Mt Pleasant High School

Levy Fees Processing System

Development Manual

BY

Tinovimba G Motsi

www.geocities.com/tgmotsi

“Not all those who wander are lost”

Gandalf, The Fellowship of the Ring (J.R.R. Tolkien)

Terms of Reference

This project has been duly authorised for submission in partial fulfilment of the requirements of the Bachelor of Business Studies and Computing Science (Honours) degree program under course code CT260. The Permanent Secretary of the Ministry of Education, Sports and Culture granted the ministry's head office research clearance on Thursday the 27th of February 2003. The Regional Director (Harare Province) also cleared the research on Friday the 28th of February 2003. Please refer to the Appendix for research clearance documents

Preface

The documentation for this project has been organised into four major sections. The first of these, entitled Requirements Analysis, documents the process that led to the formulation of the functional and non-functional requirements of the system. It starts off by stating the rationale of the study and then moves on to describe the methodology used in studying the current system. A major subsection is then devoted to a description of the current system with the section ending with the listing of the requirements along with a study into their feasibility.

The Requirements Analysis section is followed by a section on design. The design starts off with a brief overview of the design philosophy that guided the design process. Being a data heavy application, the overview is followed by a subsection on the modelling of the system database. In line with the method proposed by Somerville (1995), this is followed by Architectural design with the section concluding with a section on the design of all the identified components.

A section on the evaluation and review of the system follows the Design and serves as a conclusion to the developer's section of the documentation. This section concentrates on highlighting the bond between the requirements and the implementation along with the known inadequacies of the system. It concludes with proposals on plans for the future development of the system.

The last section contains a user manual that contains detailed instructions on using the system. This chapter concludes the main body of the documentation. The documentation also includes an appendix with the research clearance documents.

Acknowledgements

First and foremost, I would like to thank my family for their patience and understanding during the many hours I disappeared and became very antisocial. My heartfelt thanks also go to Dr G. T. Hapanyengwi, my project supervisor, for his patience and guidance over the course of the project. I would also like to express my gratitude to Ms L. Bonda, my first project supervisor, for her advice on carrying out the background research and refining my project proposal.

My gratitude also goes to Ms D Ndlovu, the acting chairperson of the University of Zimbabwe Department of Computing Science for preparing the letter of authorisation that was required as part of the application for clearance from the Ministry of Education, Sport and Culture. I would also like to thank Mr Sidimeli, Mr Chinyama, Mrs Kabasa, Ms Chokumanja and Mr Mukwandara of the same department for their assistance with access to the departmental resources.

At the Ministry of Education, Sport and Culture head office, I would like to thank Mr Chinamasa and Mr Kuona along with Mr Denhere of the same ministry's Harare regional office for their assistance in processing the applications for ministerial approval of the use of Mount Pleasant High School as a case study.

I would also like to thank Mr Mugwenda and Mrs Gozho, the head and deputy head respectively of Mt Pleasant High School along with their administrative assistants, Mr Muswe and Mrs Nyaruwa, for making the interview arrangements with the Bursar's and Levy office on my behalf. I would also like to thank Miss Patsika, the Bursar of Mt Pleasant High School, for outlining the government fees section of the fees system and Mr Mukanyi of the Levy Office for outlining the Levy processing system.

References

The reader may find it useful to consult the following documents that played an integral role in shaping the course of this project.

1. Hapanyengwi GT & Mazhindu-Shumaba RK, 2003, Project Planning and Implementation For the Computing Science Student, University of Zimbabwe Course Pages, http://www.uz.ac.zw/science/cs/ct260/Project%20Planning%20and%20Development%20Book.DOC
2. Microsoft, 2003, Issues Migrating from DAO/Jet to ADO/Jet, Microsoft Support, http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/q225/0/48.asp&NoWebContent=1
3. Microsoft, 2001, Microsoft Developer's Network, Microsoft, www.msdn.microsoft.com
4. Motsi TG, 2001,Librarius 2000 Plus, Geocities Homepages, www.geocities.com/tgmotsi
5. Pierre-Emmanuel G, 2003, RTF Printing Routines, Code Guru Website, http://www.codeguru.com/vb/openfaq/comments/159.shtml
6. Somerville Ian, 1995, Software Engineering, Addison Wesley

Contents

Requirements Analysis

Rationale

The administrative office of Mt Pleasant High School has three computers. Most of its processes however continue to operate on manual systems. Of particular note has been the fees processing system. It has therefore been decided to explore the benefits that computