7	Implementation	36
7.1	Database.....	36
7.2	User Interface.....	36
7.3	Security	37
7.4	Data Entry and Validation.....	37
7.5	Development Summary.....	38
7.6	How the System Works.....	39

7.7	Reporting System.....	40
8	Testing.....	41
8.1	Code Testing.....	41
8.2	Testing Results.....	42
9	Evaluation.....	43
9.1	Evaluation Criteria.....	43
9.2	Functionality.....	44
9.3	Reliability.....	46
9.4	Efficiency.....	46
9.5	Usability and Accessibility.....	47
9.6	Portability and Maintainability.....	47
9.7	Comparison to existing systems.....	47
9.8	User evaluation.....	48
9.9	Evaluation Results.....	48
9.10	Suggested improvements.....	48
10	Conclusion.....	50
11	Bibliography.....	51
12	Appendix Table of Contents.....	55

1 Introduction

Prenatal screening covers the procedures that are available to pregnant women in order to find out those at risk of having a child that is affected by congenital disorders, especially chromosomal abnormalities such as Down syndrome. Screening is mainly carried out using biochemical markers (hormones) or ultrasound (sound waves) techniques. Recent developments in prenatal screening such as new ultrasound and biochemical markers and the generation of likelihood values at different stages of the calculation process are not available in existing software (Wilson, 2006). In addition, as screening becomes more widespread, screening centres increasingly wish to be able to use highly advanced systems that allow them to calculate risk results from wherever they may be in the world. The aim of this project is to develop such a system for calculating the risk likelihood value.

1.1 The Problem Domain

Over the past two decades, prenatal screening for fetal defects has become a standard part of prenatal care (Kristol, 1993). Tests conducted during the first trimester of pregnancy are designed to detect a wide range of genetic and other disorders and to give prospective parents the option of obtaining abortions if defects are diagnosed (Cuckle, 1997). This development has been heralded as a breakthrough in the age-old battle against illness. Others consider it as more than that: an instrument to improve society. There are invaluable benefits to screening, lives can be saved or improved and major cost savings incurred by the health system and families. Like any major medical development likely to occur in our age, prenatal screening has its own share of controversial consequences. The issues surrounding the morality of screening are the stuff of moral philosophy (Kristol, 1993) and agreement on the acceptable social consequences is difficult to achieve, however that should not obscure the benefits that can be gained from prenatal screening.

The most common form of prenatal screening is ultrasound screening; about 80% of all pregnant women receive ultrasound screening (Jones, 2005). This method uses sound waves to generate a sonogram or image of the fetus in the womb. The main purpose of this test is to establish the gestational age of the fetus and in addition other defects such as malformed organs might be noticed. There are several indicators that can be found during ultrasound screening which have a significant association with chromosomal abnormalities. These “markers” in combination with other biochemical markers can be used to discriminate between unaffected fetuses and positive ones (Smith-Bindman, 2001). Examples of ultrasound markers are nuchal translucency (NT - the thickness of the nuchal area at the back of the neck) and neuroblastoma (NB), a cancer that affects nerve cells of immature or developing cells such as those found in a fetus.

In addition to ultrasound screening, physicians carry out serum screening - a procedure in which they check a mother’s blood to obtain certain measurements such as alpha-fetoprotein (AFP), unconjugated estriol (UE3) and human chorionic gonadotropin (HCG). The blood test results of these independent

“markers” together with the fetus’ gestational age and other risk factors related to the pregnancy can be used to calculate the risk of a fetus having congenital disorders or chromosomal anomalies.

Complex risk calculation procedures are then used to determine the likelihood of a pregnancy being affected based on the clinical test results, gestation, population parameters and other factors related to the mother (Wald et al, 2004). If the results are positive (justifying further investigation), more dangerous, definitive and invasive tests are then carried out before prospective courses of actions are considered.

The broad scope of this project is to respond to the present need for software systems that are configurable, available on the internet and incorporating some of the latest research findings on ultrasound and serum markers. A system that produces results at each major stage of risk calculation instead of one final risk value and hides the complexity of result interpretation from the user (Wilson, 2006). There has been widespread need for provision of screening services from the internet with the aim of enabling clinicians to utilise the advantages of web based systems (Jones, 2005). The delivery of prenatal screening systems that allow clinicians to configure calculation parameters and customise them to local population averages would be a useful development in the prenatal screening industry. The aim of this project is to deliver such a system, a system that will be web based and incorporating some of the latest technological advances in prenatal screening. The system will serve both clinical and research interests of clinicians due to its highly configurable functionality. This project will be developed in conjunction with Media Innovations Ltd and they will be involved in every stage of this project’s lifecycle.

1.2 Media Innovations Ltd

Media Innovation Ltd (MI) is a subsidiary company of the University of Leeds. It is a leading developer of multimedia training materials across a wide range of industry and professional sectors. It is also a significant developer of Training Systems, Risk Management and Diagnostic Support Tools to the medical profession (Isaacs, 2004).

Some of the software products that are developed by MI are based on research projects of its own, the University and other consultants. Its market is made up of the general public, healthcare professionals, prenatal screening centres and other companies such as Perkin Elmer Ltd, one of the leading developer and distributor of healthcare software in the world. The MI range of products can be divided into two categories: multimedia and scientific software products. Most of the multimedia software is designed for use as medical self-help training materials while the scientific software products are principally medical risk management and diagnostic support tools.

1.2.1 Organisational Structure

MI is made up of three departments: Scientific Software, Multimedia and Textiles departments. The Multimedia department specialises in developing self-help training materials. The Scientific Software department develops (medical) risk management software and diagnostic support tools. The textiles

department develops self-learning products and e-learning systems on various textiles and apparel subjects.

This project was proposed by the scientific software department of MI. The department assigned a project supervisor who was responsible for overseeing this project. The supervisor was actively involved in the all stages of the project lifecycle, final user acceptance testing and user evaluation.

1.3 Project Aim

The main aim of this project is to develop a web application that will accept maternal, biochemical and ultrasound test results from clinicians and calculate the likelihood of the given pregnancy being affected by congenital disorders, especially those caused by chromosomal anomalies.

1.4 Project Objectives

The objectives of this project are as follows:

- Research the theory of screening and the causes of congenital disorders such as genetic malformations and evaluate the benefits of prenatal screening.
- Evaluate existing prenatal screening systems, noting any useful features that could be implemented into the final solution and weak points to capitalise on.
- Elicit requirements from stakeholders and develop a requirements specification.
- Evaluate software development methodologies and technologies that could be used to develop the application and choose the most suitable ones with justification.
- Design and develop a software application that meets at least all the minimum requirements and some of the requirements extensions if possible.
- Test and evaluate the developed application against a test plan and the evaluation criteria.

This project aims to produce a beta test version of the final system, further independent evaluation and testing by MI and external evaluators will be conducted before the system can be used clinically. However during the external evaluation, the beta system will be available for non-clinical use to registered physician for evaluation and testing purposes. It will also serve to advertise the system before the release of the clinically validated alpha version.

1.5 Minimum Requirements

The minimum requirements of the project and their possible extensions are:

Minimum Requirement	Possible Extension
A commercial web application for calculating risk results	Batch calculation system that allows users to calculate results offline
Database system for data storage	Import and export functions that enable users to easily upload input from files

Data input and results reporting systems	An advanced data input system that allows users to enter dates using date pickers etc.
Security system for the application	An automatic client side input validation system and session tracking system
Help system delivered in the form of a web page	An integrated help system that uses compiled help pages or case sensitive functionality

Table 1 - Minimum Requirements and Extensions

Possible additions to the minimum requirements and extensions are:

- Page caching functionality that stores static elements of the applications' web pages on the client-side to improve performance
- Enabling the application to be usable from Internet Explorer, Netscape and Fire fox browsers
- Internationalising the application and making it customise to the user's locale and display dates in the user's language

1.6 Deliverables

Deliverable name	Description	Delivery date
Mid term report	A progress report and background chapter	09 December 2006
User manuals	A user guide describing how to use the system	18 March 2006
Technical documents	Documents explaining the system's components and other technical implementation information	07 March 2006
Application (Eclipse) and Database	The final product and the database used to store clinical data for use in calculations	01 March 2006
Testing documents	Document describing the validation and testing processes of the system	25 March 2006
Final project report	A report on this project that documents the project lifecycle	25 April 2006

Table 2 - Project Deliverables

1.7 Schedule

A detailed project schedule is included in Appendix C - Introduction; this section gives an overview of the project schedule.

Tasks	Implementation Period
Relevant research, background reading and planning	27 Sep 2005 - 01 Jan 2005
Report Planning and Write-Up (Incremental)	12 Nov 2006 – 15 Apr 2006
Inception and Requirements Analysis	13 Nov 2005 – 26 Nov 2005

Methodology and Design Specification	27 Nov 2005 - 15 Dec 2005
Implementation	15 Jan 2005 – 24 Mar 2006
Verification and Validation	05 Mar 2006 – 18 Mar 2006
Review and Evaluation	19 Mar 2006 – 24 Mar 2006
Release Packaging & Report Finalisation	24 Mar 2006 – 08 Apr 2006
Appendix and Glossary Compilation	16 Apr 2006 – 20 Apr 2006
Reflection and Project Review	15 Mar 2006 – 21 Mar 2006

Table 3 - Schedule Overview

1.7.1 Revised schedule

Based on advice from the project supervisor and the feedback from the mid-project report, the developer decided to increase the time allocated for background reading to ensure he gained extensive knowledge of available solutions to problems before embarking on a task. In addition, the developer increased the time allocated for writing up appendices and for testing in order to spare time for resolving errors and for contingency. It became apparent that more time was vital during implementation to write-up quality implementation documents. Table 4 shows the revised schedule.

Rescheduled Tasks	Implementation Period
Relevant research and background reading	27 Sep 2005 - 01 Jan 2005
Report Planning and Write-Up (Incremental)	12 Nov 2006 – 22 Apr 2006
Methodology and Design Specification	27 Nov 2005 - 15 Dec 2005
Implementation	15 Jan 2005 – 24 Mar 2006
Verification and Validation	05 Mar 2006 – 18 Mar 2006
Release Packaging & Report Finalisation	24 Mar 2006 – 08 Apr 2006
Appendix and Glossary Compilation	16 Apr 2006 – 25 Apr 2006
Reflection and Project Review	15 Mar 2006 – 21 Mar 2006

Table 4 - Revised Schedule

1.8 Relevance to Degree

This project will require the developer to utilise a wide range of knowledge gained throughout his degree program. The proposed solution involves considerable planning, programming and report writing. Furthermore extensive application of technologies studied will be required; examples include databases, internet/distributed systems, software project development and many others. The developer will also use skills acquired during his degree such as team working and research techniques. This assessment shows that the project is of substantial relevance to his degree program and will enhance or enable them to learn new skills and technologies which may be useful later in his career.

1.9 Evaluation Criteria

The success of this project will be judged on completion. To ensure this can be done successfully, a set of evaluation criteria was drawn up and these are:

1. Does the system fulfil the project objectives and fulfil the project aim?

This criterion evaluates how well the system meets its objectives and aims. The success of the system is based on how well the delivered features satisfy the project objectives.

2. Have the specified minimum requirements been met and extended?

The second criterion judges how well the system satisfies the user requirements and how these were expanded upon. MI have to benefit from the business value of the system.

3. Does the system calculate correct results and function efficiently?

The third criterion will be assessed during the testing and evaluation processes. Testing aims to validate the correctness of the system results and the level of user satisfaction with the final solution.

4. How does the system compare to existing systems?

The system is judged by how well it compares to existing system, it must bring new benefits to stakeholders to justify its creation.

5. Does the system meet the usability requirements and development guidelines?

Although it is important to give priority to system functionality, its usability is of great importance because it determines acceptance and adoption levels. This criterion is a heuristic evaluation of the application and checks its usability, accessibility and conformance properties.

6. Has the project been delivered on time and passed user acceptance testing and evaluation?

The project must meet the schedule requirements specified in the project plan to be successful. If a project is delivered late, the required functionality may have changed and this might lead to loss of customers and project failure.

In addition, the system will be evaluated according to the ISO 9126 software evaluation criteria, section 9.1 provides a detailed explanation of this criteria.

1.10 Report Structure

The structure of this report mirrors the project development process itself. Most chapters begin with a summarisation of the background research carried out. The next chapter explains the problem domain in detail and justifies the methodology used in solving the problem. It is followed by an analysis of current systems, project issues and project feasibility. The design chapter explains the modelling of the system and is followed by an implementation chapter that explains the creation of the final system based on design decisions. The testing and validation of the system is described in the chapter after implementation. The penultimate chapter evaluates the final solution and the report concludes with a conclusion section that explains how the system can or could have been improved. This report also includes a glossary in Appendix B to explain the medical and technical terms used in this report.

2 Background

This chapter provides a general background to prenatal screening; it explains in general terms the theory behind prenatal screening, how the screening process works and identifies the current issues related to screening. It concludes with an explanation of the risk calculation process.

2.1 Screening Theory

Screening is a systematic process of identifying the possibility of an individual being affected by a disorder or a disease when no problem is suspected. It is a means of finding out those individuals who are most at risk of having a certain condition and give them further, more expensive, inconvenient and even dangerous diagnostic tests (Wald et al).

There are a multitude of diseases for which prenatal screening programmes have been developed. Several conditions can be tested for in a fetus or embryo before it is born (Jones, 2005). The aim is to detect chromosomal disorders such as Trisomy 21 (caused by incorrect chromosome distribution), metabolic disorders (structural malformation of a fetus) and anatomical malformations. In addition, during prenatal screening, several other conditions (e.g. syphilis) or infections that may adversely affect the pregnancy may be discovered or tested for.

2.2 Congenital Disorders

Congenital Disorders are medical conditions that are present at birth that have been caused by genetic or environmental factors. Chromosomal disorders are caused by an irregular expression of genes in a person resulting in a clinical phenotype (Oncology, 1999). Possible causes are gene mutation and abnormal chromosome distribution, as in Down's syndrome (extra chromosome 21).

Some of the common congenital disorders are:

- **Down syndrome**, a variable combination of congenital malformations caused by Trisomy 21. It is the most commonly recognized genetic cause of mental retardation. Because of the morbidity associated with Down syndrome, screening and diagnostic testing for this condition are always offered as part of prenatal care (Hernandez et al 1996).
- **Edward's syndrome**, a trisomy of chromosome 18 whose effects include mental and serious health problems.
- **Patau Syndrome/Trisomy-13**, a trisomy of chromosome 13, its symptoms and effects are similar to those of Edward's syndrome.
- **Klinefelter's syndrome**, an abnormal gene distribution affecting men that results in an extra sex chromosome. Men with Klinefelter's syndrome are usually sterile and maybe dyslexic (Medline Plus, 2006).

- **Spina bifida**, malformation of the spine resulting in a spinal opening. Effects include; paralysis, nerve damage and bowel-bladder complications (GM Stocks, 2005).
- **Ancephaly**, a condition that results in incomplete brain formation, its effects include missing brain hemispheres and life expectancy for affected infants is less than a few days.

2.3 Screening and Diagnostic Methods

Screening and Diagnostic methods are the techniques used to detect fetal abnormalities. These methods can be subdivided into invasive and non-invasive procedures. Invasive methods give more conclusive results and involve elaborate, risky and penetrative procedures for diagnosis. Non-Invasive methods estimate the likelihood of the abnormality and are more prone to error.

2.3.1 Invasive Techniques

Invasive procedures involve inserting a needle guided by ultrasound into the uterus, abdomen or other internal parts of the body and withdrawing a small amount of fluid for cell analysis.

- **Amniocentesis**: A procedure in which a needle guided by ultrasound is inserted into the uterus and withdraws some amniotic fluid for cytogenetic cell analysis and diagnosis.
- **Chorionic Villus Sampling (CVS)**: A small amount of villi (hair-like fringes of the placenta) is removed using a needle inserted into the abdomen (Cunniff C, 2004).
- **Embroscopy**: Involves putting a probe with a camera into the uterus to observe the fetus.

2.3.2 Non-Invasive Techniques

Non-invasive methods involve taking fluids (blood or urine) from the mother for analysis or using external diagnosis mechanisms.

- **Ultrasound**: The most common method of prenatal screening is ultrasound testing. It uses ultrasound waves (high frequency sound waves) to produce an image or sonogram of a fetus. In addition some other measurements can be carried out for further analysis; ultrasound screening is explained in more detail in section 2.5.3.
- **Maternal Serum**: Maternal serum screening involves measuring biochemical markers present in the mother's blood during pregnancy (Haddow et al, 1999). Abnormal levels of these chemicals may indicate increased risk for certain birth defects and genetic diseases.
- **Physical examination** of a pregnant woman's stomach may reveal the presence of an abnormality to an experienced physician.

2.4 Reasons for Prenatal Screening

Knowing that a child will be affected by a disorder enables parents to plan for birth and receive counselling. Clinicians may also prepare for the birth and have necessary treatment ready (Medline Plus, 2006).

- Diagnostic tests such as amniocentesis are expensive and risky; screening gives a means of deciding which pregnancies warrant further investigation
- Prenatal screening eradicates illness pre-emptively and in doing so it has altered fundamental society attitudes towards parenting and people's notions of medicine.
- Screening is very useful in identifying and verifying conditions deduced from symptoms or physical examination and can even detect diseases in the pregnant mother
- When a chronic condition can be avoided, early diagnosis not only saves the health system money, but gives the individual an improved quality of life
- Post-screening treatment may reduce the risk of developing a given condition or its complications
- Prenatal diagnosis of fetal anomalies gives parents the choice of continuing or terminating an affected pregnancy. Moreover, it is crucial for them to be prepared for a newborn disabled child.

2.4.1 Issues in Prenatal Screening

Although prenatal screening is considered routine in our century, a number of issues have arisen from the differences in opinion of various stakeholders and the controversy created by the somewhat negative effects of screening on society.

- Invasive methods do pose known dangers; there is a miscarriage risk of 1-2 percent following CVS (Jones, 2005)
- Some experimental forms of genetic screening are socially controversial, examples include vitro fertilisation, a procedure in which embryo's are genetically analysed and those which have signs of genetic defect are discarded
- People could abuse the system, leading to wrongful abortions

2.5 Prenatal Risk Calculation

The risk values calculated during prenatal screening are used to determine whether further investigation is warranted. The likelihood of a disorder is normally calculated using two main factors, the risk of the disorder due to age (**maternal age risk**) and the risk due to test results (**posterior risk**) from maternal serum markers, ultrasound tests and other risk factors (Cuckle H, 1997). The final likelihood of having a given disorder is then calculated from the age and posterior risk.

2.5.1 Maternal Age Risk

The incidence rate of fetal anomalies is directly related to maternal age. The risk of having an affected child increases exponentially after the age of thirty as shown in Figure 1. Historically, maternal age has been used as the first screening test for fetal genetic disorders. Maternal age risk is calculated using a population age risk equation and maternal age (Thompson SG, 1997). The age risk is based on the average disorder incident rate in the population for the mothers' maternal age. Figure1 shows the relationship between maternal age and the risk of Trisomy 21.

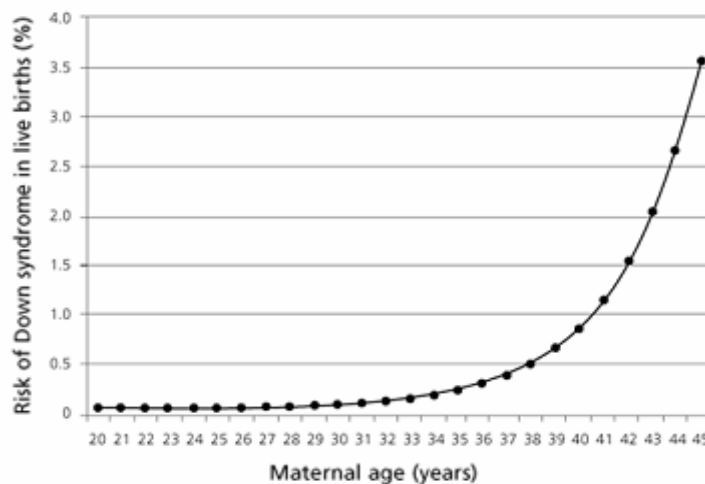


Figure 1 - Age Risk in Relation to Maternal Age (Hook EB, 2000)

Furthermore before serum results can be evaluated, the gestational age (GA) of the fetus has to be estimated as accurately as possible because correct analysis of the different biochemical marker results depends on knowing the gestational age precisely. The three basic methods used for calculating the gestation of a fetus are: menstrual history, clinical examination and ultrasonography, with the latter being the preferred method due to its accuracy.

When GA is calculated from menstrual history, the gestation is based on the first day of the last menstrual period (LMP), such that the GA is equal to the day of testing minus the LMP date. GA estimation by clinical examination of the pelvis or abdominal palpation is useful up to 28 to 30 weeks gestation, after which it becomes too inaccurate for GA calculation (Wilcox, 1993).

Ultrasonography is by far the more accurate technique for estimating GA. Ultrasound methods for calculating GA include crown rump length (CRL), biparietal diameter (BPD) and head circumference (HC). CRL is the longest verifiable length of the fetus excluding limbs while BPD is the diameter of the fetal head. An appropriate formula is then used to calculate the gestation from the obtained CRL, BPD or HC measurement (Westerway, 2000).

Ultrasound Screening is mainly used to determine the gestational age of a fetus; however other serious medical problems such as blocked intestines or organ defects may be detected during ultrasonography. The most established ultrasound marker is nuchal translucency (NT), normally sampled between weeks 11 - 13 of gestation. 30% of fetuses with increased nuchal translucency (found in the nuchal area at the back of the neck) were found to have chromosomal disorders or other fetal anomalies (Smith-Bindman Study, 2001).

2.5.2 Posterior Risk

Posterior risk is calculated from the results of clinical tests carried out on the mother's blood to measure biochemical markers, ultrasound markers results and other pregnancy related factors. The most common biochemical markers used in maternal serum screening are:

- Alpha-fetoprotein (AFP) - a protein made in the liver of the fetus. In neural tube defects, the fetus' skin is not closed so large amounts of AFP leak into the mother's blood. Low levels of AFP in the second trimester (14 - 26 weeks gestation) may indicate a T21 positive pregnancy
- Human chorionic gonadotrophin (HCG) - A hormone produced by the placenta, it indicates pregnancy and a part of it; beta sub-unit is increased in affected pregnancies. High levels of HCG in the second trimester may indicate T21
- Unconjugated oestriol (UE3) - A female sex hormone produced in the placenta, it is decreased in Down syndrome affected pregnancies.
- Pregnancy associated plasma protein A (PAPP-A) - low levels of PAPP-A in the first trimester may indicate T21 and other disorders.
- Dimeric Inhibin-A - an active form of the glycoprotein Inhibin-A that is produced in the placenta, T21 affected subjects may have high levels of Dimeric Inhibin-A in the second trimester

Once values of some markers have been obtained, a **Posterior Risk** value is calculated based on the difference of the marker values to the normal marker values (population marker median) of the testing laboratory. Correct analysis of the different biochemical marker results depends on knowing the gestational age precisely and proper adjustment of the marker values for Covariables. Covariables are pregnancy related factors of the mother such as weight, smoking and diabetes. For example weight gain indicates good fetal health therefore marker values have to be adjusted for these factors to enable correct regression analysis.

Posterior risk is also affected by other risk factors. Examples of such risk factors include previous T21 or T18 positive pregnancies. These factors may increase the likelihood of a pregnancy being affected by a disorder. For example a previous positive Downs or Edwards syndrome pregnancy may increase the risk of a new pregnancy being positive due to the increased risk of recurrence if a previous fetus was affected.

2.5.3 Final Risk

The final risk value is calculated from the posterior and age risks, a constant risk cut off value is then used to determine whether a pregnancy is positive or negative and if positive, further definitive tests are then carried out. Definitive prenatal diagnosis is then carried out using invasive techniques such as those discussed in section 3.3.2.

2.6 Project Methodology

In order to solve the problem of this project, the developer considered various methodologies for software development. These methodologies are generic models which give useful abstractions that explain different approaches to software development. The methodologies considered for providing a guiding framework for this project are described below:

- **Waterfall model** – The waterfall model was derived from engineering processes (Royce, 1987). It separates fundamental software processes into separate stages, requirements specification, design, implementation, testing and maintenance. This model enables the development of well structured systems based on initial requirements and easier project planning. It suffers from inflexibility, making it difficult to respond to changes in requirements under this model (Somerville, 2004).
- **Prototyping** – An evolutionary development model that interleaves the specification, development and verification of systems. An initial system is rapidly developed and several refinement cycles are carried out based on user feedback to produce the final system. It enables rapid software development but has drawbacks in that software documentation is difficult and prototyped systems are often poorly structured due to continual change (Herndon, 1997).
- **Spiral Model** - The spiral model splits the development process into a sequence of activities, “Setting objectives”, “risk assessment”, “development and validation”, and “planning”. At each stage or loop, the above activities are carried out in order, leading to incremental development. It enables risk minimisation because of its emphasis on risk management; however it is more suited for large projects (Schaum, 2001).

Some development methodologies are better suited for some parts of the system than others for example the Waterfall model is not suitable for user interface development while the evolutionary approach would be well suited for such a task. The Spiral Model was considered unsuitable for this project because project scheduling is difficult at the start since some tasks are not analysed until they are going through design. Eventually it was decided that a mixed process was the most suitable model, in order to incorporate the best features of the Waterfall and evolutionary methodologies. It would involve using an evolutionary approach to resolve requirements for a given component and then finish implementation using a more structured waterfall based approach. System components whose specifications were unambiguous would be developed using the Waterfall model while other parts were developed evolutionarily.

The mixed approach is justified in that it enables the developer to reap the benefits of both methods. It enables development to start before requirements are fully established, and as users develop a better understanding of the problem this can be reflected in the software. This methodology makes it easier to maintain software and when errors or new required functionality is identified, the system can easily evolve to remain useful. This project will chiefly follow the Waterfall model, such that the chapters of this report will mirror the stages of the Waterfall model.

3 Analysis

This section will analyse the general issues concerning this project's development and will evaluate its feasibility and associated risks. It aims to model the system in real world terms taking into account the project's constraints. It analyses the project constraints and development issues; and describes the results of the feasibility study in detail. Furthermore, alternative systems will be evaluated with the aim of identifying ways of improving this application and to ensure that the assessments of those systems are taken into consideration before embarking on design. This analysis is performed to ensure the project is feasible and to pinpoint the requirements, constraints and scheduling needs of the project and maximise its likelihood of success. Requirements analysis cannot start until a statement of the objectives, scope and feasibility of the project have been established.

3.1 Considerations

Successful projects are those that increase business value and satisfy or exceed user needs. They are delivered on time and on budget. Before commencing analysis, the developer took some time to reflect on the issues and problems that have affected other projects both negatively and positively. Considering such issues is important because as Stepanek (2005) explains, doing so can help the developer learn from others mistakes and good practices. There is no overriding factor for project failure; however, there are some fundamental problems that regularly face projects that fail.

Lack of planning may lead to project failure because developers may fail to foresee likely problems and may lead to things being done in an unorganised manner. Long schedules or a lack of them may be life threatening to the project because it leads to systems being delivered late or when their functionality is out of date. Poor requirements specification might mean that users will not be satisfied by the application and if it delivers the wrong functionality then the project might fail. Another common cause of project failure is scope creep, a situation whereby the project grows insidiously during the project lifecycle. User involvement is crucial to this project; lack of it may be fatal to the project and result in the system being unacceptable or fails to meet expectations. User involvement is essential for users to feel as contributors to the final solution and might make it easier for them to accept the new system (Stellman and Greene, 2005).

In view of these problems, the developer sought to identify ways of maximising the success of this project. There is a clear need for planning; it enables problems to be put into perspective and for the developer to choose the right solution. Clearly defined and measurable results are very important for scheduling, user acceptance testing purposes and to the success of the project (Stepanek, 2005). To that end, the developer compiled a user requirements specification that was reviewed by the user before design was started. During the early stages of the project, the developer negotiated with the MI project supervisor to establish the main requirements of the system, such that the project's scope

could be established as early as possible and for drawing up the project schedule. The project schedule was a useful tool for project management and progress monitoring.

3.2 Constraints

Constraints are items that by their nature restrict the choice of solutions and constrain the scope of a project. Various limits and constraint exist in a project development lifecycle due to financial, environmental, legal and other factors (Schwartz, 1997). Consequently it is crucial for such constraints to be evaluated before the development of an application commences to ensure informed decisions are made.

This project was developed for an external company, who along with other stakeholders were likely to impose various requirements and constraints on the project. These requirements had to be fulfilled in order for this application to satisfy their needs. The product would have to interface with existing software, a dynamic link library that encapsulates the clinical rules for risk likelihood calculation. The final system is scheduled to be finished by the end of March 2006, a month before the deadline for project submission. The system must use development technologies that are accessible to MI since they will be the main user of this system and will host it on their web server. The product was also required to be written according to the Software development guideline of MI, the guideline can be found in Appendix E – Analysis.

3.3 Stakeholders

This project was proposed by Media Innovations (MI), a company that specialises in developing screening, diagnostic and behavioural health software. They were actively involved throughout the lifecycle of this project. The main project stakeholders were MI and the end users. The developer had contact with end users such as consultants and some clinicians who work for the Leeds Screening Centre. Their active involvement was crucial in ensuring the application met user needs. They were an invaluable source of advice and information. In particular, MI assigned the developer Mrs C.J. Wilson as the project supervisor to represent the company's interests in this project. She acted as the project contractee and was actively involved in overseeing the development of the final system.

The majority of end users are people involved in screening, clinicians, consultants and screening centres. It is possible that researchers may use it to test certain assumptions and compare the system to other existing systems. Prospective clients of MI might also explore this system to find out more about the company and its products.

3.4 Feasibility Analysis

The feasibility study was integral to this project, it was important that the system was shown to be theoretically practical before commencing on design. The feasibility study analysed how the system would operate under different assumptions or constraints and to infer whether the system was viable. In order to evaluate the project's potential for success; the feasibility study focused on five main sections: Technological, Legal and Political, Ethical, Economical, Operational and Scheduling.

3.4.1 Technological

This section analyses whether it is possible for the system to be successfully implemented given the current technology. The system will be hosted on a web server running either SQL Server or Microsoft Access. Both database management systems are satisfactory for the data storage requirements of the application. The final solution will be developed on a platform that has language facilities for developing web applications. Of the current technologies that are available, most are either free or already installed on MI machines. The MI web-server is connected to the internet via a gigabyte internet connection. Therefore it will be able to provide the bandwidth required to satisfy clients using the application. The web server has about 150 gigabytes of disk space, enough disk space to store the settings data required by the application. Therefore the developer concluded that it was possible to develop the final solution using the current technology.

3.4.2 Legal and Political

Medical software faces legal and political challenges that impose constraints on what such software can do. Compliance with legal requirements is crucial for the system to be feasible since non-compliance is not an option. The system must abide by the Data Protection Act of 1994 which states that systems must protect the privacy of data and ensure that it is up-to-date, accurate and not misused. The system will encrypt highly sensitive data and authentication details. Users will be required to login before they can access the system. In addition the web server is protected by a firewall system that prevents unauthorised access to settings data used by the application. Besides the normal data management laws, medical software is bound by laws that regulate the functions that such a system can perform. The system must be clinically validated before it can be used; to that end MI will use external evaluators to test the system. Under medical law, the system must only be accessible to registered physicians. The final solution will only allow registered medical users who have applied to MI for login details. Beside the normal health regulations that govern medical institutions and practices, so far there is no legal provision for telemedicine. Nevertheless the system will be developed with due consideration to medical law and other applicable legal constraints.

3.4.3 Ethical

Prenatal screening has controversial benefits and the social issues arising from it are even touchier. Eradicating illness through screening requires a change of fundamental attitudes towards parenting and social responsibility; this has given rise to a multitude of ethical issues related to screening. Prenatal diagnosis with a view to termination of the pregnancy is widely acceptable (Mahoney, 1990). Most people fear that prenatal screening may lead to people choosing what children they want to have. Is a mentally or physically different child less valuable to society? Although parents are fully informed prior to every stage of the screening process, making decisions regarding life is very difficult and will always be controversial, but as Stephen Thacker of the centre for disease control says, "Screening examination in early pregnancy is here to stay" (Mahoney, 1990). Prenatal screening

has undoubtedly widespread advantages as well as negative social consequences; however the medical community generally agrees that screening has considerable benefits (Jones, 2005).

3.4.4 Economical

Policy makers and health services are under increasing pressure to improve health provision and cut down soaring health costs (Scott, 2005). Widespread screening and termination of positive pregnancies would reduce both problems, although it is arguable whether cost benefit analysis is acceptable when it comes to health. For example Down's syndrome accounts for 15 percent of the institutionalised mentally retarded population (UKGTN, 2005) and is highly costly to both patients and society. The benefits in the economical sense far outweigh the costs.

From Media Innovations' point of view, a comparison of the development costs and the business value the project brings to the company and their clients shows that the development costs of the final solution are minimal. The requisite development software is already available on their system and their server has all the required capabilities for hosting the final solution. After installation, running costs will only amount to the cost of the disk space and bandwidth that is used by the application. In contrast, the company can only benefit from the revenue that might be generated from this system. It is therefore estimated that the cost of producing the final solution requires a small budget and developing the system is economically feasible.

3.4.5 Operational

The operational feasibility of the application is concerned with the viability of the day to day running of the final application. The application uses one main resource that accrues costs on an operational basis. The system will require an internet connection and this costs the company an annual fee of £250. However regardless of the existence of this application, the company would still require internet connectivity and the introduction of this application does not increase operational costs. Therefore at best the operational costs are minimal. The system does not require any attendance from a human being except for occasional maintenance tasks.

3.4.6 Scheduling

This section analyses the viability of the project being completed on time. Development of the final solution is predicted to end on the 12 of March 2006 as shown in the project schedule - Appendix C. The implementation of the solution and documentation are scheduled to be completed by the 27 of March 2006. Enough time has been allocated to cater for delays and testing. Although this project is complex and requires a considerable amount of time to complete, it should be possible to complete it on time if things go according to plan.

3.5 Risk analysis

Theoretically every decision in a project should be subjected to a risk assessment in order to manage risks effectively (Crump, 2005); however such an approach is impractical for all but major project risks. Effective risk management involves both informal awareness and a structured approach in

which potential risks are identified and mitigation strategies are systematically implemented. Complex software development is risky and understanding risks is a crucial step in avoiding becoming another failed software project. Many post-mortems of project disasters indicate that problems would have been avoided or reduced if there had been explicit concern in identifying and resolving high risk elements (HRS Consulting, 2000). This section will identify, analyse and prioritise risks. It will describe mitigation strategies and allocate project resources for managing risks.

In order to manage risks effectively, the process of identifying risks and developing strategies for avoiding, transferring or reducing the negative effect of the risk has to be documented clearly (Boehm, 1991). The developer compiled a risk register as part of the project risk management. The risk register is a log that lists all the identified risks, their analysis and mitigation strategies. Contingency plans were also drawn up for every risk to provide alternative solutions should the mitigation strategies fail. This document was used to track and monitor risks throughout the life cycle of the project. The risk register can be found in Appendix C – Analysis.

Provided the strategies specified in the risk register are followed, with a little fortune the likelihood of the risks identified in the risk register happening is minimal. The risk register showed that the risks most likely to affect the project would not be fatal for the project if mitigation strategies were implemented early enough. Throughout the lifecycle of the project, the developer consciously tried to make informed decisions to prevent risks and involved stakeholders with the intention of stipulating preventive actions.

3.6 Existing systems

Existing systems that are used in prenatal screening were analysed to aid the design of the system. The analysis evaluated a Screening Calculator previously developed by MI and the Benetech PRA system. The MI Screening calculator is still in use, however this system is Windows based and cannot be used over networks. It has notable batch calculation functions and has been clinically validated however it is not widely used by physicians. It has disadvantages due to a lack of configurability and is quite complex to use because it lacks a results interpretation mechanism.

Benetech PRA (2006) is a widely distributed system in North America that uses the latest database technology and has a good user interface. It does not include the latest markers and heavily relies on one marker - alpha-fetoprotein. This makes its usefulness rather limited since results are more accurate depending on the breadth of input information. These systems are not network/internet enabled and do not yet include the latest ultrasound markers. Due to the drawbacks of the above systems, MI felt that a new robust, configurable and internet enabled system was needed to deliver the latest developments in prenatal screening (Wilson, 2006).

4 System Requirements

This section defines the high level functions and services that the system was expected to provide. User requirements analysis produces descriptions of content, functionality and quality required of the final solution. The software specification defined here served as the basis for more detailed design and implementation. Software system requirements are often classified as functional or non-functional requirements.

4.1 Elicitation Techniques

There is no perfect or universally applicable requirement gathering technique, several approaches were considered for this project. Four main techniques were considered with the aim of choosing the most suitable ones, they were ethnography, scenarios, view-point based elicitation and interviews.

Ethnography is an observational technique in which an analyst observes the work environment where the system will be used (Agar, 1996). From this social or organisational context, the analyst can discover implicit requirements which reflect actual work processes. It can be particularly useful at discovering process details that are often missed by other techniques. However due to its focus on the end user it wont discover other, more specific or new requirements. As a consequence of its incompleteness ethnography has to be used in conjunction with other elicitation techniques.

Scenarios are a requirements gathering techniques in which users describe how they might interact with a software system in a given real life scenario. The scenario descriptions can then be used to formulate system requirements. This technique is particularly useful for adding detail or clarifying outline requirements. However it requires the developer to have considerable understanding of how the final system might work if they are to present the user with enough scenarios to comprehensively capture requirements.

VORD (Viewpoint-oriented requirements definition) is a service oriented method for requirements elicitation. It is concerned with identifying different stakeholders or system viewpoints such as users, data sources or any other receivers of services. Identifying the requirements of these viewpoints enables system requirements to be easily gathered. VORD is particularly valuable in discovering requirements conflicts (Easterbrook and Nuseibeh, 1996).

A key strength of VORD is that it identifies the diversity of requirements sources and therefore imparts a sense of completeness. However it is less appropriate for structuring requirements as there is no connection between some internal system viewpoints and system end users.

Interviews of key stakeholders are crucial to the requirements gathering process. Structured interviews involve asking prepared question to stakeholders of the system to find out what they require in the new system (Pressman R.S, 1992). Interviews are a straightforward means of gathering requirements. The specifications produced are more acceptable to users since they are based on their

opinions. However considerable time is required to prepare for the interview and study background information extensively (Kotonya, 1998).

After considering the advantages and drawbacks of the requirements elicitation techniques discussed above, the developer decided to use stakeholder interviews and VORD for gathering requirements. The system would be broken up into viewpoints as shown in Figure 2 and interviews would be used to gather requirements from the stakeholders for all real viewpoints.

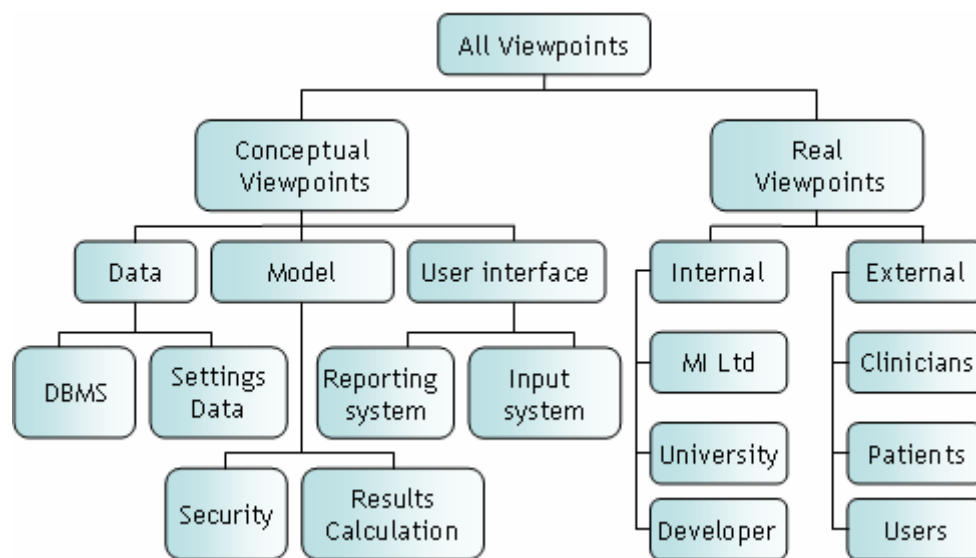


Figure 2 - System Viewpoint Hierarchy

The requirements of conceptual viewpoints would be based on the services that the conceptual viewpoints were required to provide in order to satisfy the real viewpoints (Figure 2). For example the user viewpoint requires a list of markers to be displayed on the user interface; the data source viewpoint can meet that requirement by storing data in a database and making it available to the system. So in turn the data viewpoint would have a requirement for a table that stores marker information.

During the project inception stage, interviews were carried out to establish the needs of users and the scope of the project. Interviews were carried out with MI representatives and prenatal screening consultants. The interviews were carried out to find out what the users wanted from the final solution. Prior to every interview, an interview plan would be created; the plan contained a list of interview questions and a description of the goals of the interview. The user also had a copy sent to them prior to the interview. Appendix F – User Requirements shows a sample interview transcript.

4.2 Requirements Gathering

In addition, to interviews, the MI project supervisor would review the application on a regular basis, give suggestions and clarify requirements. Figure 3 shows that system requirements gathering was an iterative process with continual feedback from each task.

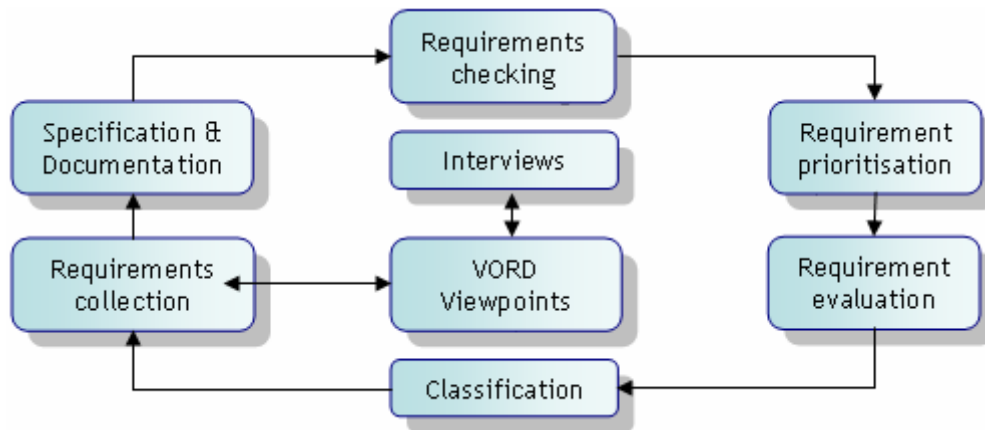


Figure 3 - Requirements Specification Process

4.3 Functional and Data Requirements

Functional requirements describe the services that the system is expected to provide. They describe all external and programming interfaces of the system and its functionality requirements such as results calculation, data manipulation and processing. Appendix F - User Requirements gives a full listing of the system’s functional requirements.

4.4 Non-functional Requirements

Non-functional requirements define specific system quality attributes, properties and behaviour. They are just as important as functional requirements since they determine how usable a system is to users. The paragraphs below describe the non-functional requirements of the final solution.

Appearance - The system shall comply with the existing colour scheme used on existing MI applications. It should use similar images and banners to those already in use on the websites and products of MI. Labelling must be concise, grammatically correct and consistent, appropriate terms must be used for labels.

Usability – The usability of the application measures the efficiency and satisfaction with which users can achieve certain goals using the program interface [ISO 13407, 1996]. The product must be enjoyable and uncomplicated to use. The usability requirements of this system are based on the Massachusetts Institute of Technology (MIT) information services and technology usability guideline. The MIT accessibility guideline can be found in Appendix F.

Internationalization – The application must be designed to work in diverse locales, it must show dates in the user’s locale format. The date entry control must display months in the user’s language. Likewise numbers must be interpreted and validated with regard to the user’s locale. Moreover, the system must be usable from most widely used browsers.

Accessibility - The product must be accessible to users with common disabilities. The system shall use colour schemes that enable partially-sighted and colour blind users to use the system. The system must be designed to comply with the W3C accessibility guidelines reproduced in Appendix F.

Performance Requirements - The system must provide a reasonable response time of less than nine seconds over an average internet connection. Input validation must be performed on the client to ensure post-backs are cut down to a minimum and on the server to cater for non-JavaScript enabled browsers. In addition static data must be cached to improve performance. If the batch calculation feature is implemented, the system must provide a timely (less than 24hrs) response to batch transactions.

Operational Requirements - The system cannot store patient data on the server due to the stringent data confidentiality and protection rules that have to be followed. Non-registered users must not be able to access the system. It must validate user input in order to ensure that valid data is used in calculations, although the user takes responsibility for the values that are used in calculations.

Maintainability and Support Requirements - A document must be written to explain how the system was implemented. The system must be written in a proper object oriented approach, and reusable functions and modules must be created whenever possible.

Security Requirements - The application must have some security features that will prevent unauthorised access and prevent access to internal resources. Login details must be protected during transmission over the internet. The system shall prevent access to users when they have logged out and shall not store confidential data on the users' machine, in accordance to confidentiality guidelines.

Project documentation - The application's implementation documents must be written according to Media Innovations document writing guidelines. In particular, documents must have version, addition control systems and must be written to a very high standard of grammatical correctness and consistency. Software documentation must be written during the development process to ensure details are not overlooked or forgotten. A list of tasks to be done must be maintained for every document in order to ensure necessary additions are always done. Product documentation can be found in Appendix G - Product Content.

Legal Requirements - The system shall fulfil the validation requirements specified for medical software. It must comply with the Data Protection Act of 1984, medical and civil software regulations.

Reliability and Availability - The system shall be available for use 24 hours a day and operate independently without need for regular human intervention.

4.5 External Interface Requirements

The final solution will provide a web form based user interface. These web forms will have controls that enable users to enter clinical data, calculate and view results. This section describes the main components that the user interface will have. The system is required to provide a graphical user interface that will allow users to:

- Login securely

- Input and calculate single patient results
- Import and calculate batch/multiple patient results
- View and save results, preview and print reports and results
- Access and view help

In addition, user controls must be implemented for any reusable graphical component to enable efficient date entry and page navigation. The user interface must be adaptive to user locales and display dates and numbers in correct formats for the user's locale.

Software Interfaces - The application will interface with an existing software component that encapsulates the rules for calculating marker results. The system is required to communicate with the external software components by passing data in the correct format and data type.

4.6 Requirements Evaluation

The International Standards Organisation (ISO 9126, 1991) specifies 6 quality evaluation criteria for software requirements. In theory, good requirements are feasible, clear, verifiable, complete, consistent and testable. Essentially, requirements must be testable, this is important for validation purposes. However some requirements are not testable by nature and for those requirements, other evaluation criteria were used to evaluate those requirement is the Testing chapter.

A test plan was drawn up based on the functional requirements, the tests cases in the test plan will be used to check the functionality of the final solution. Additionally, the test plan will serve to prove the testability and verifiability of the system requirements. The test plan can be found in Appendix E- Testing. If a requirement cannot be tested or is infeasible it will either be eliminated or scaled down to a more attainable, specific and testable requirement. The requirements specification in Appendix F assigns unique identifiers to every requirement to allow the design, code and test cases to be traced back to the requirement (IEEE, 1998).

The use of the VORD technique to divide the application into separate view points was done with a view of ensuring that the elicited requirements were consistent and compatible (not conflicting with each other). In the case where independent viewpoints had identical or conflicting requirements, such issues were discovered and rectified using a simple comparison of each viewpoint's requirements to those of other viewpoints. It is assumed that by following the guidelines explained here and evaluating the system requirements as specified, the risk of implementing a system that does not satisfy requirements will be mitigated.

5 Methodology and Development Technologies

This chapter assesses web system development methods and technologies which were considered for developing the final solution and concludes with the final choices made regarding which method and technology was used. Methodology in this context refers to generic software development patterns that describe fundamental software process activities, their timings and the results of each activity. Development technologies are the platforms that facilitate the development of software applications.

5.1 Development Technologies

There are a variety of mechanisms by which a server can generate web content dynamically in order to satisfy requests from remote browsers/clients sent via the Common Gateway Interface (CGI). CGI is a standard that enables a web server to execute applications, generate and send content to browsers for display. The development technology used to implement a system normally determines the web server on which it is capable of efficiently running on for example Microsoft based systems are more suited for running on a Microsoft web server. Five potential technologies for implementing the final solution were considered with the aim of selecting the most suitable, they were: JavaServer pages, ASP.Net, Perl, Python and PHP.

5.1.1 Java Server Pages

JavaServer pages (JSP) technology was developed by Sun Microsystems and enables the development of interactive pages for web applications. It is part of the Java architecture and has access to all its components such as JavaBeans and servlets. JSP was designed to be both platform and server independent and adheres to the write once and run anywhere philosophy of the Java architecture (Sun, 2005).

JSP technology is platform independent and can run on any web server without having to be design for portability. The use of Java for scripting provides many benefits over basic scripting languages such as superior performance and scalability. Because JSP is developed by the java community, an open source software development process, it has undoubtedly gained from the robustness of that extended community (Sun, 2005). It has a highly structured architecture that enables the development of object oriented applications that are maintainable in a straightforward manner. JSP pages are highly efficient since the Java virtual machine stays up and a new lightweight thread is created to handle requests. It is generally free or very cheap, in contrast to most commercial quality web systems that are relatively expensive.

Debugging JSP pages is still complex due to an absence of a powerful integrated development environment. Database connectivity is not as efficient as it should be since connection pooling has to be done manually using custom code (Horton, 2003). To date, there is no what you see is what you get editor for JSP pages, consequently web page design and development has to be done using other HTML mark up technologies.

5.1.2 Active Server Pages .Net

Microsoft ASP.Net is a programming framework for building web applications. It is part of the .Net framework and succeeds Microsoft's Active Server Pages (ASP). The .Net framework is a development and execution environment that allows different programming languages and libraries to interoperate seamlessly in a language neutral common runtime language (MSDN, 2006).

The use of a common language runtime enables ASP.Net to provide a multi-language platform such that web pages can be written in VB.Net, C#, J#. It provides true object oriented programming with capabilities to handle polymorphism, inheritance and encapsulation (Liang, 2003). The caching mechanism provided in the framework greatly improves performance. Business logic is totally separated from presentation through the "code behind" development model. It provides library classes and controls that enable easy access to databases and rapid application development. ASP.Net can maintain session variables enabling a web server to recover lost data when an exception happens and to provide state to the inherently stateless web.

ASP.Net is primarily restricted to Microsoft Windows platforms, because it uses platform specific objects. The use of scripting languages for client side code does not scale properly for large solutions. Memory usage is not very efficient due to the large amount of code it generates resulting in longer execution time. ASP.Net is expensive in comparison to other web technologies.

5.1.3 Perl

Perl is a remarkably powerful scripting language and is highly efficient in terms of string processing, memory management and data typing. It is a procedural programming language that derives mainly from C and has one of the biggest collections of third party modules (Wall, 2000).

Perl is powerful, very expressive and provides an infrastructure for handling common tasks efficiently. It is widely used and supported; database drivers are easily accessible. Perl can be implemented on many platforms consequently Perl code is highly portable. Many tasks such as memory management are handled automatically.

Perl is excessively complex and compact, once code is written, it is very difficult to understand, it seems Perl has more than its fair share of terse and complex language constructs. Perl does not enforce data security; access to private data is not prevented by convention.

5.1.4 Python

Python is a dynamic, interpreted object oriented programming language developed as an open source project and managed by the Python Software Foundation.

Python is easy to use due to its clean and simple language constructs. It is free and highly portable, like Java. It is an object oriented language that supports important modern language features such as garbage collection and exception handling (Python, 2006)

As a new language, Python is not widely used; scripts may not work on some machines and as a rapidly developing language, it is highly unstable

5.1.5 PHP

PHP is a scripting language that depends on pre-processing of HTML tags. When a server's PHP pre-processor encounters PHP language tags, it invokes the PHP engine to execute the code.

PHP is free since its open source software; there are no upgrades or licensing fees. PHP works with the APACHE web server which has a proven track record of reliability and good security (Hull, 2006). It is a cross-platform technology that can operate on Linux, Windows, Solaris and other UNIX platforms. PHP produces shorter code; as a result PHP pages use memory efficiently and execute quickly (Lerdorf, 2005).

There are problems in PHP's object model. It was not designed as an object-oriented language and as a result it lacks features such as exception handling and garbage collection. As with most scripting languages, PHP lacks a proper debugging tool and as result debugging can be very complicated.

5.1.6 Conclusion

Based on the performance, security, ease of use and advantages of the systems described in this section, it was concluded that ASP.Net and JSP would be ideal platforms for implementing the final solution. The developer has studied all the above technologies except for Perl and PHP. PHP was considered unsuitable for this project due to its lack of object oriented architecture. Perl was discounted due its considerable complexity and lack of security drawbacks. Python does not provide a graphical interface for designing the applications user interface, consequently the developer felt that it was unsuitable due to its lack of an integrated development environment (IDE). With the tight project schedule in mind, it seemed prudent to use a platform that the developer was very proficient in and a platform that provides a powerful IDE. Having to use an independent HTML editor to create web pages in JSP made it less attractive as a medium for developing the final solution.

Based on these requirements, ASP.Net was chosen for developing the final solution because its minor drawbacks are unlikely to negatively affect the final solution. In addition, the proliferation of Windows based systems in the health industry made ASP.Net a more attractive choice.

5.2 Programming Languages

The chosen development technology - ASP.Net provides three main programming languages, VB.Net, C# and J#. J# was not considered as a viable option due to its lack of widespread use and support. C# and VB.Net perform largely equivalently since they have access to the same framework libraries. However VB.Net is simpler and more productive due to better intellisense, optional parameters and background compilation. Consequently VB.Net was chosen for writing the final solution due to its relative simplicity and natural language, case insensitive approach. It also has key strengths in its support for optional parameters in method invocation and good exception handling.

5.3 Scripting languages

Client scripting is used to make web pages dynamic and interactive, for example it might be used to validate a field so that a user gets immediate feedback when they enter invalid data. The main scripting languages considered for client side scripting were Jscript, VBScript and JavaScript. The main features affecting the choice of scripting language are target platforms and language features.

VBScript is a subset of the Visual Basic language. It is flexible in that it can be used in a huge variety of applications and provides good error handling. However it does not work on Netscape browsers rendering it unsuitable for the purposes of this system.

Jscript is a Microsoft implementation of the ECMA script an international standard scripting specification. Jscript is a dynamic, object oriented language similar to Java. Its object framework is very similar to that of JavaScript.

JavaScript is a Netscape client side scripting language. It is the de facto industry standard for scripting. It is object based and certainly it is superior at transforming a lifeless page into a fully interactive page with real-time response. The availability of control flow like case statements gives JavaScript a considerable advantage. It also provides a comprehensive assortment of built-in functions for formatting, mathematics and many other mundane programming functions.

The advantages and versatility of JavaScript led to it being chosen as the client side scripting language of the final solution. JavaScript is highly documented and can run on most browsers; making it a practically viable choice.

6 Design

This chapter describes the conversion of the system requirements into design specifications that define the overall system architecture. It models the system components, modules and data required to satisfy requirements from an implementation perspective. Before system design commenced, a design pattern was chosen based on suitability and fitness for purpose to provide a guiding framework in which the final solution was designed.

6.1 Design Pattern

Design patterns describe classical models of solving recurring problems in software design. ASP.Net provides a classical model for implementing web applications called the Model View Controller (MVC) pattern. The main objective of this model is to decouple the system's presentation layer (View) from the data processing layer (Model) so that the same model layer could be used for various views; it also enables better organisation and code reuse (Design Patterns, 2004). The MVC pattern was selected as the overall framework in which further architectural design of system components was done using design models such as the object oriented model.

Model – The model layer handles data processing and storage, it also encapsulates most of the code functions and result calculation logic.

View - The View is the graphical user interface, it handles the presentation of results produced by the model and is separated from the complex data and calculation operations.

Controller – The controller handles events from user interactions and in turn calls some methods of the model layer which eventually update the view.

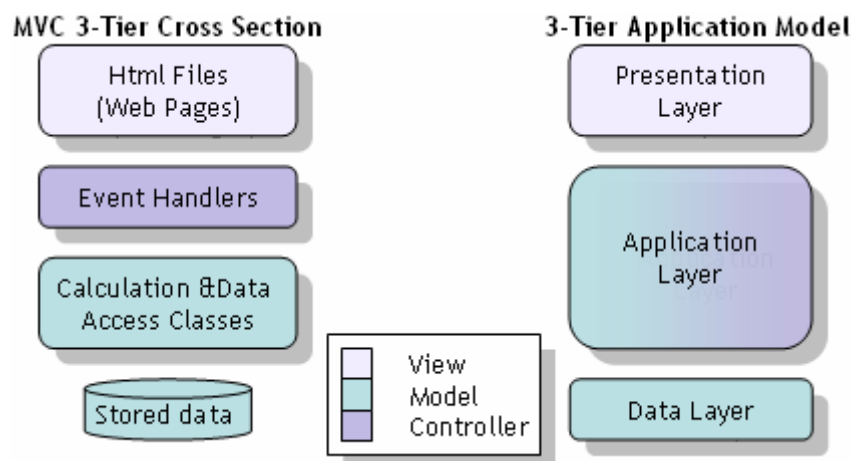


Figure 4 - Design Pattern Overview

The system was dissected into MVC layers as shown in figure 4. After this initial analysis, appropriate design models were chosen for designing the system components in the MVC layers. The Model layer components were to be designed using an object oriented model because the results calculation objects represent real world entities and are particularly well suited for such a model.

6.2 Architectural Design

Once design patterns had been established, the first stage of the design process was to decompose the system into a set of interacting subsystems. Figure 5 is a structural model of the architecture of the system based on Figure 4 that shows the system's components in their respective MVC layers.

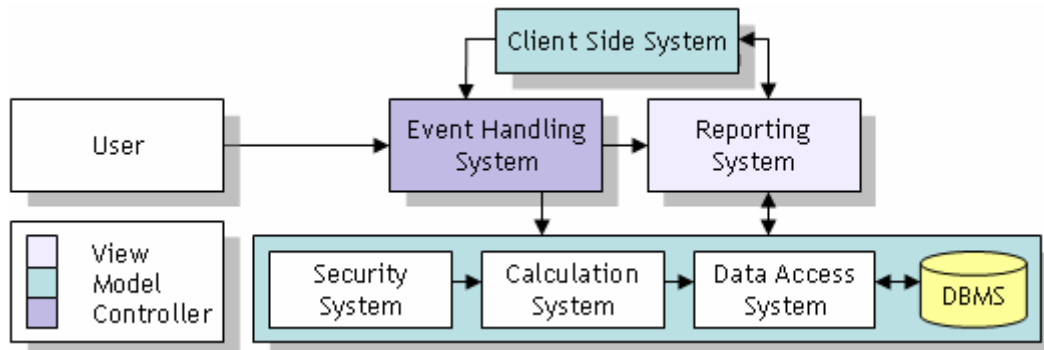


Figure 5 - System Structure

6.2.1 "Model" Layer Design

The design of the system components that belong to the MVC model layer was carried out using an object oriented approach. The objects in this layer carry out calculations and data access.

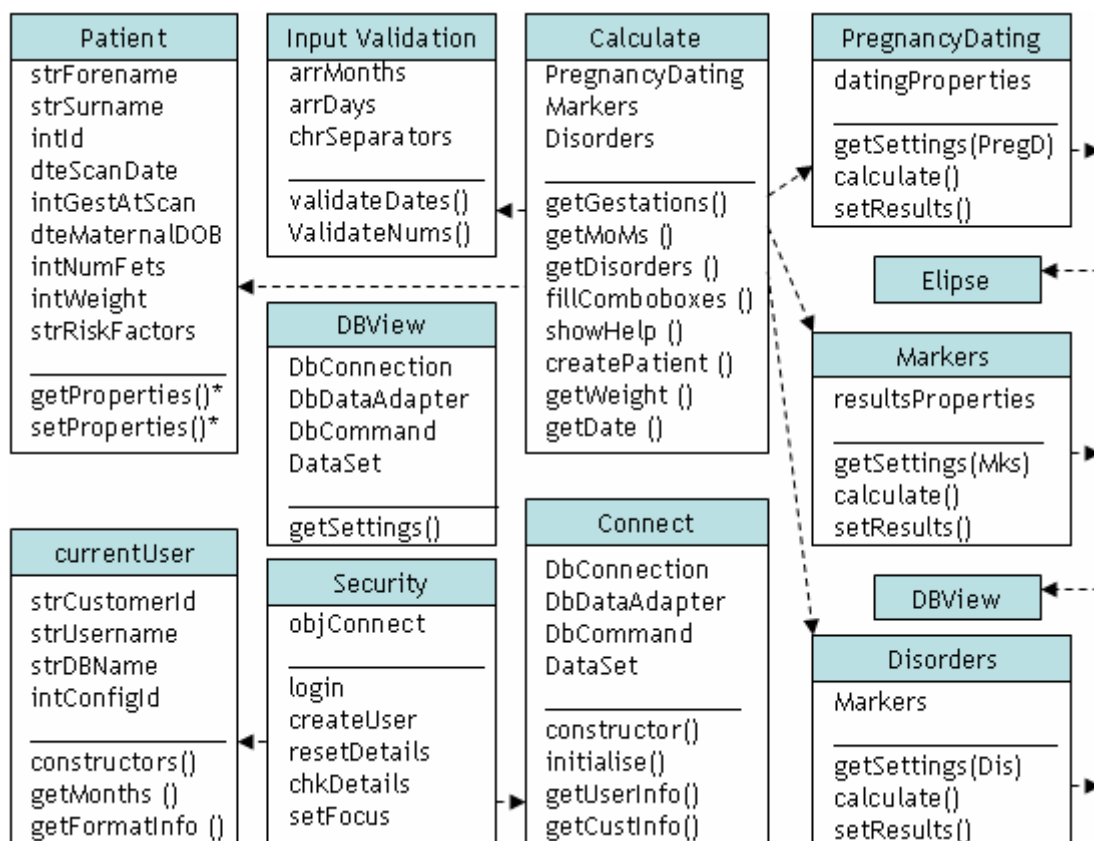


Figure 6 - System Classes

Figure 6 shows class diagrams of the objects and their relationships, each class will be implemented in a separate file. The following paragraphs explain each class in more detail.

6.2.2 Connect

This class connects to an SQL Server database using a native database connection object. It then uses a precompiled SQL command to retrieve user details and store them in a dataset. A precompiled command is used to prevent attackers from injecting malicious SQL commands into login fields. It is mainly used by the security class to retrieve user details from the database.

6.2.3 CurrentUser

The CurrentUser class stores the name, user identity and configuration details of the current user; this information is used to identify the correct settings records of the current user when calculating results and for reporting purposes. It has a get and set method for every property and the set methods validate new values before updating the current value of a property.

6.2.4 Security

The Security class provides security functions for the whole application; it encrypts user's login details in preparation for transmission over the internet using a secure hash algorithm. When users initially access the application, their login details are checked against stored values. If the login details are correct, a CurrentUser object is created and stored in a session variable.

6.2.5 Patient

The Patient class stores details of the current patient; patient information is displayed on patient reports for identification purposes.

6.2.6 PregnancyDating

This class calculates pregnancy dating results using an Elipse object, the gestational age of a fetus and mother's maternal age at expected date of delivery.

6.2.7 Markers

The Markers class calculates marker median values from the input biochemical marker test results using an Elipse object.

6.2.8 Disorders

This class calculates the final risk values for the selected disorders using the results from the pregnancy dating, markers and other additional risk information.

6.2.9 DBView

The DBView class connects to the database and retrieves settings data for use by the PregnancyDating, Markers and Disorders classes in calculating results.

6.2.10 Calculate

The Calculate class is the main class of the system; it has PregnancyDating, Markers and Disorders objects that calculate results from the input data in conjunction with the Elipse object. It uses the information provided by the user to create a patient object, handles results and report generation. It forms the code behind the calculation page which is described in section 6.4.

6.3 Elipse Object

The Elipse COM+ dynamic link library encapsulates the rules for calculating pregnancy dating, markers and disorders results. This component was developed by MI. The final system uses this component when calculating results. The Elipse object receives values and setting data from the system classes and calculates the required risk values. The Elipse object has already been validated and tested by MI.

6.4 User Interface (View) Design

The purpose of the user interface is to enable users to interact with the application, supply information for calculating results and initiate various application processes. Consequently, the user interface has to provide controls that enable users to input data, calculate and view results and perform various operations. Figure 7 shows the fields that each proposed page was required to provide for the underlying classes to work.

Security Interface Fields			
Customer Id			
Username			
Password			
Operations: Login, Reset			

Calculation Interface Fields			
Patient	Pregnancy Dating	Markers	Disorders
Forename Surname Id Maternal DOB Ethnic Weight	Num of Foetuses Monozygous Dating type Measurement Dating Date Donor Egg Egg Extraction date Donor Age at Extraction Age Type Donor DOB	Marker Name Assay Date Marker Value Site Id Days Gestation Maternal Weight Unit Value Is Normalised	Disorder Prior Risk Risk factors Diabetes Smoking IVF Previous Disorders Down syndrome Edward syndrome
Operations: Calculate, Help			

Patient Report	Results Report			
	Patient	Pregnancy Dating	Markers	Disorders
Shows all patient Information and the input data required to reproduce results And results as in the results report	Forename Surname Id	Expected Date of Delivery (EDD) Age at EDD (Based on patient details and dating parameters)	Marker MoM Gestation	Prior Risk Final Risk Result Interpretation
	Operations: View Patient Report, Print and Export			
Operations: Print, Export				

Figure 7 - User Interface Fields

The security, calculation and results components of the user interface (view layer) will be implemented as web pages. The fields contained in those components will be represented by text fields, check boxes and drop down boxes. Operations will be initiated by clicking appropriate buttons. The patient report will be created as a PDF or HTML file. The user interface will be implemented according to the usability and accessibility guidelines described in Appendix G – Design.

6.4.1 Colouring and Graphics

The system will use a colour scheme that makes it consistent with current Media Innovations software products. Colour is a very important attribute of the user interface and can improve usability and accessibility of the application. Text will have a white background and a black fore colour. This means the application will be usable on most systems and accessible to most users including colour blind people. Buttons will be accompanied with small icons that illustrate what the button does. The icons will improve the usability and appearance of the system.

6.5 Controller Design

In the MVC pattern, the controller is responsible for handling events raised from user interactions to initiate appropriate system processes. ASP.Net hides the complexity of handling events that happen on the server or client from the developer, such that code the required to implement the controller is relatively simple and straightforward.

To receive notification of an event, the application is required to declare an event handling method for that event. When the event occurs, the runtime environment invokes and executes the declared event handler. If the event occurred on the client side, ASP.Net will capture the event information, transmit the event message to the server and call the appropriate event handling method. As a result, all the application has to do is declare event handling code for the events that it wants to be notified of. This made it unnecessary to specifically design the system controller.

6.6 Security

Threats faced by the system can be categorised based on the goals and purposes of the attacks. Understanding threats helps in developing security and mitigation strategies for those threats. The STRIDE acronym (Microsoft, 2005) will be used to categorize threat types:

Spoofing - Spoofing refers to users gaining access to the system using a false identities gained by stealing user credentials.

Tampering - Tampering is the unauthorized alteration of stored data or data in transmission over a network between a client and the server.

Repudiation - Repudiation is the ability of users (legitimate or otherwise) to deny that they performed specific activities or transactions.

Information disclosure - Information disclosure is the unauthorised revelation of private data. For example, a user might view the contents of a table or file he or she is not authorized to open, or monitors data passed in plaintext over a network.

Denial of service - Denial of service is the process of making a system unavailable. This threat might be accomplished by bombarding a server with requests to consume available system resources or by passing it malformed data that could crash an application process.

Elevation of privilege - Elevation of privilege occurs when a user with limited privileges assumes the identity of a privileged user to gain privileged access to an application.

6.6.1 Countermeasures

For most threat categories described above, the security system will implement countermeasure techniques that will mitigate or reduce risk where possible.

Threat	Countermeasure to be implemented
Spoofing user identity	The system will use strong authentication and will not transmit login details in plaintext.
Tampering with data	Data hashing and signing are possible solutions for this problem but due to performance and project constraints; this solution was deemed too complex to implement in this version.
Repudiation	The use of strong authentication and encryption means that users can be identified easily. However the repudiation threat does not have harmful consequences for this system.
Information disclosure	Login details will be protected by encryption; however takes considerable time, processing power and bandwidth to encrypt all the other information transmitted to the client.
Denial of service	The current server system have resource and bandwidth throttling techniques for overcoming this threat and the system will validate and filter input to protect the application.
Elevation of privilege	The principle of least privilege will be used when accessing resources and the server's security system will prevent external access to the internal file system.

Table 5 - Threats and Countermeasures

6.7 Database Design

The entity relationship modelling technique was used to create the structure of the database. An entity relationship diagram is an illustration for describing high level abstract data models using a graphical notation (Chen, 1993). It contains four key elements: entities, attributes, identifiers and relationships. Entities represent real world object and relationships capture how two or more entities relate to one

another. Attributes are properties of entities for example the height attribute of a person entity. Entities must also have identifiers which “name or uniquely identify instances of entities” (Elmasri and Navathe, 2003).

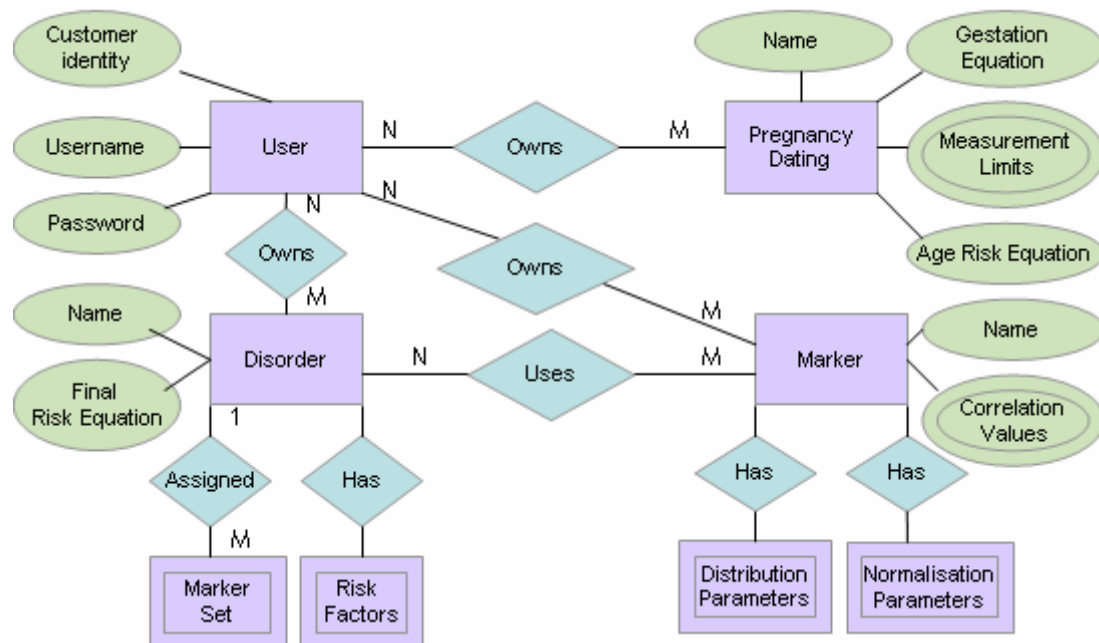


Figure 8 - Entity Relationship Diagram

The relational model enables the design of a consistent logical representation of information. The logical representation of the database was normalised to third normal form (Appendix E) to improve performance, avoid data duplication and inconsistency problems. The eventual relational schema to be implemented in SQL Server is shown below.

- User (**Customer Id**, Username, Password)
- Pregnancy Dating (**Id**, **Name**, Gestation Equation, Min Measurement, Max Measurement)
- Markers (**Id**, **Name**, Disorder, Min Value, Max Value, Gestation Range)
- Disorder (**Id**, **Name**, Age Risk Equation, Final Risk Equation)
- Marker Set (**Id**, Disorder, Markers List of between 1 and 5 markers)
- Distribution Parameters (**Id**, Marker, Distribution Equation, Weight Equation)
- Normalisation Parameters (**Id**, Marker, Normalisation Equation)
- Risk Factors (**Id**, Name, Disorder, Factor)

The design of the database has 8 relation variables/tables: User, Pregnancy Dating, Markers, Disorders, Marker Set, Distribution Parameters, Normalisation Parameters and Risk Factors. The bold underlined fields are primary/candidate keys and the non-bold underlined fields are foreign keys. A primary key is a unique record identifier and a foreign key points to a primary key of another database table. Foreign keys in this database will be governed by integrity constraints. The integrity constraints

are database rules that are executed on record update to maintain referential which means that a foreign key in a given table can only contain values in the primary key column of the referenced table.

6.8 Development Plan

The solution will be developed using a mixed evolutionary and waterfall model. When the requirements for a given component are clear and stable, that component is implemented using a Waterfall based model. When requirements are not clear or unspecified, an exploratory approach will be used to develop a prototype and user feedback will be used to improve the component until it is satisfactory. The user interface will be developed using an evolutionary approach. A skeleton graphical user interface will be initially developed to get user feedback and enable unit testing of the system's software components as they are implemented.

6.8.1 The Implementation Process

The software design will be transformed into executable code using the following guidelines:

- Analyse the design specification of a software unit and plan its code
- Implement the unit's functionality and write its code in accordance with Media Innovations Software development guidelines
- Test each software unit immediately after implementation before integration into the system
- Evaluate code and test results using the following criteria: satisfaction of requirements, appropriateness of coding methods, conformance to guidelines and performance
- Improve the software unit and repeat the testing process before updating the test plan and the implementation documents before moving on to another unit

After all components and improvements have been implemented, integration testing will be preformed and the application will be validated as explained in the testing chapter. Table 6 outlines the functionality to be added at each phase of implementation.

Iteration	Functionality Added
1	Project setup in the ASP.Net development environment, creation of directories, documentation files, setting permissions and test to make sure everything works
2	Database creation and query testing using SQL server query analyser
2	Skeleton login page with user identification fields without any styling or colours
3	Database class for retrieving user details from the database
4	Logging in functionality and the security features described in section 8.5.1
5	Improvement of the login page based on user feedback (evolutionary process) and interface design guidelines
6	Skeleton calculation page containing the fields required to perform calculations without

	any styling or colours
7	Skeleton results reporting page containing the fields required to display calculated results for unit testing purposes
8	Pregnancy Dating class implemented
9	Markers class implemented
10	Disorders class implemented
11	Improvement of the calculation and results page based on user feedback (evolutionary process) and interface design guidelines
12	Patient report page implemented and improved based on user feedback
13	Help system implemented
14	Testing and integration of: Pregnancy Dating, Markers and Disorders classes
15	Final verification and validation

Table 6 - System Functionality

6.9 Design Review

A design review was carried out to ensure that the system can proceed into implementation and meet the stated requirements in the available time and other system constraints. The review assessed whether the implementation platform provides the required capabilities and services needed for the implementation to succeed. In addition, the MI project supervisor analysed the functional, graphical interface designs to determine their completeness before implementation could begin. The manager's review evaluated system designs to ensure they correctly implement all system requirements and that requirements traceability is maintained in documentation.

7 Implementation

This chapter describes how the system was developed based on the design decisions and explains the code behind the system’s functionality. Due to document scope reasons, not all aspects of the system could be explained in detail in this chapter. Consequently only a limited range of fundamental or advanced functions of the system were described. Appendix H – Implementation explains in detail the code of a broader range of the final solution’s components.

7.1 Database

The database was implemented in SQL Server 2000 (Appendix G explain how to install the database). 8 tables were created, corresponding to section 6.6. The structure of all tables is identical to the structure defined during design. In addition all fields were assigned appropriate data types, for example numerical fields were given an ‘Integer or double’ data type. The relationship between tables was defined using table constraints to ensure referential integrity was maintained.

7.2 User Interface

The user interface web pages were created using ASP.Net. Usability and Accessibility guidelines were followed to improve the system’s usability, accessibility and appearance. The look of the application is similar to that of other software products developed by MI. Figure 9 shows a screenshot of the final calculation screen - the main page of the application.

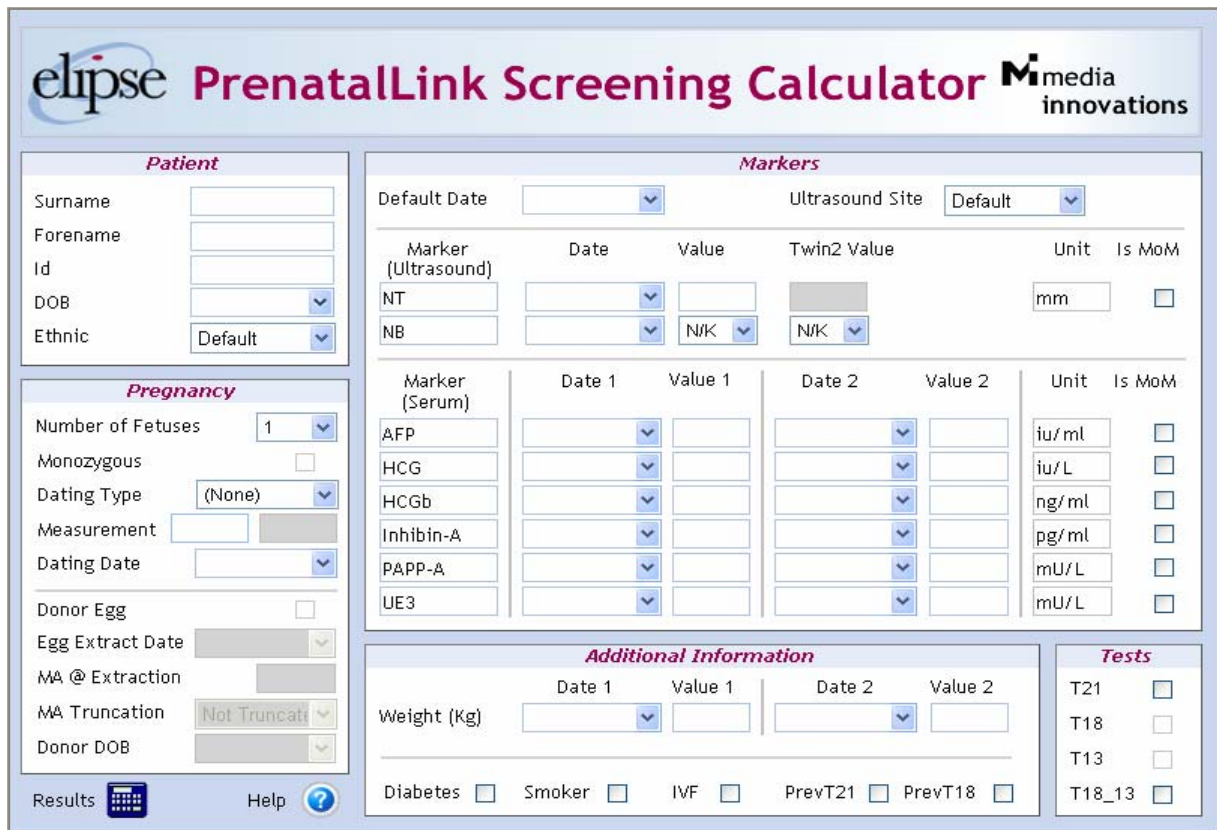


Figure 9 - Calculation page screenshot

7.3 Security

The system needs to protect sensitive information from unauthorised access. As shown in section 6.5, some protection features are provided by the security system of the server. As a result the application only needs to provide some of the absent security measures. The system protects login details during transmission to prevent session hijacking. Encryption is used to protect login details of a user by obfuscating them before transmission to the server. Figure 10 illustrates how login details are transmitted from the client to the server.

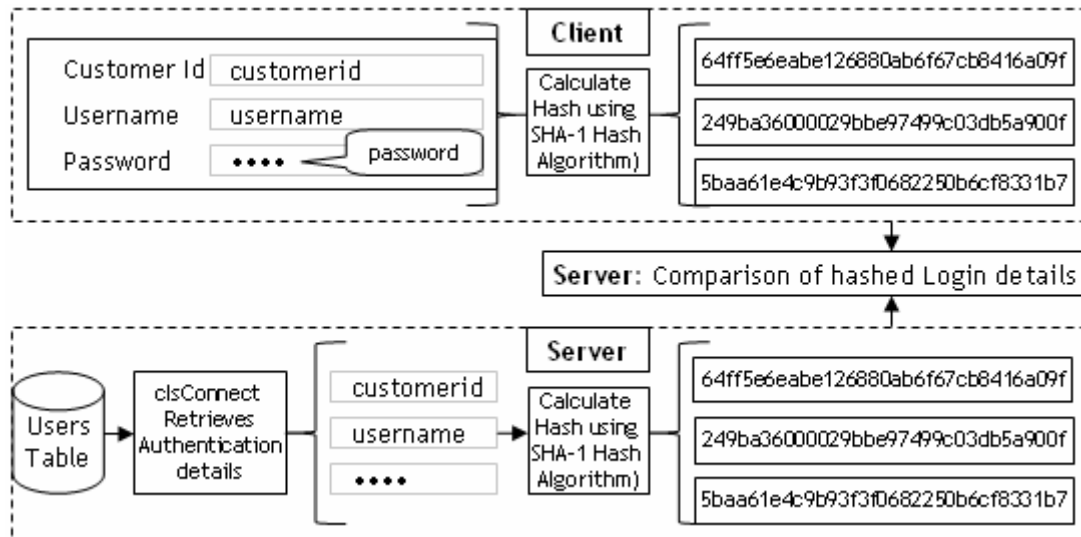
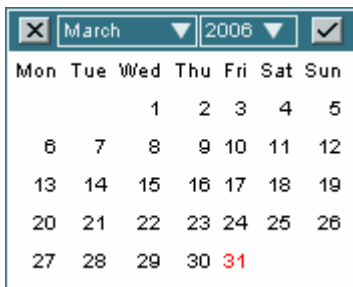


Figure 10 - Secure Hash Authentication

The security class uses ‘challenge hash authentication’ to avoid transmitting passwords in clear text. The password entered by a user is hashed and sent to the server where the hashed password is compared to the hash code of the stored password. In addition to hash authentication, users are tracked using automatically generated session keys. This ensures that users cannot access other people’s data, also, session keys are destroyed when users log out or when their session expires.

7.4 Data Entry and Validation

A date input control was created using dynamic HTML and JavaScript. The control enables users to enter dates in a convenient and efficient manner and prevents them from entering invalid dates.



The date picker is local sensitive and displays month names in the language of the user’s region. Users cannot enter invalid dates when they use this control since it only displays the valid days for a given month and year. Alternatively users can enter dates directly into a date field. Numerical fields and date validation is done on both the client and server sides to ensure users enter valid dates and to cater for

non-JavaScript browsers. The validation code for dates and numbers is locale sensitive and uses the correct thousands, decimal and date separators for the user’s region. Input validation is done on leaving a field for JavaScript enabled browsers and on the server side when a form is submitted.

7.5 Development Summary

This section describes the features that were implemented in relation to the system's requirements. The software for the application was developed according to Media Innovations software development standards which are reproduced with permission in Appendix H. The final solution's code is located on the accompanying compact disk at D:\Software\PrenatalLink.

7.5.1 Features Delivered

Figure 11 describes the features implemented to satisfy minimum requirements and some of the requirements extensions.

Requirement	Implemented Feature
Web application to allow calculation of results	The calculation feature was implemented in the Calculate class and is supported by three other classes: Pregnancy Dating, Disorders and Markers and the Elipse object.
Database system for data storage	An SQL server database was created using the schema described in section 8.6 and database classes, Connect and DBView were implemented to handle data access.
Data input and results reporting systems	The frmCalculate web page was created using ASP.Net web forms designer to enable data input
Security system for the application	A security class and login page were implemented to enable user authentication and for protecting login details during transmission.
Help system	A help was developed and can be accessed from the calculation page.
Date input system	A JavaScript/html based control was implemented to enable efficient entry of date values
Automatic client side input validation and session tracking systems	The application uses random session keys to track users and validates dates and numerical fields to ensure appropriate values are entered and to remove any code or injected SQL commands.
Page caching functionality for static web page components	All images and static content are cached on the client side, in order to reduce response times and bandwidth usage.
Wide browser compatibility	The application is usable in Internet Explorer, Netscape, Fire fox and Mozilla
Internationalising the application so that it customises to locales	The application displays dates in the user's language and validates both dates and numbers according to the user's locale formats.
An integrated help system that uses compiled help pages	A help system was developed and shows short messages explaining how to perform the current task.

Figure 11 - Requirements vs. Features Delivered

7.5.2 Features Not Delivered

Figure 12 describes the requirements extensions that were not implemented in the final solution and gives reason for their non-implementation.

Feature/Requirement	Reason for non-implementation
Batch calculation system that allows users to calculate results offline	Storing patient data would bring considerable complexity and security demands on the system. It would also put a huge strain on server resources, so it was decided not to implement this feature.
Import and export functions for file upload and download	This feature was no longer required since the batch processing feature was not being implemented.

Figure 12 - Extensions Not Delivered

In summary, all the minimum and additional requirements were met except for the batch calculation features that were not implemented.

7.6 How the System Works

Exception handling is done whenever a method is executed and an error message is displayed when an exception happens. The inbuilt try catch construct of ASP.Net was used to handle exceptions.

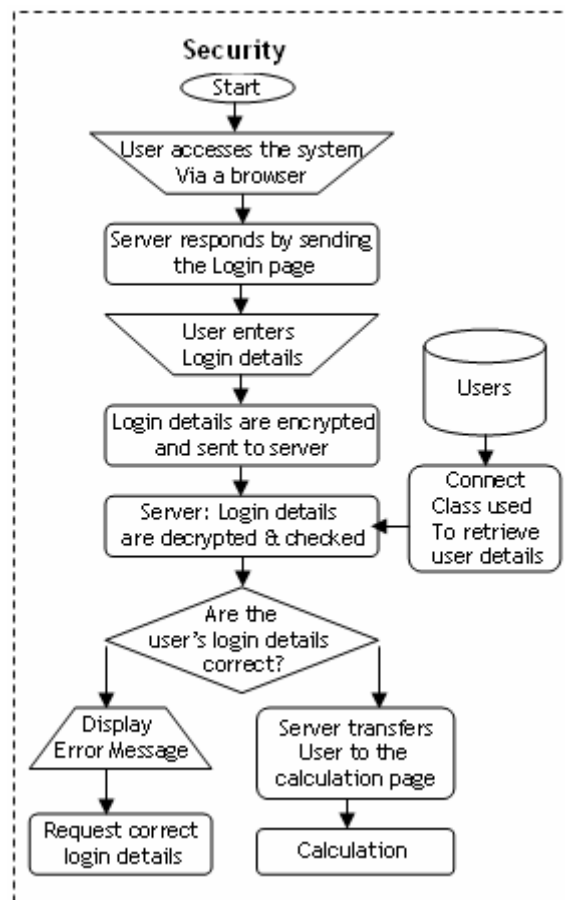


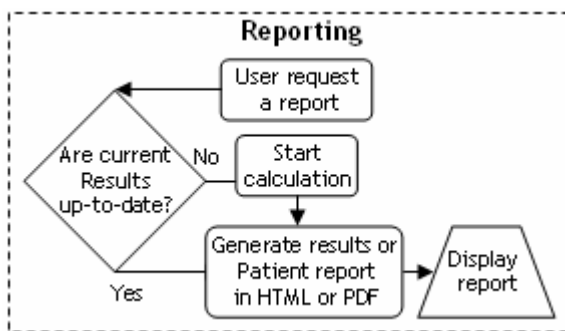
Figure 13 - System Flow Diagram

Users have to login first before they can access the system. If they successfully login, they will be transferred to the calculation page. On the calculation page, users are required to provide valid clinical

information in the fields of the calculation page. This information is used to calculate Pregnancy Dating, Markers and Disorders results. User input is validated immediately on leaving a field and if the input data is invalid, an error message is displayed to notify them of the error. On the server, more comprehensive validation is carried out to ensure input values are clinically valid and within acceptable ranges. Only when the provided information is valid does the system proceed to calculate results and generate patient/results reports.

7.7 Reporting System

The application produces Results and Patient reports in PDF and HTML formats. When a user calculates results, a results page is automatically displayed showing the calculated results and the user can request a PDF version of the report which they can print, save or open on their machine.



The figure to the left shows a flow chart illustrating the routine for displaying reports. The Patient report is generated dynamically in code and downloaded to the client. Appendix H explains how a PDF report is generated in memory. Figure 14 below shows a screenshot of the Results page; see Appendix H for a sample patient report.

Results						
Patient		Markers				
Surname	SSS	MoM	Gestation	Twin2		
Forename	FFF			MoM	Gestation	
Id	DD	NT	0.70	12 + 4	4.00	12 + 6
EDD	27 Oct 2006					
MAEDD	23.82					
Preview Print						
Risks						
	Prior Risk	Final Risk	Result	Final Risk	Result	
T21	131	353	Negative	4	Positive	

Figure 14 - Results Page

8 Testing

Testing was conducted throughout the implementation process; this chapter explains the validation or verification of the system. Validation tests check whether we are building a system that meets the user requirements. Verification test cases check the code to ensure it works correctly. The software was subjected to unit, integration and system testing, unit testing checked every unit of the system to ensure that it worked as expected while integration testing checks whether a collection of units interoperate correctly. At the end of the testing process, system testing was done to check whether the system was acceptable to users and assess how it coped in various conditions. During the analysis and design stages, verification test procedures were used to review the deliverables of each completed stage. The reviews were conducted in conjunction with the MI project supervisor. The specification review would check that the specification document captured all the user requirements and that the requirements were feasible.

8.1 Code Testing

Two main techniques were used to test software units: white box and black box testing. In White box testing, test cases aimed at checking code fragments such as loops, algorithms and conditional statements to ensure that they executed correctly. Black box test cases tested the functionality of the code without regard to how the code worked. Black box test cases were based on checking the visible functionality of the code by validating the results calculated by the system.

Unit Testing – Soon after implementation, a unit was individually subjected to white and black box test cases. All the units that it interacted with were replaced by dummy units. For the purposes of this report, a unit was assumed to be a single class. Small programs were written to automate most unit tests and to act as dummy units. During unit testing when an error was discovered, the developer would resolve the error in code before repeating the test again to ensure the unit worked properly. Because unit testing was done soon after implementing a given unit, not all the errors rectified during the implementation could be documented in the test plan. The test plan can be found in Appendix E.

Integration Testing – When units had successfully gone through the unit testing procedure, they were combined with other units to form an interoperating, coherent component. The emphasis of integration testing was on checking that units work properly together. The main components of the application were: security, results calculation, data access and the reporting system. See Appendix E – Testing for the integration test cases.

System Testing – After finishing implementation, unit and integration testing, the performance behaviour of the system was checked to ensure that response times were acceptable and that the system could cope with normal levels of usage and data. In addition, the system underwent tests under extreme conditions such as high numbers of users and unexpected input data. All tests checked that the system produced correct results and that the system's exception handling system produced the

correct error messages. The security system was also tested to make sure unauthorised users could not gain access to the system. The system testing section of Appendix I- Testing describes the system test cases and their results.

Acceptance Testing – User acceptance testing encapsulates the test procedures that result in the formal acceptance of the system. End users of the system were actively involved in this test. The test plan for acceptance testing was primarily based on the user requirements, to which the system must conform. User Acceptance testing was done to scrutinise the performance of the system in real world conditions, and to ensure that the user is satisfied that the system meets their requirements and performs at an acceptable level. The MI project supervisor was responsible for accepting this software solution; therefore she was the appropriate person to carry out user acceptance testing. A user acceptance testing report written by the MI project supervisor can be found in Appendix F.

8.2 Testing Results

The primary objective of the code testing process was to uncover as many errors as possible and fix them before the acceptance testing was done. White box testing enabled the developer to uncover and correct most logic or program structure based errors. As described by Hetzel (1988), white box tests are typically applied to small program components and in complement black box testing validates the functionality provided by the system. All errors discovered during unit testing were resolved before integration testing was initiated. System testing test cases evaluated the functionality of the system and its performance. The evaluation chapter explains in detail the results of testing the final solution.

After completing code testing, user acceptance testing was conducted by Mrs C Wilson (MI Project Supervisor). The system met acceptance standards in almost all cases. The one requirement that the system did not meet was the batch calculation functionality, but in this case, both the developer and the user agreed that providing this function was infeasible under the current project constraints. Therefore both the developer and Media Innovations Ltd agreed that the system had passed the acceptance testing.

As experienced developers say “testing never ends, it just gets transferred from the developer to the user and every time they use it, they will be performing a test” (Klein, 2005) so in essence, our aim was to design test cases that were as complete as possible in the hope that we would uncover the highest number of errors and correct them before user testing began in earnest.

9 Evaluation

The main aim of this section was to appraise the system’s functionality and to retrospectively assess the quality of the system. It also evaluates whether the intended benefits have been achieved and identifies unintended benefits of the system to end users. The evaluation processes assess the final solution by describing its good features and where applicable show how things could be improved. It concludes with a description of additional features which would be useful additions to the system.

9.1 Evaluation Criteria

The evaluation of the system was based on the success of integral functions and the evaluation criteria set out in section 1.9 during project conception. The criteria are based on established software evaluation standards. The International Standards Organisation (ISO) specifies six characteristics for software evaluation (ISO 9126, 2003) as shown in figure 15. The widespread use of these standards for evaluation means they are a useful means of appraising software (McMullan, 2006).

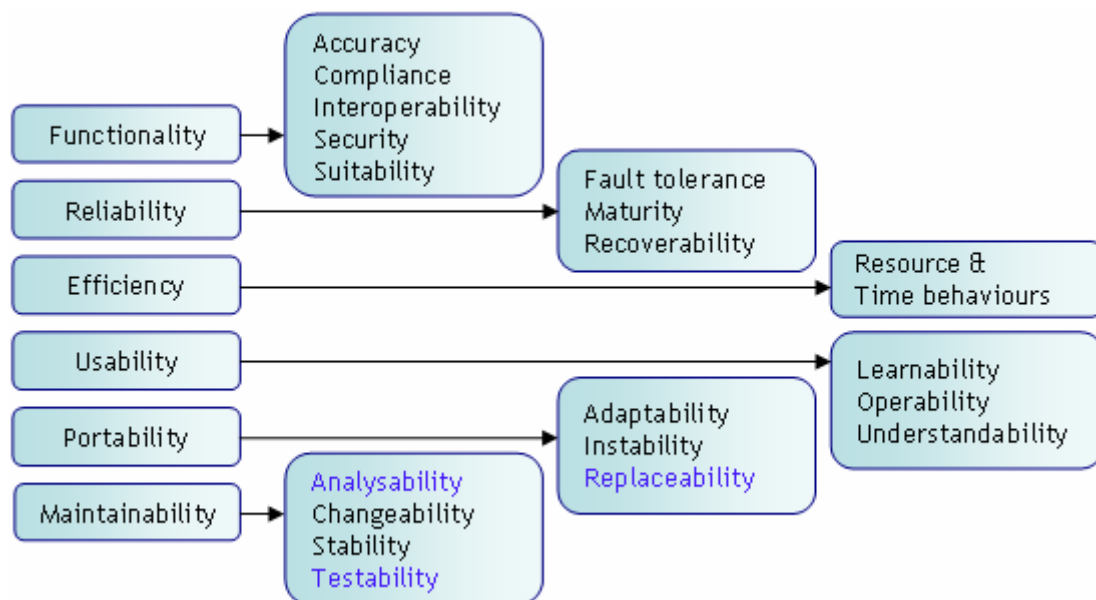


Figure 15 - ISO Software Evaluation Criteria

The evaluation criteria for the solution are:

- **Functionality:** How well does the system satisfy user requirements, aims and objectives?
- **Reliability:** The results of testing will be used to demonstrate the accuracy of functionality, fault tolerance and recovery.
- **Efficiency:** The performance and speed of response of the final solution.
- **Usability and Accessibility:** Usability is the effectiveness, efficiency and satisfaction with which users achieve specified goals. Accessibility is the opportunity for everyone regardless of technical or physical inclination, such as colour blind people, to access software products.

- **Portability and Maintainability:** Portability is the relative ease with which the system can be installed across platforms. Maintainability is the ease of making modifications and additions to a software system.
- **User evaluation results:** In addition the solution was compared to existing systems and subjected to user acceptance testing and user evaluation was used to evaluate the system.

9.2 Functionality

Functionality refers to the services provided by the system to satisfy user requirements. The solution's requirements were specified through project aims, objectives and requirements specifications. It is logical to assume that the success of the system is judged on how well and to what quality it satisfies user requirements, aims and objectives.

9.2.1 User Requirements

Table 7 evaluates the quality of the functionality delivered by the final system and the extent to which it satisfies the main requirements of the solution.

Requirement	How it was achieved
Web application for calculating risk results	During implementation and testing, tests were carried out to check the performance of the system's components. The system responds in real time to most functions that are executed on the client side. This means that the quality of the system's performance is very high in comparison to existing systems. The system can be easily used over a network enabling a screening centre to monitor all operations from a single server machine. The system delivers good usability and minimal understanding of prenatal screening is required to use the system. The system delivers more by means of a multi-stage results reporting system that interprets the results for the user, giving them more valuable information.
Database system for data storage	During implementation, a database was created for storing settings data. The database was normalised to third normal form, which means the database organises data efficiently and reduces the potential for anomalies during data editing. The database organises data in a consistent manner as a result and allows quick data access and updating with minimal risk of causing inconsistencies.
Data input and reporting systems	The data input system enables users to input values in a convenient manner; it validates input immediately after editing to ensure that users get real-time alerts to invalid inputs. The validation checks input according to the user's local formats such that a French user can enter a number with French decimal and thousands separators and the number will be validated appropriately and interpreted correctly. The reporting system produces reports in various formats so that users can save the reports, e-mail them or print them. The PDF format for patient reports enables users to secure reports on a file level, use them across platforms

	and the appearance of printed PDF files is of high quality.
Security system for the application	The system authenticates users and encrypts user login details to prevent unauthorised access. The technique used for encrypting data is secure and no encryption keys are transmitted from the client to the server. The security system is efficient and fast because it does not decrypt user details; instead it encrypts the comparison details stored on the server and compares them to the encrypted login details sent from the client-side. This protects the server from any malicious input. However, the security system is not comprehensive; there are some threats that still need to be mitigated for the system to fully protect data during transmissions.
Help system	The help system created for the application is adequate for most purposes. All the tasks that a user might want to carry out on the system are explained in detail. The user guide was written in a concise and straightforward manner that enables the user to quickly find the information they need.
Date input system	The HTML based date picker was implemented to enable efficient entry of date values. It prevents users from entering invalid dates and speeds up the date entry process. This control is cached on the client such that the control needs to be downloaded only once every sixty days, greatly improving bandwidth usage.
Page caching functionality	All images and static parts of web pages are cached on the client side to reduce response times and reduce bandwidth usage. The system's pages take less than 5 seconds to download on a broadband internet connection.
Internationalisation	The application displays dates in the user's language and validates both dates and numbers according to the user's locale formats. The application automatically detects user's date and number formats and displays month names in the correct language. However the system could be improved so that it displays text in labels using the language of the current user.

Table 7 - Requirements vs. Implemented Features

9.2.2 Aims and objectives

This aim of the project was to develop a web application that will accept maternal, biochemical and ultrasound test results from clinicians and calculate the likelihood of the given pregnancy being affected by congenital disorders, especially those caused by chromosomal anomalies. The final solution calculates the risk value for a pregnancy. It uses some of the latest ultrasound markers such as Nuchal Translucency and Nasal Bone and calculates results for more than eight disorders. The system can calculate results for combinations of between one and eight markers from different sample dates enabling results for multiple markers to be calculated in a single transaction. The calculation of risk values is amongst the most mathematically complex tasks in medical support (Wilson, 2006) and the level of interpretation provided by the system in results presentation makes the system very useful indeed.

9.3 Reliability

Software reliability is the “ability of a system to perform its required functions under stated conditions for a specified period of time” (IEEE 982.2, 1987). The correctness of the results generated by the system is crucial to its quality and reliability: incorrect results are unacceptable. The system was tested using unit, integration, system and acceptance tests. The unit tests were carried out soon after implementation to resolve any coding errors and to check program logic. After verifying that the code of a unit was correct, the unit was integrated into a larger system component such as the security component. The rationale for unit tests was to resolve coding errors immediately after implementation. In addition it reduced the number of errors that were likely to be encountered during later testing stages. Integration testing was carried out thereafter, the main focus of this test was to prove the compatibility of a given unit to the units it would interoperate with. These tests enabled the developer to discover integration errors when the knowledge of the likely cause of the error could be easily discovered and to reduce the number of post implementation errors and interoperability issues.

System testing was of great importance to the evaluation of the system. It emphasised on checking the correctness of the results calculated by the system and verifying that the system was working correctly, interpreting user input and settings data correctly. The system calculates correct results based on the comparison of the results it produces to the results generated by the MI calculator. The pregnancy dating results that it generates were verified by comparing them to manually calculated results. Several test cases were executed to test the performance of the application in extreme conditions such as large numbers of settings records, low bandwidth and large numbers of users. The results of these tests showed that the system was reliable and could cope in extreme conditions as demonstrated in Appendix E. In addition, the system was subjected to invalid input tests to check how well it recovers from errors or exceptions. Exception handling tests results (Appendix E) demonstrated that the system always displays an error message and cancels the current operation when a run-time error occurs. All software routines of the solution use the error handling routine explained in section 7.6 to ensure faults are detected, handled and reported to the user. The validation system aims to prevent errors from happening by validating input to prevent buffer overruns, data type conversion errors and formatting errors. Indirectly, the validation of input also increases reliability because users can only enter valid input which the system is capable of handling. Since the 10 of April 2005, the beta version of the system has been operating on the MI web server without any reported faults (Wilson, 2006).

9.4 Efficiency

The efficiency of the system measures the system’s level of performance in relation to available resources. The main efficiency measurements criteria for the solution are response time, bandwidth and storage usage. The total size of the whole system is 11.9 megabytes (MB) and during operation, this value will only be increased by additional settings records added to the database, making its usage

of storage space very efficient. The response times of the application are less than 5 seconds on average over a 1 MB connection since the mean size of the application's pages is less than 1 MB. The patient report is 256 kilobytes in size and takes a second to download over a megabyte connection, showing that the response times and performance of the application is acceptable.

9.5 Usability and Accessibility

The system was designed to meet Nielsen usability and MIT accessibility guidelines. The purpose of such guidelines is to ensure that the system provides an easy to use interface that is accessible and meets acceptable usability standards. After implementation, the system was evaluated to check that it met the requirements of these guidelines. The evaluation checked every guideline against the system to see if the system complied with all guidelines. The results of this evaluation showed that the system met all the guiding principles except for five guidelines which were not crucial or irrelevant to this application. Appendix E shows the guidelines that were used to evaluate the system's usability and accessibility.

Additionally, the system's user interface uses simple well known medical terms in labels. With minimal understanding of prenatal screening, a novice who has access to the correct clinical input data can efficiently use the system in a straight forward manner and with minimal user involvement.

9.6 Portability and Maintainability

The final solution is a web based application. Having been implemented using ASP.Net, the application is installable on any web server and the software required to run the application is easily accessible at a small cost. However any machine can access the application via a web browser. The system can also adapt to different user locales and will automatically detect user's locales and display month names in the user's language and validate input using the user's regional formats.

The system's code was written in an object oriented manner, with classes that can potentially be extended and inherited without having to modify existing code. Should the need arise to modify the solution, the code is well documented and implementation documents were created to ensure the system is maintainable. These software and technical documents were reviewed by the MI project supervisor to check that they complied with the MI software engineering guidelines.

9.7 Comparison to existing systems

The Analysis chapter gave an analysis of currently available systems and described their advantages and disadvantages. It seems fitting to evaluate the current system to ensure it does not suffer from the same drawbacks and that it delivers more than the existing systems.

The previous system developed by MI did not offer the latest marker technology; it was not network enabled and was only Windows based. The final solution incorporates some of the latest research findings in prenatal screening technology; it is configurable and can be hosted on a web server making it usable over a network. The final solution is more efficient, and provides a better reporting

system that produces multi-stage results so that users can see the results of every important calculation stage. The application hides the complexity of results interpretation from the user and uses a more improved database that is normalised to third normal form. Benetech PRA (2006) is a good system with an advanced data storage system and good security. However it is rather limited in terms of the number of markers that can be used in calculations. This means the final solution is better since it enables users to use the most suitable marker collection for their purposes. In addition, the current database system of final solution is adequate for the current market needs. In conclusion, the final solution gives added benefits to users; it meets the needs of the market that were identified in the project summary. The system provides an improved quality of service than most of the existing systems and delivers functions that utilise some of the latest developments in prenatal screening.

9.8 User evaluation

The MI project supervisor carried out a system evaluation on completion of the beta test version. The evaluation was split into three sections, the first analysed how well the system satisfied user requirements. The second described the tasks that were required to create a satisfactory system and the last section was an evaluation of the system in terms of performance, usability and efficiency. Mrs C Wilson, the MI project supervisor compiled a user evaluation document which was reproduced in Appendix F – Evaluation. The user evaluation showed that MI was satisfied with the final solution's performance and level of functionality.

9.9 Evaluation Results

In conclusion, the system achieved the required level of functionality. The final solution met all the user requirements except for a small number of supplementary requirements. It performed reliably and efficiently during testing and generates responses promptly. The system is highly portable and can be accessed from most platforms with web browsers. In addition, the system can be installed on most web servers which have an installation of the requisite software. The system met over 95% of the Nielsen usability guidelines and 90% of the MIT accessibility guidelines. The system passed the user evaluation and performed satisfactorily from the developer's and user's point of view, therefore the final solution satisfies all the evaluation criteria.

9.10 Suggested improvements

During post-implementation reviews, testing and user evaluation, it was realised that some functions of the system could be improved and new ones could be added. These improvements would enable the system to offer better efficiency and security.

During evaluation users raised a need for the date picker to have a dynamic dropdown control for the year so that they could select or type in the year. This would give users the option of typing in the date when selection from a drop down list would take considerable time. In addition, users felt that the date picker must provide a textbox showing the current value of the date to inform them of the selected date.

The security system of the application could be further improved so that all sensitive information transmitted between the client and the server is encrypted or protected in some way. Currently the application only encrypts authentication details during transmission from the client leaving clinical information vulnerable to electronic eavesdropping. In future, implementing such a feature would be a useful improvement.

The current help system is not adequate; a search utility that would enable users to enter a keyword and display the relevant help topics would make help more accessible. It would enable the user to quickly find the information they are looking for.

The application validates numeric input and dates using the user's locale formats. However non-numerical fields are not validated, it would be useful to implement code that validates non-numerical input such as names in future versions. The application performs basic validation – numerical and date validation on both the server and the client; this is done to cater for clients whose browsers are not JavaScript enabled. Ideally the application should be able to do more complex validation on the client side as well; checking that values are within acceptable ranges as is done on the server would be more efficient if done on the client where possible. This would prevent unnecessary delays because a user would be informed of an invalid value soon after inputting it, greatly improving the system's performance and usability.

When a user moves out of the application using the back and forward browser navigation buttons, they can come back into the application by navigating in a similar way. This is a security vulnerability since an attacker might be able to access a page of the application that a user has just visited beforehand. Future versions of the system must monitor the navigation on the client browser so that when users navigate out of the systems boundaries, the system automatically logs them out and destroys their session variables to prevent anyone from accessing the session data or pages the user might have visited earlier. This solution would greatly improve security and reduce wastage of server resources used to maintain session data.

The batch calculation function was cancelled due to project constraints; however future versions of the system must provide an offline batch calculation service to serve the needs of customers who handle large numbers of patients.

Finally the performance of the system could be optimised by the use of embedded images. This would ensure that images are transmitted to clients before the system needs to display them, greatly improving response times and the appearance of the system.

10 Conclusion

The main focus of this project was to develop a web system that would enable clinicians to calculate the likelihood of a pregnancy being affected by a given disorder. This project satisfactorily achieved that objective; however there is still room for improvement. The developer aimed to deliver functionality first before embarking on other secondary aims of improving the usability of the application.

The usability of this project was based on the Nielsen's usability guidelines. The application was a success in terms of usability according to those guidelines because throughout the implementation stage, newly developed components were analysed to identify possible means of improving usability. Users and Media Innovations Ltd were fully involved in the development process and this meant they had a sense of enthusiasm about the project due to their involvement.

After implementation, the application went through testing and validation processes. These processes evaluated the correctness of the results calculated by the application. The testing process was made up of several cycles of error discovery and resolution. On completion of testing, the user carried out acceptance testing to check if the program met their requirements. Because the user had been actively involved throughout the development process, acceptance testing was a secondary procedure for the user to perform a final check on the system.

This project achieved the set objectives and requirements although there is still room for expansion. Some functions which had not been thought of during requirements elicitation were realised to be useful if implemented, in particular, the security of the system could be made more comprehensive. In addition the batch calculation functionality would be a welcome addition for some customers if it were implemented in future versions. Overall, the initial reaction from the company and the MI project supervisor was very positive, and the high level of user motivation indicates that they are likely to adopt the system. However, the real test of success will be based on the number of users who will use the system on a regular basis and adopt it as their main prenatal risk calculation system, although it is early to predict anything, the prospects of the system look positive.

11 Bibliography

- [1] Agar Michael, (1996), **The Professional Stranger: An Informal Introduction to Ethnography**, Academic Press.
- [2] Boehm, (1991), **Software Risk Management, Principle and Practices**, IEEE Software, Vol. 8, January 1991, pp. 39-44
- [3] Chen, (1993), **The Entity-Relationship Model - Toward a Unified View of Data**, ACM Transactions on Database Systems Volume 1 Number 1- pp 9-36 (1993)
- [4] Crump, (2005), Risk Management, http://www.saferpak.com/project_management_art2.htm, [10 March 2005]
- [5] Cuckle H, (1997), **Estimating a woman's risk of having a pregnancy associated with Down's syndrome using her age and serum AFP**. Obstet Gynaecology Journal 1997; Volume 94;387-402
- [6] Cunniff C (2004), **Prenatal screening and diagnosis for paediatricians**, Paediatrics 2004 Sep; Volume 114 pp 889-94.
- [7] Design Patterns, (2004), The Model-View-Controller Design Pattern http://www.adobe.com/devnet/flash/articles/mv_controller.html
- [8] Easterbrook and Nuseibeh, (1996),. **Using Viewpoints for Inconsistency Management**. Software Engineering Journal, Volume 11, No 1, Jan 1996
- [9] Elmasri R, Navathe S, (2003), **Fundamentals of Database Systems**, fourth edition, Addison-Wesley, pp 177-182.
- [10] GM Stocks, (2005), **Related Articles, Links Spina bifida, tethered cord and regional anaesthesia**, Anaesthesia PubMed volume 60, 2005 Nov; 60; pp 1114-9.
- [11] Haddow J.E, (1999), **Screening of maternal serum for fetal Down's syndrome in the first trimester**, 1998 pg 955-961
- [12] Hernandez D and Fisher EMC (1996), **Down syndrome genetics, unravelling a multi factorial disorder**, Publisher: Planet papers, pages 1411-1416, 1996.Medline Plus, 1996
- [13] Herndon (1997), **Software Productivity Consortium: Evolutionary Rapid Development**. SPC document SPC-97057-CMC, version 01.00.04, June 1997. Page 9.
- [14] Hetzel (1988), **The Complete Guide to Software Testing. QED Information Sciences**, Wellesley, Massachusetts, second edition, 1988.
- [15] Hook EB (2000), **Rates of chromosome abnormalities at different maternal ages**. Obstet Gynaecology Report 2000; Volume77, pg 282-5
- [16] Horton, (2003), **Beginning Java 2, JDK 5 Edition**, ISBN: 0-7645-6874-4 Dec 2004. pp 99 - 111

- [17] IEEE (1998), **Human-centred design processes for interactive systems**, International. Standard - ISO 13407.
- [18] IEEE 982.2 (1987), **Guide for the Use of Standard Dictionary of Measures to Produce Reliable Software**.
- [19] Isaacs (2006), The Media Innovations Ltd <http://www.media-innovations.ltd.uk/mainframeset.htm>
- [19] ISO 13407 (1999), **Human-centred design processes for interactive systems**, International. Standard - ISO. 13407.
- [20] ISO 9126 (1991), **Information technology - Software product evaluation -- Quality characteristics and guidelines for their use**
- [21] Jones (1999), **Discussions with Dr Rick Jones**, Leeds Teaching Hospitals Trust and a Media Innovations Ltd Medical Consultant
- [22] Julie McMullan (2006), **Achieving Software Product Quality**, ESSI SCOPE, Publisher: Den Bosch, pp 76-87
- [23] Klein (2005), **Software Project Secrets- Why Software Projects Fail (Expert's Voice)**, Apress 2005
- [24] Kotonya (1998), **Requirements Engineering, Processes and Techniques**, John Wiley & Sons pp17
- [25] Kristol (1993), **Picture Perfect: The Politics of Prenatal Testing Elizabeth Kristol** 1993 pages 17-24. Publisher First things, the journal of Religion, Culture and Public life
- [25] Lerdorf (2002), **Programming PHP**, O'Reilly Media, Inc.; 1 edition pp 2
- [26] Liang, Y. Daniel, (2003), , **Introduction to Java Programming**, Fourth Edition ISBN: 0022R-5, 0-13-100225-2
- [27] Mahoney (1990), **The Ethics of Sex Selection. In, Medicine, Medical Ethics and the Value of Life**, John Wiley & Sons, Chichester, 1990. pp. 111–17.
- [28] Microsoft (2005), Overview of Web Application Security Threats – <http://msdn.microsoft.com/library/security>
- [29] MSDN (2006), MSDN home
<http://msdn.microsoft.com/netframework/gettingstarted/default.aspx>,
- [30] Oncology (1999), **Molecular Biology of Neuroblastoma**, Journal of Clinical Oncology, Vol 17, Issue 7 (July), American Society for Clinical Oncology

- [31] Pressman R.S (1992), **Software Engineering, A Practitioner's Approach (third edition)**. McGraw-Hill, New York. 1992. pp 66.
- [32] Python (2006), Python home, <http://www.python.org/doc/faq/>
- [33] Royce (1987), **Managing the Development of Large Software Systems**, IEEE Computer Society, 1987, Vol 9, 13 - 37
- [34] Schaum (2001), **Schaum's Outline of UML**, McGraw-Hill, ISBN, 0077096738
- [35] Schwartz (1997), **Learning Perl (A Nutshell Handbook)** Publisher, O'Reilly, ISBN, 1565922840
- [36] Scott (2005), (Wal-Mart chief executive), "Wal-Mart to expand health cover", <http://news.bbc.co.uk/2/hi/business/4745464.stm>
- [37] Shaffiq (2004), Elipse Homepage, <http://www.elipse.org.uk/>
- [38] Smith-Bindman (2001), **Second-trimester ultrasound to detect fetuses with Down syndrome** pg 1044-1055,
- [39] Somerville (2004), **Software Engineering**, Pearson and Addison Wesley, 7th edition, 2004.
- [40] Stellman and Greene (2005), **Applied Software Project Management**, O'Reilly Media. ISBN 0596009488, pp 44-49
- [41] Stepanek (2005), **Why Software Projects Fail**, ISBN 1590595505, Publisher: Apress
- [42] Sun (2005), JavaServer Pages[tm] Technology - Comparison with ASP, <http://java.sun.com/products/jsp/jsp-asp.html>
- [43] Thompson SG (1997), **Estimating a woman's risk of having a pregnancy associated with Down's syndrome using her age and serum** pg 387-402
- [44] UKGTN (2005), ACOG Practice Bulletin. **Prenatal diagnosis of fetal chromosomal abnormalities**. Obstet Gynecol. Vol 97(Part 1),
- [45] Wald N.J, Rodeck C, Hackshaw A.K, Walters J, Chitty L, Mackinson A.M (2004). Journal of Medical Screening, Volume 10, Number 2, pp. 56-57 Publisher, Royal Society of Medicine Press
- [46] Wall (2000), **Programming Perl**, O'Reilly Pages - 1104
- [47] Westerway (2000), **Ultrasonic fetal measurements, new Australian standards for the new millennium**. Obstet Gynaecology 2000 Aug; Vol 40(part 3), 97-102
- [48] Wilcox (1993), **Estimating a woman's risk of having a pregnancy associated with Down's syndrome using her age and serum 1997**; pg 387-402
- [49] Wilson (2006), **Discussions with Mrs C.J. Wilson**, The Media Innovations Ltd Scientific Software Department Manager

12 Appendix Table of Contents

1	Appendix A: Reflection	57
2	Appendix B: Glossary	59
3	Appendix C: Introduction	61
3.1	Project Schedule	61
4	Appendix C: Analysis	63
4.1	Risk Register	63
4.2	MI Software Engineering Guidelines.....	68
5	Appendix D: System Requirements	75
5.1	Interviews	75
5.2	Functional Requirements	77
6	Appendix E: Design	81
6.1	Database Normalisation.....	81
6.2	Usability Guidelines	82
6.3	Accessibility Guidelines	83
7	Appendix F: Implementation	86
7.1	Database	86
7.2	Security.....	86
7.3	Validation	87
7.5	Reporting System	88
8	Appendix E: Testing	90
8.1	Test Plan	90
8.2	Test case specifications	91
8.3	Calculation Test Cases	95
8.4	Requirement vs. Test Case Traceability Matrix	102
8.5	Testing Results	103
9	Appendix F: Evaluation	104
9.1	Media Innovations Ltd Project Evaluation.....	104
10	Appendix G: Product Content	106

10.1	Installation and Setup	106
10.2	Implementation Document	107
10.3	User Guide.....	108

13 Appendix A: Reflection

Like every other major project, my final year project was both enjoyable and demanding at times. I gained a lot from the experience from developing a big application and the responsibility that came with the solution I was implementing. Dealing with prospective end users and other stakeholders really widened my people handling skills and enlightened me to the industrial world. This project enabled me to learn many skills, technologies and improved me in many ways; the paragraphs below explain ten different lessons that may be useful to other students undertaking a final year project in the future.

Plan before doing things – taking a moment to organise things, imagine and analyse possible solutions to a problem is very important. Doing so will help you to do things using the easiest and time efficient approach. Plan before doing most tasks, write up detailed plans for major tasks and think up smaller plans for mundane tasks before starting to do them. In addition, planning enables you to do things in a structured manner that makes it easier to manage your project and ultimately enable you to complete it successfully and on schedule.

Project schedule – Create a reasonable project right at the start and stick to it. Spare adequate time for completing a task before starting another one. Take note that when writing up and implementing the solution, problems do arise and they might take time to solve so leave some unallocated time between some tasks to ensure that you don't fall out of schedule as soon as a delay happens.

Background reading – Although it is important to read a very wide variety of books for background research purposes, take some time to validate the relevance of articles before reading them. This will not only save you time but will enable you to get quality information whilst saving you time.

Modules – Since the project is worth 40 credits, it requires a lot of work and time; I therefore recommend doing at least 6 modules during the first semester, such that in the second semester you will have 3 modules to do. This will give you more time to work on your project.

Write-up – Begin the project write-up as soon as you start working on your project, it is better to refine what you already have than to not have anything at all. The write up takes a

substantial amount of time to do and accounts for a major portion of the marks so I found it useful to start the write up during the project conception stage.

Coding – I found out that writing your code in a clear, well documented manner will not only enable you test and debug the code easily but will make it easier to write the implementation chapter.

External Companies – My project was proposed by an external company, when writing up the report, I was assigned a project supervisor to be my point of contact. I felt that it is very important to take the users into consideration whenever possible. They will be using the system; therefore their needs must be satisfied by application. In addition try to establish the user requirements as early as possible to make sure that your plans are appropriate for the project.

Project Supervisor and feedback – Over the years your project supervisor will have seen many final year projects, arrange to have weekly meeting with your supervisor to get feedback from him. The meeting will enable you to get constant help and advice as well as inform you supervisor of your progress. I found the feedback very useful, by meeting my supervisor every week, it meant that he clearly understood my project and was always up-to-date on what I was doing and could give me advice proactively.

Team work – some of this project's development had to be done at Media Innovations Ltd. During this time I was required to work with other Media Innovations employees and this project's MI supervisor. Developing a project for an external company gave me the opportunity to experience how things are done at a professional level. I would advise future students that developing a solution that solves a real problem simplifies a lot of tasks and may give you the opportunity to improve skills such as team working. When dealing with third parties always plan for the meeting beforehand and behave in a highly professional manner, in addition, value their time because most people are likely to be very busy.

My project enabled me to learn the ASP.Net framework and VB.Net. It also honed some of the skills that had been learnt during my degree. Overall I feel that my project was very successful, the aims, minimum requirements and objectives were achieved to what I felt was a high standard and on schedule. The solution that was implemented is currently undergoing clinical validation and trial before being launched online if all goes well. It gives the author great happiness in knowing that this project produced a useful application that could improve and save lives. Working on this project has been a very enjoyable experience and a good learning process. I believe that I could have improved things by starting my write-up earlier.

14 Appendix B: Glossary

Alpha-fetoprotein (AFP)	A protein made in the liver of the fetus. Low levels of AFP in the second trimester (14 – 26 weeks gestation) may indicate a T21 positive pregnancy
Amniocentesis	A procedure in which a needle guided by ultrasound is inserted into the uterus and with draws some amniotic fluid for cell analysis and diagnosis
Amniotic fluid	Protective fluid that fills the sac surrounding the fetus
Anatomical malformations	A structural malformation of a fetus
Ancephaly	A defect in the closure of the neural tube during foetal development
Anencephaly	A failure of the neural tube to close properly in a fetus, it may also have symptoms such as incomplete brain and skull bones
Application	The program written for this project
Biochemical markers	Maternal Serum hormones that are present in a pregnant woman's blood such as AFP and UE3
Chorionic Villus Sampling	A small amount of villi (hair-like fringes of the placenta) is removed using either a needle inserted into the abdomen
Chromosomal Abnormalities	A disorder caused by the abnormal distribution of chromosomes that carry genes
Chromosome	A component of a cell that contains genetic information (DNA and proteins)
Clinical phenotype	The expression of the genes present in an individual e.g. blood group
Clinical phenotype	An image of a persons chromosomes used to screen for disorders
Congenital Disorders	Disorders that are present at birth
Cytogenesis	The formation, development and variation of cells
DBMS	A Database management system, it handles the creation of databases, their management, data storage and access. Examples are SQL Server, Oracle and MySQL
Embroscopy	Procedure that involve using a probe with a camera to inspect inside the uterus
Fetal Anomaly	A malformation of the fetus during its development
Fetus	An unborn baby after 8 weeks from conception to the delivery date
Gene mutation	Gene mutation is a permanent change in the DNA sequence that makes up a gene
Gene mutation	An alteration of a gene, it may have hazardous consequences on the

	affected fetus
Genetics	A topic related to hereditary and gene related issues
Gestational Age (GA)	The period from the conception date to the current date
Human Chorionic Gonadotropin (hCG),	A hormone produced by the placenta, it indicates pregnancy and a part of it; beta sub-unit is increased in affected pregnancies.
Intellisense	A feature of the .Net framework that enables the auto completion of code statements.
Klinefelter's syndrome	Abnormal gene distribution affecting men that results in an extra sex chromosome. Men with Klinefelter's syndrome are usually sterile and maybe dyslexic
Maternal Serum	Biochemical markers present in the mother's blood during pregnancy (Haddow et al). Abnormal levels of these chemicals may indicate increased risk for certain birth defects and genetic diseases
Menstrual history	The history of the menstrual cycles of the mother
Metabolic disorders	Refers to digestion disorders, it refers to the uptake, digestion and utilisation of food
MIT	Massachusetts Institute of Technology is an engineering, research institute and University in Cambridge United States of America.
Neural tube defect	Birth defect resulting in improper development of the brain or spinal cord
Patient cohorts	Patients and all those who receive the services of this system
Prenatal diagnosis	Procedures that are used definitively test for the presence of a medical condition in an embryo or a fetus
Risk Factors	Factors that increase the risk of an individual developing a disorder
Risk likelihood	The risk of a pregnancy being affected by a given disorder e.g. 1 in 100 means the probability of the pregnancy being positive is 0.01
Screening	A systematic process of identifying the possibility of an individual being affected by a disorder
Spina bifida	Malformation of the spine resulting in a spinal opening. Effects include; paralysis, nerve damage and bowel-bladder complications
Trimester	Pregnancy is divided into three trimesters. The first trimester is 0-13 weeks gestation. The second trimester is 14-26 weeks gestation.
Trimester 18 (Edward's syndrome)	A trisomy of chromosome 18; effects include mental and serious health problems
Trisomy 21 (Down syndrome)	Down syndrome is a variable combination of congenital malformations caused by Trisomy 21. It is the most commonly recognized genetic cause of mental retardation

15 Appendix C: Introduction

15.1 Project Schedule

Month >	November				December				January				February				March				April				
Week Ending >	05	12	19	26	03	10	17	24	07	14	21	28	04	11	18	25	04	11	18	25	08	15	22	29	
Deliverable/Task ↓																									
Preparation																									
Research and objectives specification																									
Requirements gathering																									
Requirements specification																									
Requirements evaluation																									
Background reading																									
Feasibility Analysis																									
Risk analysis																									
High-level design																									
Low-level design of components																									
Object design																									
User interface design																									
Database design																									
Design review and evaluation																									
Documentation planning																									
Mid term project report																									

Month >	November				December				January				February				March				April			
Week Ending >	05	12	19	26	03	10	17	24	7	14	21	28	4	11	18	25	4	11	18	25	8	15	22	29
Deliverable/Task ↓																								
Project document write-up																								
Proof reading																								
System implementation																								
Integration of components																								
Technical documentation																								
User documentation																								
Testing planning																								
Test code implementation																								
Test execution																								
Review and evaluation																								
Release packaging																								
Documentation finalisation																								
Reflection and project review																								

8 - Schedule - Gantt Chart

Key

	Completion
	Work in progress
	Exam and holiday period

16 Appendix C: Analysis

Technical specifications are typically written to describe how a project will be implemented. They are created from the functional specifications

16.1 Risk Register

Risk Register			
Risk description	Likelihood (low, medium & high)	Risk Management (risk reduction plans)	Contingency (plans for when the risk happens)
Requirements instability	medium	Try to establish main requirements as soon as possible	Discuss with client if a requested take too long to implement
Incomplete Requirements	Low (the main aims are clear)	Considerable time and effort will be set aside for eliciting, documenting and reviewing requirements	Unallocated time will be left to code the missed requirements
Unclear requirements	low	The existence of a project supervisor/contractor means clarification will be available when required	components will be created using an exploratory model to obtain clarification from user reviews
Infeasible requirements	high	The development platform limits what can be done	The requirement will be scaled down or changed to fit into the requirements
Hardware constraints (capacity limitations)	low	A feasibility study of the available technology will ensure only feasible tasks are carried out	Consideration of alternative solutions
incompatible development and target hardware	low	The application will be regularly tested on the deployment server	Consider hosting the application an alternative, compatible servers
Meeting human needs	high	Continuous prototyping and demonstration will be used to ensure mutual agreement	User review may be carried out to get more specific feedback
Unsuitable development environment	low	Background research will be carried out to familiarise with the technology	Affected units can be created on other platforms & imported into the system

Failure of the management process	Medium (Political risks)	Existence of a single person to report to makes it unlikely for a communication breakdown to happen	Plan to discuss problems with the Media Innovations project supervisor.
Lack of time to perform all testing	High	As much testing will be done during testing so that final testing does not take too long	The test plan will have to be scaled down. Only the major units would be tested
Lack of cooperation or conflict	low	A good relationship was established during the developer's placement at MI	Improve communication so that goals are understood by all stakeholders
Insufficient budget	low	The development costs are minimal	The developer is prepared to spend a bit of their own money to support the project if necessary
Project unacceptable to the University	medium	Constant dialogue will be maintained with the University supervisor to ensure such a problem is discovered early in the term	Another project will have to be chosen to meet the needs of the University
Project fails to meet legal constraints	high	During unit testing, time will be allocated for checking that the system meets the legal requirements	Corrections will have to be done to ensure the system works correctly and that data is secure

>Risk	Intellectual property conflict	
>Effects	Loss of main product, litigation etc	
Impact	Likelihood	Type
high	low	strategic
>Action	Copyrights and patents might be used to protect the system	

>Risk	Low Customer satisfaction levels	
>Effects	This can provide bad publicity, seriously damage the business's reputation in the market and contribute to losing customers.	
Impact	Likelihood	Type
Moderate	Moderate	Reputation
>Action	The system must be highly usable and efficient	

>Rationale	Good usability and appearance will improve customer satisfaction
------------	--

>Risk	Web sever failure	
>Effects	The system would not be accessible	
Impact	Likelihood	Type
High	Low	Technical
>Action	The system can be hosted on a secondary server during a breakdown	
>Rationale	If the main server fails then the website would still be accessible	

>Risk	Network failure	
>Effects	Anything stored on the physical server at offices will be inaccessible, and customer service would suffer as staff will not be able to run website	
Impact	Likelihood	Type
Moderate-High	Moderate	Technical
>Action	Backup all server and workstation data in a remote location. Add ability to use a workstation away from the network as well as within it to enable the site to be managed.	
>Rationale for >evaluation	Backing up all data will enable remote access and the website will be managed as normal. Customers will also be able to have access to the entire database as opposed to just what is stored on the web server.	

>Risk	Failure to follow future usability and pricing trends	
>Effects	Customers may turn elsewhere if other services are more accessible and user friendly. Will lose business if opportunities are not taken.	
Impact	Likelihood	Type
Medium	Low	Operational
>Action	Specify job tasks and objectives when hiring staff - commercial advisor and support will need to stay in touch with changes to such businesses	
>Rationale for >evaluation	Keeping ahead of the times and providing fresh changes/updates to the website and competitive prices will keep customers interested and ensure their business continues to come our way.	

>Risk	Software failure	
>Effects	Poor service, low sales, customer dissatisfaction	
Impact	Likelihood	Type

Medium	high	Operational
>Action	Intensive testing and evaluation must be done before system installation. Software audits must be taken regularly.	
>Rationale >evaluation	for	Such risks are expected to occur and if monitored correctly, quick response will ensure that few customers are affected

>Risk	Rising competition	
>Effects	Low sales, funds and low return on investment	
Impact	Likelihood	Type
medium	medium	Operational
>Action	Counter measures to combat competition must be implemented	
>Rationale >evaluation	for	Competition will always exist in any successful industry and companies must keep ahead to stay alive

>Risk	Lack of maintenance	
>Effects	Degradation of service quality, low sales and system failure	
Impact	Likelihood	Type
medium	medium	Operational
>Action	Maintenance registers must be filled at regular cycles and administration must ensure that regular system servicing is done.	

>Risk	Data loss	
>Effects	Widespread, loss of service	
Impact	Likelihood	Type
High	High	operational
>Action	Regular backups are compulsory. And backup copies are stored offsite in very secure surroundings.	
>Rationale >evaluation	for	If backups are done, minimal data loss must be incurred and only a few customers will be affected while restoration is done

>Risk	System security failure	
>Effects	Disruption of service and disclosure of confidential information	
Impact	Likelihood	Type
medium	medium	operational
>Action	The security system of the server will protect stored information, we aim to develop a security system to protect data in transit	
>Rationale	The information an attacker will be able to gain is not of much use so	

	it is not an urgent priority to protect it.
--	---

16.2 MI Software Engineering Guidelines

16.2.1 Project Organisation

Development documentation; as a minimum, always have the following (separate) documents to hand, and keep them up to date:

- Specification: formal, if it exists, otherwise informal (as complete as possible, and updated as required).
- High-level list of additions, modifications and corrections to make; add to this list *immediately* when an item comes to your attention.
- Current task(s): expanded information.
- Information required / decisions to be made.
- Design decisions: decisions taken about functionality and implementation, with brief reasons.
- Essential information for project handover: development packages used, components and their locations, version information, passwords, lists of finished and unfinished tasks.

Product documentation:

- User manual: use AMIE Beta manual as template.
- CD label and cover: use Template\CDLabel as template.

Code storage and version control:

- Folder structure: C:\MIDev\\Code (or Debug etc.) contains current version. Released versions are stored on MI-SRV.
- In general, don't keep previous versions. If there is a particular reason for keeping a version then ensure that folder/file names make it clear that the version is not current.
- Version numbering (Major:Minor:Revision):
 - Start project with 0.0.0.
 - First release version to customer should be 1.0.0. If immediate revisions are necessary, increment Minor.
 - Increment Major for upgrades only.

16.2.2 Coding

Always check whether we have existing code examples or components before writing your own (look in MI-SRV.MIDev\Utilities).

Development tools/packages:

- Databases: Microsoft SQL Server 2000, Access 2000.
- ActiveX components: Visual Basic 6.
- Win32 clients: Visual Basic 6.
- Web clients: Visual InterDev 6/Visual Basic .Net
- Front ends for database configuration: Access 2000 (.adp), Visual Basic 6/Visual Basic .Net

Program structure:

- Think client/server; ensure that it will be easy to encapsulate the functionality in a dll etc.
- No 'server' functionality in forms.
- Minimise repeated code: use modules with shared subs, and classes.
- Hardcoded values: use constants.
- Think DCOM: client, server and database may be on different PCs. (Pass values ByVal etc.)

Providing objects (classes):

- An object should generally contain data and functionality that together represent some entity (real-world object, related group of tasks etc.).
- Public properties should be general to the class. Input data for a particular sub should be passed as parameters; conversely, data required by more than one sub should be public properties.
- If your class contains methods and functions only (no properties) then it should probably be a standard module (unless, obviously, it is part of an ActiveX interface).

- If your class is just a data store (properties only) then it is probably unnecessary.
- For public properties use module variables and property Set, Get, Let, not public variables.
- If a group of input data must be set before calling any methods (general class data, not specific to the method), then consider using module variables instead of public properties, and setting them all in a single data-passing method (DCOM). [elipse.Markers.CovarInfo](#)
- Don't return results (functions or public properties) as user-defined types (they can't be accessed by VBScript).

Using objects:

- Declare object variable as specific class type (not 'As Object') wherever possible (early binding).

Providing collections:

- Create your own specific collection class instead of using the generic Collection class. [elipse.Markers](#)

ActiveX dll interface:

- Where useful to client programmer (and sparingly), provide component 'global variable'. (Declare as public in standard module, read/write using property Set/Let/Get in GlobalData global multi-use class used for this purpose only.)

Database connection: for each database have <DatabaseName>Connect class that just handles database operations (reads, writes etc). Use ADO for any type of database. [elipse.elModelConnect](#)

- If connection information shouldn't change during running of client program, receive it in component 'global variable(s)'.
- In Class Initialize: set module variable to open ADODB.Connection, using connection information.
- In Class Terminate, set this variable to Nothing.
- For database read, sub returns a recordset.
- For database write, data supplied as recordset.

Code layout:

- Use Option Explicit and always declare variable type. Declare private objects together at beginning of proc. Declare global variables together in a single modDeclarations module.
- Commenting: use comment blocks at start of object/section etc., and in-line comments. [Res.asp \(elipseWeb_v2\)](#), include:
 - At start of each module and proc: what it does.
 - Each variable.
 - Code section for a particular task: at start, what it does.
 - Why something is done in a particular way, if not obvious.
 - All database fields (unless completely obvious) and component type libraries.
- Indent to show grouping/nesting (prefer 2 or 3 spaces).
- In general, don't have code lines much longer than screen width (use _).
- Use If, If...Elseif, With only where this makes code clearer.

Naming conventions:

- *Never* use the same name for different entities. For entities representing the same quantity (e.g. property, module variable representing it, parameter, procedure variable representing it, variables with overlapping scope) use variations on the root name.
- Ensure the name conveys sufficient information, and don't abbreviate unnecessarily, e.g. dtmTransactionDate, not dtmTrnsDte.
- strWordStringsLikeThis, not strWord_Strings_Like_This or strwordstringslikethis.
- For module variables, use m<SensibleName>, e.g. mTransactionDate.
- For global variables, use g<SensibleName>, e.g. gLanguage.
- See Appendix for prefixes.

Using 3rd party controls: set properties in code rather than in settings pages (easier to copy and modify).

Concatenating strings: always use & (*not* +).

Error handling in all procs (unless they should be handled by the calling proc, or in simple Property Get etc). Method: always use Handler at end of proc. [elipse](#)

16.2.3 User Interface

Overall organisation and navigation:

- MDI main (startup form) with ActiveListBar at left and Menu Bar at top.
- MDI size equivalent to full-screen at resolution 800x600 (allowing for Windows task bar).
- Child forms not sizable. [DemoBatch](#)
- Blank (or background) form when program group changed.

Think ease-of-use:

- ListBar organisation: logical order of groups and items, items in a group should be in some way comparable.
- Don't duplicate ListBar functionality in menu bar.
- Helpfulness of labels.
- Tool tips.
- Tab order.

Data entry fields:

- Date fields: always display as Medium Date.
- Lists: use combo boxes.
- All fields must be validated for data type, range etc., preferably on leaving field

On any change in input (on leaving field):

- Wipe all results and graphs
- *Don't* process the new values: wait for user instruction (click on Calculate etc.)

Consistency checks:

- Numerical fields: decimal precision
- Language/terminology/abbreviations
- Fonts
- Colours
- Control size and alignment (especially textboxes).

16.2.4 Locale Independence

All code must work correctly in any Windows locale

General

- Program components must receive parameters/properties in the correct type: dates, double, integer, etc and NOT variant.
- Don't pass a date strings directly to a recordset date field, instead convert it using CDate first and pass it using the MIUtils.sqlDate function
- Use formatting to convert a date/number into a string when displaying dates and numbers
- Dates and numbers must be stored in variables of the correct type.
- Use validation (see the user interface section) to prevent invalid values being passed to a conversion function, the source value must be valid for the destination data type or an error occurs.
- Most VB.Net formatting and parsing methods have the ability to dynamically change based on the current culture, (Always make sure a method has such capabilities before using it and if it doesn't, you will need to send it Culture and formatting information to ensure the correct formats are used)

Dates

- Always use named formats when formatting dates for display (Use the MIUtils DisplayDate function whenever possible)

- Avoid storing dates as strings in your code
- Enter dates in code in the format #month/day/year# to ensure that the date will be interpreted correctly in any system locale.
 -
- VB.Net
 - The System.Globalization namespace contains classes that define culture related information, the calendars in use and the format patterns for dates and numbers
 - The .NET Framework also defines standard format providers and specifiers (e.g. 'd' - for ShortDatePattern) for formatting and parsing dates - see the DateFormatInfo class
 - Input From:
 - User
 - Use Date.Parse, (inverse of formatting) to create a date from a date string
 - Always store the resulting date (from parsing) in a date variable of the correct type, Date, DateTime etc
 - Passing To:
 - SQL Server 2000: ###
 - Access 2000: ###
 - Conversion
 - Use the IsDate function to determine if a value can be converted to a date before converting
 - Display:
 - Format using the overloaded ToString method when displaying dates, e.g. Date.ToString (Date, Format Provider)

Numbers

- When displaying, don't use explicit formatting e.g. Format (3, "\$##0.00"),
- Instead use the format function e.g. Format (3, "0.00") without any literal characters: characters are displayed exactly as typed
- VB.Net
 - Use standard numeric format strings when formatting common numeric types for display. They take the form Axx where A is the format specifier (e.g. D for decimal), and xx, an optional integer, is the precision specifier (e.g. Number of decimal places). E.g Double.ToString (Cdbl(17.3344), "D3")
 - VB.Net does not convert strings to numbers for you, so make sure you convert numerical strings to numbers first before formatting, thus:
 - Format ("1.234", "#.#") ' Displays "#.#".
 - Format (CSng ("1.234"), "#.#") ' Displays "1.234"
 - Input From:
 - User and Database:
 - When reading in a numeric string, use the Parse method of the target variable (e.g. MyInt.Parse (myNumString, [Format Provider], [Named Format], [Precision]) to convert the string before storing it in a variable of the correct data type
 - Passing To:
 - Other methods
 - SQL Server 2000:
 - Access 2000:
 - Conversion:
 - Use local aware conversion functions to convert strings to numbers and numbers to other numbers (CSng, CInt etc)
 - Before conversion, make sure the expression passed to the conversion function is within the range of the datatype to which it is being converted; otherwise you will get an error or unwanted truncation
 - Display:
 - Use the ToString method and the correct NumberFormatInfo's FormatSpecifiers e.g. MyDouble.ToString ("dx") where d is the format specifier and x is the precision specifier (e.g. no of decimal places)

- Web applications
 - Set the current culture (system locale in VB.Net) of every user's thread on form load using the Culture information supplied by the Browser object.
 - Follow the localisation instructions given for VB.Net dates and numbers.

1. Variable Scope Prefixes

Scope	Prefix	Example
Global	g	gstrUserName
Module-level	m	mblnCalcInProgress
Local to procedure	None	dblVelocity

2. Suggested Prefixes for Variables

Data type	Prefix	Example
Boolean	bln	blnFound
Byte	byt	bytRasterData
Collection object	col	colWidgets
Currency	cur	curRevenue
Date (Time)	dtm	dtmStart
Double	dbl	dblTolerance
Error	err	errOrderNum
Integer	int	intQuantity
Long	lng	lngDistance
Object	obj	objCurrent
Single	snq	snqAverage
String	str	strFName
User-defined type	udt	udtEmployee
Variant	vnt	vntChecksum

3. Suggested Prefixes for Controls

Control type	Prefix	Example
3D Panel	pnl	pnlGroup
ADO Data	ado	adoBiblio
Animated button	ani	aniMailBox
Check box	chk	chkReadOnly
Combo box, drop-down list box	cmb	cmbEnglish
Command button	cmd	cmdExit
Control (specific type unknown)	ctl	ctlCurrent
Data	dat	datBiblio
Data combo	dbc	dbcAuthor
Data grid	dqd	dqdTitles

Data list	dbl	dblPublisher
Data repeater	drp	drpLocation
Date picker	dtp	dtpPublished
Directory list box	dir	dirSource
Drive list box	drv	drvTarget
File list box	fil	filSource
Flat scroll bar	fsb	fsbMove
Form	frm	frmEntry
Frame	fra	fraLanguage
Gauge	qau	qauStatus
Graph	qra	qraRevenue
Grid	qrd	qrdPrices
Hierarchical flexgrid	flex	flexOrders
Horizontal scroll bar	hsb	hsbVolume
Image	ima	imalcon
Image combo	imacbo	imacboProduct
ImageList	ils	ilsAllIcons
Label	lbl	lblHelpMessage
Line	lin	linVertical
List box	lst	lstPolicyCodes
ListView	lvw	lvwHeadings
MAPI message	mpm	mpmSentMessage
MAPI session	mps	mpsSession
MCI	mci	mciVideo
Menu	mnu	mnuFileOpen
Month view	mvw	mvwPeriod
MS Chart	ch	chSalesbyRegion
MS Flex grid	msq	msqClients
MS Tab	mst	mstFirst
OLE container	ole	oleWorksheet
Option button	opt	optGender
Picture box	pic	picVGA
Picture clip	clp	clpToolbar
ProgressBar	prq	prqLoadFile
Remote Data	rd	rdTitles
RichTextBox	rtf	rtfReport
TabStrip	tab	tabOptions
Text box	txt	txtLastName

Timer	tmr	tmrAlarm
Toolbar	tlb	tlbActions
UpDown	upd	updDirection
Vertical scroll bar	vsb	vsbRate

For controls not listed above, you should try to standardize on a unique two or three character prefix for consistency. Use more than three characters only if needed for clarity.

For derived or modified controls, for example, extend the prefixes above so that there is no confusion over which control is really being used. For third-party controls, a lower-case abbreviation for the manufacturer could be added to the prefix. For example, a control instance created from the Visual Basic Professional 3D frame could use a prefix of fra3d.

4. Suggested Prefixes for ADO

Obect	Prefix	Example
Command	cmd	cmdRecordSource
Connection	con	conPatientsDb
Database	db	dbAccounts
Field	fld	fldAddress
Parameter	prm	prmJobCode
Recordset	rst	rstForecast

5. Suggested Prefixes for Menus

Menu caption sequence	Menu handler name
File Open	mnuFileOpen
File Send Email	mnuFileSendEmail
File Send Fax	mnuFileSendFax
Format Character	mnuFormatCharacter
Help Contents	mnuHelpContents

6. User-Defined Types

In a large project with many user-defined types, it is often useful to give each such type a three-character prefix of its own. If these prefixes begin with "u," they will still be easy to recognize quickly when you are working with a user-defined type. For example, "ucli" could be used as the prefix for variables of a user-defined Client type.

17 Appendix D: System Requirements

17.1 Interviews

Interview Information	
Interview Title	Project Conception Interview
Interviewee(s)	Carol Wilson (MI Project supervisor), Daniel Taylor
Date of interview	17 October 2005
Location	Media Innovations meeting room
Related Documents	None
Interview Questions and Answers	
How did you learn about the need for this product?	
Could you explain the problem domain of the project and how the proposed system would be of use?	
What will be the main purposes of the system?	
Who will be the user of this system?	
What are the benefits that you or the users might expect from this system?	
When do you want the system do be delivered?	
What are the constraints of the system from a business point of view?	
In order of priority, what are the main things you require of this new system?	
How is the system going to be used?	

How would you like the system to be developed in order to ensure minimum disturbance to you whilst maximising your involvement?
What are the main or likely rival systems to this product?
What are the likely legal requirements that this system is required to satisfy?
How will the system be deployed?
What security mechanisms would the system be expected to provide?
Follow up questions and additional answers
What security features do you normally provide for your software?
What contractual requirements is the developer required to abide to?
What technologies do you use for developing your software?
What browser technology is mostly used in the health industry?
Conclusion
Plans for future meetings

17.2 Functional Requirements

ID	Calculation (RC) : Requirement
RC_1	The system shall provide a data input form that enables users to input biochemical markers, pregnancy dating and disorder information
RC_2	The system shall validate user input (numerical and dates) according to user locales formats
RC_3	The system shall give appropriate error messages in order to enable users to correct errors before submitting a form
RC_4	The system shall provide a results form for displaying the calculated pregnancy dating results, marker moms and disorder results
RC_5	The system shall provide a data import form that will enable users to upload files that contain records for result calculation
RC_6	The system shall provide a data export function that will enable users to download calculated results that will be stored in an access file or text file
RC_7	The system must calculate results based on the settings (equations and parameters) stored in a settings database: the settings database must be designed to meet the data requirements of the pregnancy dating, markers and disorders calculations. The equations and parameters required to perform these calculations will be specified in due course.
RC_8	The system shall provide a file download function that will enable users to download informative leaflets that Media Innovations publish
RC_9	The system must display informative message boxes informing the user of the progress of time consuming operations, errors and system malfunction
RC_10	The system shall validate input data for clinical validity before calculation
RC_11	The system shall provide a response time of less than 30 seconds on average
RC_12	The system must provide the functionality to automatically generate the patient report in memory using a template PDF form and the calculated results
RC_13	The system shall only perform a post back when the user initiates a calculation
RC_14	The system shall generate friendly and useful error messages as explained in the usability guidelines
RC_15	The system shall hide the database access functionality and code so that it is not specifically stated in code
RC_16	The system shall cache all static objects such as images and non changing parts of the user interface
RC_17	The system shall follow the accessibility and usability guidelines of MIT and Nielsen respectively

RC_18	The system must be tested for implementation errors and must be validated to beta test version before submission for clinical validation by external validators
RC_19	The system shall validate weight values so that they are within acceptable ranges
RC_20	The system shall validate the patient date of birth so that maternal age values so that they are within acceptable ranges
RC_21	The system shall validate marker test results values to ensure that they are within the acceptable range of values.
RC_22	The system shall not store cookies on the users machine since most windows system have cookies disabled
RC_23	The system must be able to accept uploaded files, import them to a database and format data correctly and calculate results
RC_24	The system shall be able to interface correctly with the existing dynamic link library and when it calls methods of that dll, it must do so using the correct number and type of parameters
RC_25	The system shall calculate marker mom results from the input marker values
RC_26	The system shall calculate pregnancy dating results from given pregnancy dating clinical testing results
RC_27	The system shall provide an export function that will allow users to print results and save them in some easy to retrieve format
RC_28	The system shall provide a web form navigation function on every page in the form of a control that is reused on all web forms of the application.
RC_29	The system shall provide a date entry function for entering dates into date fields. This function must also validate dates before accepting user input
RC_30	The system shall display dates and numbers in the same format as the users locale
RC_31	The system shall display numbers to a precision of 3 decimal places
RC_32	Truncation and rounding of values can only be done at the end of a calculation, the system must not truncate values during a calculation
RC_33	All operations that can be executed on the client side must be scripted for client side execution whenever possible to minimise post back times
	Security (SE)
SE_1	The system shall provide access to authorised users
SE_2	The system shall protect/encrypt sensitive information while it is being transferred over the internet
SE_3	The system shall prevent access to unauthorised users
SE_4	The system must provide a user identification functionality without having to store cookies on the users machine
SE_5	External users must not be able to access internal/server data through the system

SE_6	The system shall abide to the data protection act of 1994
SE_7	The system shall prevent access to users once they have logged out or once their session has expired until identification has been done.
SE_8	The system must not allow users to view other users data
SE_9	The system should be protected from security risk as much as possible
SE_10	The system must be capable of identifying users using a customer identity, username and password
SE_11	The system must be accessible to registered clinicians only
SE_12	The system must display a standard error message when a user enters wrong login information so that the system does not give away information unnecessarily
	User interface (UI)
UI_1	The system shall provide a data input page that accepts information required to calculate pregnancy dating, markers and disorders results
UI_2	The system shall provide a control for initiating calculations, display help and results report
UI_3	The system must provide an internationalised, client based date picker that enables users to enter to select dates in an efficient manner
UI_4	The system shall provide a panel where users can enter additional information related to a patient such as : weight, diabetes, smoking, and all risk factors
UI_5	The system shall display the names of markers and their units of a markers panel to enable users to enter valid values
UI_6	The system shall enable the user to specify the disorders for which they want results to be calculated for
UI_7	The system provide a default, security page that will be presented to the user requesting their authentication details
UI_8	The system display a “1 in” before all risk values
UI_9	The system shall display gestation in weeks and days and maternal age in years and months
UI_10	The system shall provide a link to the adobe website to enable users to download acrobat reader for use when viewing patient reports
UI_11	After calculating results, the system must automatically display the calculated results
UI_12	The system shall display all dates in the DD MMM YYYY format, the short month name must be in the current user’s language format
UI_13	The system’s date picker must display full month names in the users language
UI_14	The user interface must be consistent with other software products of Media Innovations

UI_15	The user interface must be labelled using correct and easy to understand English
UI_16	Controls on the user interface must be laid out in a clean and well formatted manner
UI_17	Numbers must be formatted to two decimal places before display
ID	Reporting (RE)
RE_1	The patient report must display user details to enable identification of the physician who calculated the results
RE_2	The patient report must show all the input data entered on the calculation form and must also show the results calculation date
RE_3	The patient report must have a disclaimer with a message that tells the user that the responsibility for entering valid data lies with them
RE_4	The patient report must show the version number of the system
RE_5	The system must display results on all reporting forms and documents in a consistent manner
RE_6	The system must produce the report in PDF and HTML formats
RE_7	The PDF patient report must be printable on most printers, it must have the default A4 page dimensions
ID	Help: Requirement (HP)
HP_1	The system shall provide a help system that explains how to use the system
HP_2	The system shall provide a context sensitive help on the major tasks: results calculating, reporting, validation, pregnancy dating markers, disorders and security if possible
HP_3	The system shall provide a form for viewing help.
	General Requirements
HP_6	The system implementation of the system must be documented in an implementation document explaining the decisions made during development and their rationale
HP_7	The system must be implemented according to media Innovations guideline, it must be written in a clear, consistent and well documented manner
HP_8	The developer shall write maintenance documents for the system. This document must explain the systems functions, their locations and how the system works
HP_9	The system must be usable on Explorer browsers since they are the default browsers in the health system
HP_10	The system must use Media Innovations Ltd logos and copyrighted graphics on the user interface

18 Appendix E: Design

18.1 Database Normalisation

Normalisation is an iterative process that eliminates redundancy, organises data in an organised manner and reduces the potential for anomalies when records are updated. Edgar Codd established originally established 3 normal forms. The first normal form states that attributes can only store a single value. A table is in second normal form if all non-key attributes are fully dependant on the primary key. A table is in third normal form if it is in second normal form and there exist no non-trivial dependencies between non-key attributes. The schema below shows the database in non-normalised form and in third normal form.

18.1.1 Un-normalised form

- User (**Customer Id**, Username, Password)
- Pregnancy Dating (**Id**, **Name**, Gestation Equation, Min Measurement, Max Measurement)
- Markers (**Id**, **Name**, **Disorder**, Min Value, Max Value, Gestation Range, Markers List of between 1 and 5 markers, Distribution Equation, Weight Equation, Normalisation Equation)
- Disorder (**Id**, **Name**, Age Risk Equation, Final Risk Equation, List of Risk Factors)

18.1.2 Third normal form

- User (**Customer Id**, Username, Password)
- Pregnancy Dating (**Id**, **Name**, Gestation Equation, Min Measurement, Max Measurement)
- Markers (**Id**, **Name**, **Disorder**, Min Value, Max Value, Gestation Range)
- Disorder (**Id**, **Name**, Age Risk Equation, Final Risk Equation)
- Marker Set (**Id**, **Disorder**, **Markers List** of between 1 and 5 markers)
- Distribution Parameters (**Id**, **Marker**, Distribution Equation, Weight Equation)
- Normalisation Parameters (**Id**, **Marker**, Normalisation Equation)
- Risk Factors (**Id**, **Name**, **Disorder**, Factor)

18.2 Usability Guidelines

Massachusetts Institute of Technology IS&T Usability Guidelines

Courtesy of MIT (<http://web.mit.edu/is/web/reference/guidelines/usability.html>)

Navigation	
Current location within the site is shown clearly.	✓
Link to the site's main page is clearly identified.	na
Major/important parts of the site are directly accessible from the main page.	✓
Site map is provided for a large, complex site.	na
Easy to use Search function is provided, as needed.	na
Functionality	
Site accommodates novice to expert users.	✓
Functions are clearly labelled.	✓
Essential functions are available without leaving the site.	✓
Plug-ins are used only if they add value.	✓
User control	
Site reflects user's workflow.	✓
User can cancel any operation.	✓
Clear exit point is provided on every page.	x
Per-page size is less than 50K, to accommodate slow connections.	✓
All appropriate browsers are supported.	✓
Language and content	
Important information and tasks are given prominence.	✓
Information of low relevance or rarely used information is not included.	✓
Related information or tasks are grouped:	✓
- on the same page or menu.	✓
- in the same area within a page.	✓
Language is simple, without jargon.	✓
Paragraphs are brief.	✓
Links are concise, expressive, and visible--not buried in text.	✓
Terms are defined.	✓
Online help and user guides	
Site is designed to require minimal help and instructions.	✓
Help and instructions, if needed, are easily accessible.	✓

System and user feedback	
It is always clear what is happening on the site -- visual hints, etc.	✓
Users can receive email feedback if necessary.	x
Users can give feedback via email or a feedback form.	x
Confirmation screen is provided for form submittal.	✓
All system feedback is timely.	✓
Users are informed if a plug-in or browser version is required and a link is provided to the necessary plug-in or browser site.	✓
Each page includes a "last updated" date.	✓
Consistency	
The same word or phrase is used consistently to describe an item.	✓
Link reflects the title of the page to which it refers.	✓
Browser page title is meaningful and reflects main page heading.	✓
Error prevention and correction	
Users can rely on recognition, not memory, for successful use of the site.	✓
Site tolerates a reasonable variety of user actions.	✓
Site provides concise instructions for user actions, including entry format.	✓
Error messages are visible, not hidden	✓
Error messages are in plain language.	✓
Error messages describe actions to remedy a problem.	✓
Error messages provide a clear exit point.	✓
Error messages provide contact details for assistance.	✓
Architectural and visual clarity	
Site is organized from the user's perspective.	✓
Site is easy to scan for organization and meaning.	✓
Site design and layout is straightforward and concise.	✓
Site design and layout are redundant only when required for user productivity.	✓
White space is sufficient; pages are not too dense.	✓
Unnecessary animation is avoided.	✓
Colors used for visited and unvisited links are easily seen and understood.	x
Bold and italic text is used sparingly.	✓

18.3 Accessibility Guidelines

W3C Checklist of Checkpoints for Web Content Accessibility Guidelines 1.0

In General (Priority 1)	Yes	No	N/A
-------------------------	-----	----	-----

1.1 Provide a text equivalent for every non-text element (e.g., via "alt", "longdesc", or in element content)			
2.1 Ensure that all information conveyed with color is also available without color, for example from context or markup.			
4.1 Clearly identify changes in the natural language of a document's text and any text equivalents (e.g., captions).			
6.1 Organize documents so they may be read without style sheets. For example, when an HTML document is rendered without associated style sheets, it must still be possible to read the document.			
6.2 Ensure that equivalents for dynamic content are updated when the dynamic content changes.			
7.1 Until user agents allow users to control flickering, avoid causing the screen to flicker.			
14.1 Use the clearest and simplest language appropriate for a site's content.			
And if you use images and image maps (Priority 1)	Yes	No	N/A
1.2 Provide redundant text links for each active region of a server-side image map.			
9.1 Provide client-side image maps instead of server-side image maps except where the regions cannot be defined with an available geometric shape.			
And if you use applets and scripts (Priority 1)	Yes	No	N/A
6.3 Ensure that pages are usable when scripts, applets, or other programmatic objects are turned off or not supported. If this is not possible, provide equivalent information on an alternative accessible page.			
And if all else fails (Priority 1)	Yes	No	N/A
11.4 If, after best efforts, you cannot create an accessible page, provide a link to an alternative page that uses W3C technologies, is accessible, has equivalent information (or functionality), and is updated as often as the inaccessible (original) page.			

Priority 2 Checkpoints

In General (Priority 2)	Yes	No	N/A
2.2 Ensure that foreground and background color combinations provide sufficient contrast when viewed by someone having color deficits			
3.1 When an appropriate markup language exists, use markup rather than images to convey information.			
3.2 Create documents that validate to published formal grammars.			
3.3 Use style sheets to control layout and presentation.			
3.4 Use relative rather than absolute units in markup language attribute values and style sheet property values.			
3.5 Use header elements to convey document structure and use them according to specification.			
3.6 Mark up lists and list items properly.			
3.7 Mark up quotations. Do not use quotation markup for formatting effects such as indentation.			
6.5 Ensure that dynamic content is accessible or provide an alternative presentation or page.			
7.2 Until user agents allow users to control blinking, avoid causing content to blink (i.e., change presentation at a regular rate, such as turning on and off).			
7.4 Until user agents provide the ability to stop the refresh, do not create periodically auto-refreshing pages.			
7.5 Until user agents provide the ability to stop auto-redirect, don't use mark-up to redirect pages. Instead, configure the server to perform redirects.			
10.1 Until user agents allow users to turn off spawned windows, do not cause pop-ups or other windows to appear and do not change the current window without informing the user.			
11.1 Use W3C technologies when they are available and appropriate for a task and use the latest versions when supported.			
11.2 Avoid deprecated features of W3C technologies.			
12.3 Divide large blocks of information into more manageable groups where natural and appropriate.			
13.1 Clearly identify the target of each link.			
13.2 Provide metadata to add semantic information to pages and sites.			
13.4 Use navigation mechanisms in a consistent manner.			

19 Appendix F: Implementation

19.1 Database

19.2 Security

```
1  /*
2  * Javascript for encrypting user login details, s is the string
3  * and k is the hash key
4  */
5  var cipher_block_size=64, encoding_buffer=1024;
6  function b0f(c) {
7      return c < 16 ? '0' + c.toString(16) : c.toString(16);
8  }
9  function bff(c) {
10     return parseInt(c, 16);
11 }
12 function salt(s) {
13     var n = 0;
14     for (var i=0; i<s.length; i++) n += i&s.charCodeAt(i);
15     return b0f(n%256);
16 }
17 function encrypt(s,k) {
18     if (s.length > cipher_block_size) {
19         var m=parseInt(s.length/cipher_block_size), t=Math.round(m/2)
20             *cipher_block_size;
21         return encrypt(s.substr(0, t), k) + encrypt(s.substr(t), k);
22     }
23     var r=parseInt(Math.random()*256), o=b0f(r), i;
24     for (i=0; i<s.length; i++) {
25         o += b0f(s.charCodeAt(i)^r^k.charCodeAt(i%k.length));
26     }
27     return o;
28 }
29 function encipher(f) {
30     var s=f.stream.value;
31     var k=salt(f.key.value), r=k, b=encoding_buffer;
32     var p=0, n=Math.floor(s.length/b)+1;
33     while (n > p++) {
34         self.status = 'Encrypting block ' + p + '/' + n;
35         r += encrypt(s.substr((p-1)*b, b), f.key.value);
36     }
37     f.stream.value = r;
38     self.status = 'Done';
39 }
```

19.3 Validation

```
1 function validateNumbers(identity, separators) {
2   //this method formats the entered string by removing characters
3   //and extra commas .i.e separator = separ
4
5   var theform = document.Form1;//the patients page
6   var Control=document.getElementById(identity+"") //edited textbox
7   var validchars = "0123456789"+ separators; //accepted values
8   var thousandSep=separators.charAt(0); //client's thousands separ
9   var decimalSep=separators.charAt(1); //client's decimal separ
10  var EnteredString=Control.value; //the entered numerical string
11  var validNumber=true; //boolean var for validating user input
12  var numOfDecSep=0; //number of decimal separators in string
13
14  if (EnteredString == "") {
15    return;
16  } //end if
17  EnteredString = EnteredString.replace(/[" "]/g, "");
18  //remove invalid characters from the entered input
19  for (var i=0; i < EnteredString.length; i++) {
20    var letter = EnteredString.charAt(i).toLowerCase();
21    //the char at position i
22    if (letter == thousandSep) {
23      EnteredString = EnteredString.replace(letter, "");
24    } //end if
25    if (validchars.indexOf(letter) == -1) {
26      validNumber=false;
27    } //end if
28    if (letter == decimalSep) {
29      numOfDecSep++;
30    } //end if
31    if (numOfDecSep > 1) {
32      validNumber=false;
33    } //end if
34  } //end for
35
36  if (validNumber==false) {
37    Control.focus();
38    alert("Value must be a valid number >=0");
39    return;
40  } else {
41    EnteredString = EnteredString.replace(decimalSep, ".");
42    EnteredString = parseFloat(EnteredString)+"";
43    Control.value = EnteredString.replace(".", decimalSep);
44  } //end if
45 } //end function validate numbers
```

Reporting System

The reporting system uses a fillable PDF form to create a patient report. Figure 1 shows the form in design view. The program reads in this file into memory and fills the fields of the template with the current patient's results

Eclipse PrenatalLink Screening Calculator					
Patient		Report		Report Date	
Patient Information					
Surname	txtSurname	Dating Type	2006	txtDatingGestOrValue	
Forename	txtForename	Fetuses		txtFetuses	
ID	txtID	Weight		txtWeight	
DOB	txtDOB	Other Details		txtOtherDetails	
MAEDD	txtMAEDD				
Food Type					
MoM Heading					
MoM		Gestation		Twin MoM	
Marker	txtMoM1	Gest	txtGest1	Twin MoM	txtTwinGest1
Marker	txtMoM2	Gest	txtGest2	Twin MoM	txtTwinGest2
Marker	txtMoM3	Gest	txtGest3	Twin MoM	txtTwinGest3
Marker	txtMoM4	Gest	txtGest4	Twin MoM	txtTwinGest4
Marker	txtMoM5	Gest	txtGest5	Twin MoM	txtTwinGest5
Marker	txtMoM6	Gest	txtGest6	Twin MoM	txtTwinGest6
Marker	txtMoM7	Gest	txtGest7	Twin MoM	txtTwinGest7
Results Header					
Age Risk		Final Risk		Twin Results	
Disorder	txtPriorRisk1	PostRisk1	txtResult1	Twin Final Ri	txtTwinResult1
Disorder	txtPriorRisk2	PostRisk2	txtResult2	Twin Final Ri	txtTwinResult2
Disorder	txtPriorRisk3	PostRisk3	txtResult3	Twin Final Ri	txtTwinResult3
User		Customer ID		Prenatal Version	
Username	txtUsername	Customer ID	txtCustomerId	Prenatal Link Version	txtPrenatalLinkVersion
Eclipse Version		Disclaimer			
This report was produced by Eclipse PrenatalLink v0.1, which is an unvalidated Beta Test version.					
THESE RESULTS MUST NOT BE USED FOR CLINICAL PURPOSES.					

Figure 16 - Template Patient Report (PDF Form)

The generated report is then downloaded to the client's browser where they can open or save the PDF file. Figure 2 shows an example patient report.

Elipse PrenatalLink Screening Calculator					
Patient Report: 24 Apr 2006					
Patient Information					
Surname:	Screening	Gestation at 24 Apr 2006:	11 weeks 3 days		
Forename:	Prenatal	Fetuses:	2		
Id:	DD	Weight:	77 Kg, 8 Kg		
DOB:	01 Jan 1983	Other Details:	Diabetes, Smoking, Previous T21, Previous T18		
MAEDD:	23 years 10 months				
Markers					
	MoM/Delta	Gestation	Twin/2nd Measurement MoM/Delta	Gestation	
NT:	0.89	11 weeks 3 days	1.66	11 weeks 5 days	
AFP:	1.87	11 weeks 3 days	1.45	11 weeks 6 days	
HCGb:	0.34	11 weeks 3 days	0.36	12 weeks 0 days	
PAPP-A:	0.81	11 weeks 6 days	1.17	11 weeks 1 day	
UE3:	22.70	12 weeks 0 days	27.29	11 weeks 5 days	
Results					
	Age Risk	Final Risk	Result	Twin 2 Final Risk	Twin 2 Result
T21:	1 in 125	1 in 2483	Screen Negative	1 in 172	Screen Positive
Username:	leeds		Customer Id:	ltest	
Elipse Version:	3.0.0.8		PrenatalLink Version:	0.1 (Beta Test)	
Disclaimer					
This report was produced by Elipse PrenatalLink v0.1, which is an unvalidated Beta Test version.					
THESE RESULTS MUST NOT BE USED FOR CLINICAL PURPOSES.					

Figure 17 - Sample Patient Report


20 Appendix E: Testing

20.1 Test Plan

Project no. and name :	Developer: Courage Chibiya
PrenatalLink	Date: 20 April 2006
Document name :	Author: Courage Chibiya
PrenatalLink Test Plan	Revision#: 4
	Date : 20 April 2006

This document describes the test cases created for the PrenatalLink application to test the system up to the beta test version. This version would be evaluated and subjected to further testing and validation by Media Innovations and eventually by external validators. This test plan is based on the test requirements in section 16.1 above.

20.1.1 Definitions and acronyms

Term / Acronym	Description
PI	Program Input (PI), the test executed on the program
PO	Program Output (PO), the out put produced by the program when the test was executed
<i>Calculate</i>	A button on a page of the system as shown here 

20.1.2 Features to be tested

#	Implemented Feature	#	Implemented Feature
1	Calculation: Pregnancy Dating, Disorders and Markers.	7	Input Validation
2	Database: Data Access and retrieval	8	Caching
3	User Interface	9	Reporting system
4	Security.	10	Internationalisation
5	Help	11	Exception handling system
6	Date picker	12	Installation

20.1.3 Features not to be tested

#	Reason for non-implementation
---	-------------------------------

1	Batch calculation
2	File importation and exportation

20.1.4 Environmental needs

The environmental needs are the software and data required to execute these tests on the system. They are:

- A settings database created in Microsoft SQL server database as explained in Appendix XX – Product Content
- A web server with an installation of Microsoft Internet Information Services

20.2 Test case specifications

20.2.1 Data access

Test Case Specification	Notes
<p>Database connection</p> <p>PI 6.1 Change connection string for settings database to an incorrect one before start up</p> <hr/> <p>PO 6.1 Error on login.</p>	Shows the correct database is being used
<p>Missing current data: No current record</p> <p>PI 7.1 Try calculations with a required settings record missing</p> <hr/> <p>PO 7.1 Error on <i>Calculate</i></p>	

20.2.2 Security

Test case specification	Notes
<p>Validate username and password (login)</p> <p>PI 2.1 On login, user enters known correct login details</p> <p>PI 2.2 On login user enters incorrect details.</p> <hr/> <p>PO 2.1 Main screen displayed.</p> <p>PO 2.2 User not allowed to proceed to main screen, error message displayed</p>	All login details have to be correct, otherwise login fails
<p>Missing login detail(s)(login)</p>	

PI 3.1 At login some login details are not provided or entered in the provided fields	are compulsory
PO 3.2 Not allowed to proceed to main screen: Error on login	

20.2.3 Date Picker

Test Case Specification	Notes
PI 43.1 Date picker shows the correct days for the selected month and year test for leap years, normal years and all months.	Test dates: 31/02/2000 01/12/2006
PO 43.2 The Date picker displayed the correct number of days for all months in leap years or ordinary years	27/02/1996
PI 43.1 Date picker pastes the selected date into the date field correctly Try 5 different dates	
PO 43.2 The Date picker reads the selected value and correctly saves it in the appropriate date field	
PI 43.1 Date picker user interface works correctly and responds correctly to user interactions	
PO 43.2 Based on the test performed above, the date picker responded well to all clicks and other user interactions.	

20.2.4 Input Validation

Test Case Specification	Notes
Numerical validation	The program validates dates and numbers using the
PI 43.1, User enters an invalid number in numerical field	

<p>PI 43.1, User enters an valid number in numerical field</p> <p>PI 43.1, User enters an number with invalid number separators in numerical field</p> <hr/> <p>PO 43.1, Error on leaving field “Invalid number”</p> <p>PO 43.1, Number is accepted</p> <p>PO 43.1, Error on leaving field “Invalid number”</p>	<p>default English locale numerical and date separators and the separators of the user’s locale.</p>
<p>Date validation</p> <p>PI 43.1, User enters an invalid date in date field</p> <p>PI 43.1, User enters an valid date in date field</p> <p>PI 43.1, User enters an number with invalid date separators in a date field</p> <hr/> <p>PO 43.1, Error on leaving field “Invalid date”</p> <p>PO 43.1, Date is accepted</p> <p>PO 43.1, Error on leaving field “Invalid Date”</p>	<p>If an invalid date or number is entered, an error message is displayed and the field is refocused.</p>
<p>PI 43.1 The validation code correctly completes the year when the user enters a year of length less than 4 digits</p> <hr/> <p>PO 43.2 System correctly auto completes the year if the user has entered less than 4 digits for the year using the 30 year rule cut-off point..</p>	<p>If the length of the year is less than 4, the validation code auto completes it</p>

20.2.5 Help

Test Case Specification	Notes
<p>Help file</p> <p>PI 43.1 Help file exists</p> <p>PI 43.2 Help file does not exist</p> <hr/>	

PO 43.1 A Help file download box opens asking the user to open the document or save it	
PO 43.2 User informed that help file not found	

20.2.6 Caching

Test Case Specification	Notes
PI 43.1 Static components are stored in the internet cache of browser	
PI 43.2 Non-Static components are never cached	
PO 43.1 All images are cached and stored in browser cache, they show up more quickly from the second access onwards	
PO 43.2 Non-static components always change showing that they are not being incorrectly cached	

20.2.7 Reporting system

Test Case Specification	Notes
PI 43.1 The generated PDF patient report corresponds to the displayed results	
PI 43.2 All fields are positioned correctly and filled with the right values, do labels match their values	
PI 43.2 The PDF file is generated with default A4 dimensions	
PI 43.1 Manual comparison of the patient report shows that the patient report is generated correctly	
PI 43.2 comparison of the patient report to the template PDF file showed that the positions of control were not being changed when the template was filled to create the patient report	
PI 43.2 The PDF file is generated with default A4 dimensions (210 ×	

297 mm)	
---------	--

20.2.8 Exception handling

Test Case Specification	Notes
<p>PI 43.1 All errors messages are cleaned of brackets and punctuation to prevent conflict with JavaScript</p> <p>PI 43.2 The correct error messages are displayed for every exception</p> <p>PI 43.2 No post backs occur when an error is raised on the client side</p> <hr/> <p>PI 43.1 All errors messages contain strings only, verified through almost test cases</p> <p>PI 43.2 The correct error messages are displayed for every exception, on every test case, the displayed error messages are checked for correctness</p> <p>PI 43.2 Visually checked by raising errors and checking that no post backs occurred when an error had been raised on the client side</p>	

20.2.9 Installation

Test Case Specification	Notes
<p>PI 43.1 The system works correctly on an internet information services server see Appendix XX – Product Content for installation instructions</p> <hr/> <p>PI 43.1 The system was installed on the Media Innovation Ltd, all the test cases where performed on the application running on the server, under the same conditions as those encountered in normal use.</p>	

20.3 Calculation Test Cases

This section outlines the proposed tests to be performed on PrenatalLink to check if it calculates the correct results for the supplied data.

In every test, all the other input parameters will be valid except for the input parameter under investigation. Doing so ensures that the test results are mainly influenced by the parameter under investigation since all the other parameters are valid. For example in the Pregnancy Dating section, on the SiteID test, when the “Invalid SiteID” test is performed, all the other Pregnancy Dating inputs have to be valid, while the supplied SiteID is of the type suggested in the test.

20.3.1 Pregnancy Dating

Parameter	Data requirements
SiteId	Must be an integer ≥ 0
Patient DOB	Valid date not more than 60 years less than the sample date
Dating type	Within the EnGestationMeasure enumeration A GestationMeasureT record must exist for the DatingID
Dating date	Valid date less than the Sample date Must be supplied where necessary for the dating type
Dating measure	Must be within the range specified in the GestationMeasureT record for the Dating type, Site and SampleDate Must be supplied when required (e.g. Gestation, CRL, BPD, HC)
Monozygous	Can only be true for twin pregnancies, note all Assisted reproduction fields will be disabled unless the dating type is assisted reproduction
Donor Egg	Can only be true if the pregnancy was started through fertilisation
Ma at Extraction	Required if AR Donor DOB is not supplied or if patient DOB is not supplied in a assisted reproduction pregnancy
AR Event	Must be Fertilisation or Implantation
AR Donor DOB	A valid AR Donor DOB must not be more than 60 years less than the sample date

Invalid inputs

1. SiteID
 - 1.1. Invalid SiteID – e.g. negative SiteID
 - 1.2. Error on calculate: “Invalid site id”
2. Patient DOB
 - Invalid Patient DOB – (Resulting age outside of the range $0 < \text{age} \leq 60$)
 - Error on calculate: “Invalid maternal age or pregnancy dating information”
 - Patient DOB > Sample date
 - Error on calculate: “Invalid maternal age or pregnancy dating information”

3. Sample Date
 - Invalid Sample Date (e.g. a sample date that is less than the dating date)
 - Error on calculate: "Invalid pregnancy dating information"
4. Dating type
 - Invalid Dating type – GestationMeasureT record not present for Site and Sample date
 - Error on calculate: "Missing configuration data for BPD @ 01/01/1999"
 - Same Dating type used twice in the same calculation
 - Error on calculate: "Unable to add element to collection: existing element with same key"
5. Dating date
 - Dating date > Sample date
 - Error on calculate: "Invalid pregnancy dating information"
 - Dating date not supplied
 - Error on calculate: "Missing pregnancy dating information"
6. Dating Measurement
 - The input measurement is outside the gestation measurement range specified acceptable for the Dating type
 - Error on calculate: "X - value out of range"
 - Measurement supplied when not required (LMP, Assisted Reproduction)
 - LMP calculations must be successful but an error on calculation must occur for Assisted Reproduction calculations
 - Measurement not supplied when required (Gestation, CRL, BPD, HC,)
 - Error on calculate: "Missing pregnancy dating information"
7. Monozygous
 - Set to true when the pregnancy is not a twin pregnancy
 - Error on calculate: "Invalid pregnancy dating information"
8. AR Event
 - Invalid AR Event (not Fertilisation or Implantation)
 - Error on calculate: "Invalid pregnancy dating type"
9. Egg Extraction date
 - Invalid Egg Extraction date (When dating type= Assisted Reproduction)
 - Error on calculate: "Invalid pregnancy dating information"
10. Age on EDD
 - Invalid Age on EDD i.e. > 60 or < -1
 - Error on calculate: "Invalid pregnancy dating information"
11. Donor DOB
 - Invalid Donor DOB when Age on EDD is not given – (Resulting age outside the acceptable age i.e. > 60)
 - Error on calculate: "Invalid pregnancy dating information"
 - Invalid Donor DOB (Resulting age > 60) when Age on EDD is given
 - Error on calculate: "Invalid pregnancy dating information"
12. Correct calculation, the Donor DOB is not used
 - Calculating the correct results for valid inputs
 - Results
 - Calculating correct results for all dating types and parameters
 - Compare with manually calculated values

Pregnancy dating results

Expected date of delivery (EDD) - supplied or calculated

Gestation - returns calculated gestation at SampleDate

MAEDD - calculated age

20.3.2 Markers

Marker Information	Data requirements
MarkerName	All required elModel records must exist for the given Marker
AssayDate	Must be supplied in all calculations
Value & Normalisation	Value must be must be supplied and Cannot be < 0 except when the marker normalisation is Delta
SiteId	Integer >= 0
Gestation Type & Value	Both must be supplied if the value is not normalised, the value must be a valid gestation value
TwinValue	Must be 0 or not supplied if it's a twin pregnancy but the marker is not fetus specific Must be supplied if it's a twin pregnancy and the marker is fetus specific
TwinGestationValue	Both must be supplied if the Twin Value is not normalised (MoM) The gestation value must be a valid gestation
MaternalWeight	If supplied, it must be a positive value
Covariables (Additional information)	Checkbox based, if supplied they will be used in the calculation of marker moms
Key	Must be a string, the same key cannot be used more than once

Invalid inputs

13. MarkerName

- Two markers with the same gestation added to the Markers Collection
- Error on calculate: "Unable to add element to collection: existing element with same key"
- Two Markers at different gestations added to the Markers collection
- Correct calculation, the two markers will have different markers collection keys

14. AssayDate

- AssayDate not supplied
- Error on calculate: “Missing marker information”
- Value and Normalisation
 - Value is < 0 when Marker normalisation is not equal to delta and ValueIsNormalised
 - Error on calculate: “Invalid Marker information”
 - Value is < 0 when Marker normalisation is equal to delta and ValueIsNormalised
 - Correct calculation
 - Valid MoM when Marker normalisation is = delta (≤ 0)
 - Correct calculation
 - Value is < 0 and marker normalisation = Level or MoM
 - Error on calculate: “Invalid Marker information”
 - Value is outside the MoM limit range specified in the Marker distribution record
 - Error on calculate: “Invalid Marker information”
 - Supplied MoM is outside the MarkerDistributionT MoM range
 - Error on calculate: “Invalid Marker information”
 - Value not supplied or equal to 0
 - Error on calculate: “Missing marker information”

15. Gestation type and gestation value

- A gestation type value is supplied but the gestation value is not
- Error on calculate: “Missing or invalid gestation information”
- A gestation type is not supplied but a gestation value is supplied
- Error on calculate: “Missing or invalid gestation information”
- The marker value is not normalised and the gestation value & type are not supplied
- Error on calculate: “Missing or invalid gestation information”
- The marker’s value is normalised and the gestation value & type are supplied
- Correct calculation, since gestation values are not required in such a case
- The marker’s value is not normalised and a gestation value & type are not supplied
- Error on calculate: “Missing or invalid gestation information”
- The markers value is normalised and the gestation value & type are not supplied
- Correct calculation, since gestation values are not required in such a case

16. Twin Gestation

- Twin value not given but normalised and gestation value & type are not supplied
- Error on calculate: “Missing or invalid gestation information”
- Normalised value and gestation value & type are supplied
- Correct calculation

17. Weight

- Weight not supplied
- Correct calculation
- Invalid weight supplied (e.g. weight < 0)
- Error on calculate: “Invalid maternal weight”

18. SiteID

- SiteID not supplied

- Correct calculation, if default records exist
 - Invalid SiteID supplied (< 0)
 - Error on calculate: “Invalid SiteID”###
19. Covariables
- Invalid CovarID supplied (<= 0)
 - Error on calculate: #####
 - The same covariable is submitted more than once
 - Error on calculate: “Unable to add element to collection: existing element with same key”
 - A value is not supplied for a covariable
 - Error on calculate: “Invalid maternal weight”
 - A value is not supplied for a boolean covariable, (CovarID >= 5)
 - Correct calculation
 - An invalid value is supplied for a non-boolean covariables (> upper limit ###)
 - Error on calculate: #####
 - Covariables used correctly in calculating the Normalised median ###
 - Correct median = Covar factor applied (+ or *) to raw median
20. Calculating the correct results
- Results
 - Calculating correct results for markers
 - PrenatalLink results must be equal to manually calculated results

Markers results

Median Value - used in calculating MoM

Normalised value - Normalised for gestation and corrected by covariable info

Raw MoM - Property Normalised for gestation but uncorrected by covariable info

Twin Normalised value - If the marker is fetus specific

Twin Raw MoM - If the marker is fetus specific

20.3.3 Disorders

Parameter	Data requirements
DisorderId	The required elModel records must exist for the given disorder For Combined disorder, >= 2 disorders must be supplied
Risk Factor	A RiskFactorT record must exist for the given risk factor
Marker Set & Markers Collection	The markers that are required in the given MarkerSet must be supplied See the Markers test proposals above for the Markers data requirements Collection not required when calculating prior risk only
SiteId	Integer >= 0

MAEDD	Valid age, within the Disorder's age range
NumFets	Valid integer > 0

Invalid Inputs

21. Disorder type
22. SiteID
 - SiteID not supplied
 - Correct calculation if defaults exist
 - Invalid SiteID supplied (string or < 0)
 - Error on calculate:
23. MAEDD
 - Invalid Maedd supplied (< 0)
 - Error on calculate: "Invalid Maedd"
 - Maedd out of Disorder's age range
 - Error on calculate: "Maternal age out of range"
24. Number of Fetuses
 - Number of Fetuses not supplied
 - Correct calculation, default number = 1
 - Invalid number of fetuses supplied (< 1)
 - Error on calculate: "Overflow"
 - Valid number of fetuses supplied but out of calculatable range (0< Num Fets <3)
 - Error on calculate: "Cannot calculate risks and results in grand multiple pregnancy"
25. DisorderID
 - Disorder not supplied / invalid (DisorderT record does not exists)
 - Warning on calculate: "No disorder results requested"
 - DisorderID is Combined and 2 extra disorders are not supplied
 - Error on calculate: "Must include >=2 disorders for combined risk"
 - When DisorderID is Combined and 2 extra disorders are supplied
 - Correct calculation
 - The same Disorder is specified more than once in the same calculation
 - Error on calculate: "Unable to add element to collection: existing element with same key"
26. Markers Collection and Marker Set
 - Marker Set not supplied
 - Correct calculation, all supplied markers are used
 - Required markers in marker set are not supplied in markers collection
 - Error on calculate: "Marker(s) not supplied"
 - Only the required markers in the marker set are supplied
 - Correct calculation
 - Extra markers that are not in marker set are supplied in markers collection
 - Same results as when only the markers in the marker set are supplied, extra markers are not used.
 - PrenatalLink must use only those markers that are in the specified marker set
 - Non-required markers that are in the marker set are supplied in markers collection

- Correct calculation, all markers that are in the marker set are used
- A marker set for NT has AFP only and AFP is supplied in markers collection
- Correct calculation, NT markers sets must have AFP only
- A marker set for NT has AFP and other markers (An NTD marker set can have only AFP)
- Error on calculate: “Invalid marker information”

Calculating correct disorders results

- 27. Prior Risk
- 28. Posterior Risk
- 29. Results (Sample calculations)
 - Calculating correct result – compare PrenatalLink V3 results to PrenatalLink V1 results for the same settings and input.
 - The results must be correct for the Posterior risk result
 - Calculating the correct result in relation to risk cut-offs, truncation limits, gestation limits and twin results cut-offs)
 - The calculated results must be correct for the input data

20.4 Requirement vs. Test Case Traceability Matrix

Requirement	Test Cases
UR-RES1	
UR-RES2	
UR-RES3	
UR-RES4	
UR-SOU1	
UR-SOU2	
UR-SOU3	
UR-USE1	
UR-USE2	
UR-USE3	
UR-USE4	

Requirement	Test Cases
UR-USE5	
UR-USE6	
UR-USE7	
UR-VIE1	
UR-VIE2	
UR-VIE3	
UR-VIE4	
UR-VIE5	
UR-REP1	

20.5 Testing Results

21 Appendix F: Evaluation

21.1 Media Innovations Ltd Project Evaluation

21.1.1 Project: Elipse PrenatalLink (Beta test version)

21.1.2 Project summary

Elipse PrenatalLink is the latest item in the Elipse suite of products. It is a web-based version of Elipse, using the same calculation engine (DLLs) as the Win32 version that is in use worldwide. The aim of Courage's project was to design and build a Web application for the existing calculation engine, developing it to the Beta test stage, with the intention that it would not require much modification for commercial release.

The Elipse PrenatalLink project is a commercial development, and Courage was required to work to professional standards in all areas of his work. This included:

- Taking primary responsibility for system design.
- Adhering to the user requirements.
- Adhering to Media Innovations software development standards.
- Producing high-quality fault-free code.
- Producing documentation in accordance with Media Innovations systems.
- Working to strict deadlines.

21.1.3 Product specification

The following functionality was required:

- A security and login system that prevents unauthorised access and allows customisation.
- Data entry and validation.
- Calculation and presentation of results.
- A permanent report that can be saved on the user's PC and printed.
- The web pages must run satisfactorily on user PCs of various specifications, on a Windows platform.
- The application must work correctly in different locales.

21.1.4 Project tasks

In order to carry out the project, Courage was required to:

- Learn about the scientific background to Prenatal Screening.
- Understand the Elipse calculation engine, its various components and its interfaces.
- Within the constraints of Media Innovations approved development platforms, decide on the division of functionality through the application (e.g. between the server and the client).
- Research and select the appropriate implementation methods and technologies.
- Carry out the problem-solving, development, testing and fault-finding tasks with minimal assistance.

21.1.5 Evaluation

The Beta testing is at an early stage, so we do not have much user feedback. Based on my own assessment and the responses we have received:

- The development met the commercial deadlines.
- The product meets the specification.
- So far, the Beta testing has revealed only minor errors, which Courage has corrected.
- The overall look is very professional, and appropriate to a clinical application.
- The layout of the user elements (data-entry form, the results screen and the printable report) is well thought out and logical.
- The product does not require much further work to be developed into the commercially-released version; some minor modifications and more extensive validation.

This was quite a complex development, requiring a good understanding of the scientific background, a variety of IT platforms and components, and issues of commercial development and deployment. It gave Courage considerable scope for using his own initiative in design and implementation decisions and problem solving, and he worked throughout without needing close supervision.

I am very satisfied with Courage's contribution to the project, and with the finished product.

28 April 2006

Carol Wilson

Scientific Software Ma

22 Appendix G: Product Content

22.1 Installation and Setup

22.1.1 System Requirements

- Microsoft IIS server
- Microsoft SQL Server

22.1.2 Installing the database

- Open Enterprise manager
- Go to database and right click the databases tab
- Select restore database
- Use the back-up file at D:/Software/ElipsePrenatalLink/DLLs/elModel_BatchTest
- In the dialog box that opens, give the file the database the name elModelV3
- Follow all the other instructions and choose security options as required
- Open the restored elModelV3 database and click logins
- Give the elUser and elAdmin permission to access the newly restored database

22.1.3 To install the system:

- Create a real folder to store the system files and copy all the files under D:/Software/ElipsePrenatalLink into that directory
- Create an IIS Virtual folder for the system according to the IIS guidelines and set the permissions according to your security requirements and point it to the real folder
- Start-up you browser and point it to the virtual folder and the login page of the application will be displayed

22.2 Implementation Document

(Sample page from the implementation document describing the security class)

22.2.1 UserConnect Class

This class encapsulates the functionality required to read user records from the ElipseUserT table

Public Sub New()

Used by the Security class, this method initialises the variables required to connect to elModelV3 (SQL Server 2000 database)

Private Sub LoadUsersDataset()

Called from default.px, this method connects to elModelV3, uses the select commands to fill the Users and Customers datasets with records from elModelV3

Public Function GetUserInfo (strFieldName, strUsername)

Takes a name of an ElipseUserT field and a username and returns the value of the specified field from the row that h the given username

Public Function GetCustInfo(strFieldName , strCustId)

Takes a name of an ElipseUserT field and a username and returns the value of the specified field from the row that h the given username

22.2.2 Security Class

Private Sub Page_Load(sender , event) Handles MyBe.Load

This method is executed every time the page is loaded, it mainly initialises the login vars

Private Sub butLogin_Click(sender , event) Handles butLogin.Click

This method is executed every time the page is loaded, it mainly initialises the login vars

Private Sub btnReset_Click(sender , event) Handles btnReset.Click

Clears the text in the login fields, only called if the client h javacript disabled

Private Function ChkUserDetails() Boolean

Checks a users login details and returns true if they are correct

Private Sub SetInitialFocus(ctrl Control)

Sets focus to the given control on this page

Private Sub SaveLoginInfo()Writes the current username used to login in the cookie username.



Prenatal  Link
Web Solution

User Guide





Contents

1	Introduction	108
2	Logging In	111
3	Calculation	108
3.1	Date Input	112
3.2	Validation	113
3.3	Calculating results:	114
3.4	Data Entry Sections	114
4	Reports	117
4.1	Results	108
4.2	Patient Report	117

23 Introduction

This document contains instructions on how to use the Prenatal Link beta test system. Although this system has been subjected to preliminary testing, it is yet to undergo extensive clinical evaluation and testing by third parties. It is expected that the user will have read the accompanying report or has background knowledge on prenatal screening. Knowledge of Prenatal screening theory is required to be able to use the system.

The PrenatalLink system was developed by Courage Chibiya in conjunction with Media Innovations Ltd.

Contact

Media Innovations Ltd

3 Gemini Business Park

Sheepscar Way

Leeds

LS7 3JB

UK

Tel: +44 (0)113 262 1600

Fax: +44 (0)113 262 1605

24 Logging In

(a) Start-up your Internet Explorer browser and enter the URL shown the image below into the address bar or if you have installed the Prenatal Link system on your machine, point the browser to the path of the virtual IIS folder that contain the system (See Appendix XX – Product Content for installation instructions).



(b) Enter the correct Customer Id, Username and Password into the appropriate fields and click the “**Log in**” button

A screenshot of a web form titled "Login". The form has a light blue border and a white background. It contains three input fields: "Customer Id", "User", and "Password". Below these fields is a checkbox labeled "Remember customer id and user name on this computer (your browser settings may not allow this)". At the bottom of the form are two buttons: "Log in" and "Reset".

Note: All login details are compulsory. If you enter incorrect login details and wish to empty the contents of all login fields, click “**Reset**”.

For testing purposes, you can login using the following login details:

- CustomerId: ltest
- Username: Leeds
- Password: pass

25 Calculation

The figure below shows a screenshot of the calculation page. To calculate results you will be required to supply patient, dating, markers and disorders information. The following paragraphs will explain how to calculate results.

elipse PrenatalLink Screening Calculator **Mmedia innovations**

Patient

Surname
 Forename
 Id
 DOB
 Ethnic

Pregnancy

Number of Fetuses
 Monozygous
 Dating Type
 Measurement
 Dating Date
 Donor Egg
 Egg Extract Date
 MA @ Extraction
 MA Truncation
 Donor DOB

Markers

Default Date Ultrasound Site

Marker (Ultrasound)	Date	Value	Value 2 (Twin)	Unit	Is MoM
NT	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/>	mm	<input type="checkbox"/>
NB	<input type="text"/> <input type="button" value="v"/>	<input type="text" value="N/K"/> <input type="button" value="v"/>	<input type="text" value="N/K"/> <input type="button" value="v"/>		

Marker (Serum)	Date 1	Value 1	Date 2	Value 2	Unit	Is MoM
AFP	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	iu/ml	<input type="checkbox"/>
HCG	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	iu/L	<input type="checkbox"/>
HCGb	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	ng/ml	<input type="checkbox"/>
Inhibin-A	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	pg/ml	<input type="checkbox"/>
PAPP-A	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	mU/L	<input type="checkbox"/>
UE3	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	<input type="text"/> <input type="button" value="v"/>	<input type="text"/>	mU/L	<input type="checkbox"/>

Additional Information

Weight (Kg)

Diabetes Smoking IVF PrevT21 PrevT18

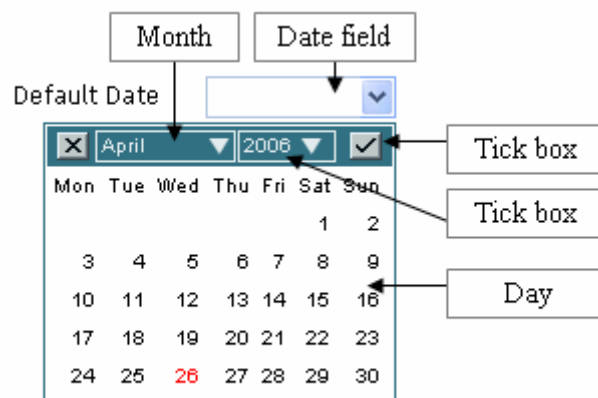
Tests

T21
 T18
 T13
 T18_13

Results Help

25.1 Date Input

(a) To enter dates, click on the drop down icon of the date field to display a date picker



(b) Select the year and month from the year and month dropdown boxes respectively

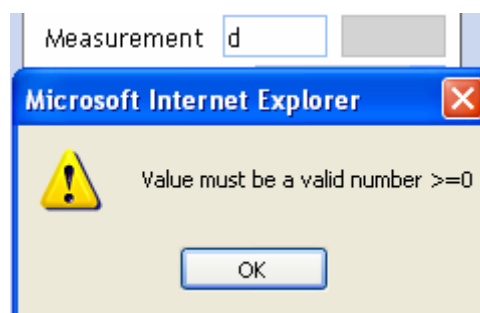
(c) Choose a day by clicking the required day number.

(e) Click to paste the selected date into the date field and close the date picker

(f) Click to cancel and close the date picker

25.2 Validation

Input validation is performed for numerical and date fields. When inputting dates and numbers ensure valid values are entered before starting calculations.



If an invalid number or date is entered, an error message will be displayed, describing the cause of the error and the corresponding field with an invalid value will be refocused to enable you to correct the entered value.

25.3 Calculating results:

Instructions for calculating results:

- In the data-entry screen form, complete the necessary fields in each section as explained in the data entry section below.
- Select the disorders for which a result is required.
- Click Calculate.
- From the results screen, click Print if required.

25.4 Data Entry Sections

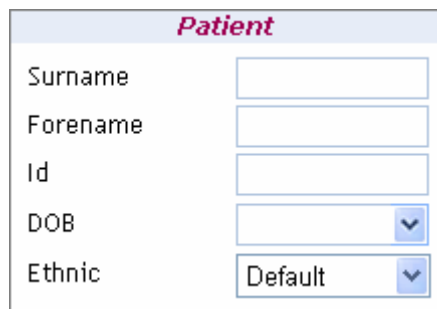
25.4.1 Patient Information

Surname, Forename, Id: These fields are optional, but appear on the report.

DOB: Required

Ethnic: Optional. Serum markers can have ethnic-specific median corrections.

(a) Enter the patient's surname, forename and id in the appropriate fields.

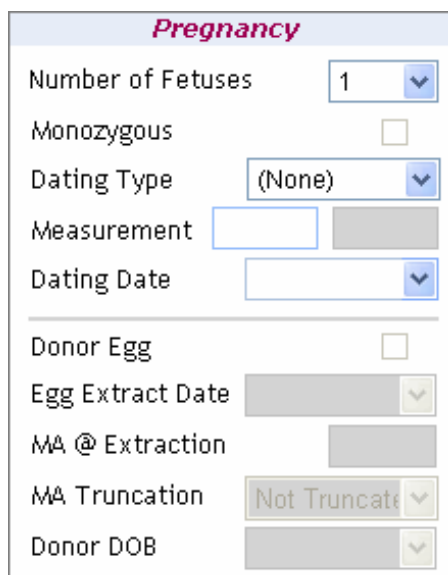


The screenshot shows a form titled "Patient" with the following fields:

Surname	<input type="text"/>
Forename	<input type="text"/>
Id	<input type="text"/>
DOB	<input type="text"/> ▼
Ethnic	Default ▼

(b) Use the data picker to select the patient's date of birth

(c) Select the ethnicity of the patient from the ethnic drop down list



The screenshot shows a form titled "Pregnancy" with the following fields:

Number of Fetuses	1 ▼
Monozygous	<input type="checkbox"/>
Dating Type	(None) ▼
Measurement	<input type="text"/>
Dating Date	<input type="text"/> ▼
Donor Egg	<input type="checkbox"/>
Egg Extract Date	<input type="text"/> ▼
MA @ Extraction	<input type="text"/>
MA Truncation	Not Truncate ▼
Donor DOB	<input type="text"/> ▼

25.4.2 Pregnancy Dating

Number of Fetuses: If twin pregnancy, select this first.

Dating Type: Required. Select the type of measurement to be used to calculate gestation ('Gestation' means that the calculated gestation will be entered in the Measurement field).

Dating Date: Required. Date at which Measurement taken.

Measurement: Required. Measurement of the type selected in the Dating Type field. For twin pregnancy with ultrasound measurements, enter a measurement for each fetus.

The remaining fields are for Assisted Reproduction (AR)

pregnancies (patient's stored egg or embryo, or donor egg) and are only enabled for that AR.

Donor Egg: Checkmark if egg donation pregnancy.

Storage Date: Optional. For stored egg/embryo: the date at which the egg/embryo (patient or donor) was extracted and put into storage, for adjusting MAEDD.

MA at Storage: Optional. For stored egg/embryo: if Storage Date (and Donor DOB for egg donation pregnancy) are unknown, enter an approximate maternal age at date of storage (in years plus decimal part-year), for adjusting MAEDD.

MA Truncation: If MA at Storage is in completed months, half-years or years, select the appropriate truncation.

Donor DOB: Optional. For an egg donation pregnancy either this field and Storage Date or MA at Storage must be completed.

25.4.3 Markers

Markers						
Default Date	26 Apr 2006	Ultrasound Site		Default		
Marker (Ultrasound)	Date	Value	Value 2 (Twin)	Unit	Is MoM	
NT	26 Apr 2006			mm	<input type="checkbox"/>	
NB	26 Apr 2006	N/K	N/K			
Marker (Serum)	Date 1	Value 1	Date 2	Value 2	Unit	Is MoM
AFP	26 Apr 2006				iu/ml	<input type="checkbox"/>
HCG	26 Apr 2006				iu/L	<input type="checkbox"/>
HCGb	26 Apr 2006				ng/ml	<input type="checkbox"/>
Inhibin-A	26 Apr 2006				pg/ml	<input type="checkbox"/>
PAPP-A	26 Apr 2006				mU/L	<input type="checkbox"/>
UE3	26 Apr 2006				mU/L	<input type="checkbox"/>

Default Serum Date: Optional. If provided, this date is automatically copied to each serum date field (but can be overwritten in individual fields if there are samples from different dates).

Ultrasound Site: Optional. If results are received from more than one ultrasound centre the appropriate one must be selected, as centres can have different calculation parameters.

Ultrasound markers: Complete all enabled fields. For a twin pregnancy a result must be entered for each fetus. 'Is MoM' means that the value entered is a MoM, not a raw level.

Serum Markers: Complete all enabled fields. Results from two different serum samples may be entered for each marker. 'Is MoM' means that the value entered is a MoM, not a raw level.

25.4.4 Additional Information

<i>Additional Information</i>					
	Date 1	Value 1	Date 2	Value 2	
Weight (Kg)	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
Diabetes	<input type="checkbox"/>	Smoking	<input type="checkbox"/>	IVF	<input type="checkbox"/>
		PrevT21	<input type="checkbox"/>	PrevT18	<input type="checkbox"/>

Weight: Optional. Enter a weight measurement taken as close as possible to the serum sample date. A second weight and date can be entered if there is more than one serum sample.

Covariables: Optional. To select Covariables, check mark the Covariables to be included in the risk calculation.

25.4.5 Disorders

<i>Tests</i>	
T21	<input type="checkbox"/>
T18	<input type="checkbox"/>
T13	<input type="checkbox"/>
T18_13	<input type="checkbox"/>

Select all the disorders required by check marking the appropriate checkbox, at most one disorder is required for a calculation to succeed.

Notes

For twin pregnancies with ultrasound marker measurements fetus-specific risks are provided. It is essential to ensure that all fetus-specific information is entered in the same fetus order in every place on the data-entry screen.

26 Reports

26.1 Results

Patient	
Surname	<input type="text"/>
Forename	<input type="text"/>
Id	<input type="text"/>
EDD	4 Nov 2006
MAEDD	23 yrs 6 mnths

Markers				
	MoM/Delta	Gestation	Twin 2 MoM/Delta	Twin 2 Gestation
AFP	11.00	12 wks 4 days		

Risks					
	Prior Risk	Final Risk	Result	Twin 2 Final Risk	Twin 2 Result
T21	1 in 1447	1 in 2771	Screen Negative		

26.2 Patient Report

- (a) Click on create report to generate a PDF patient's report
- (b) A dialog box will be displayed asking you to save or open the patient report file

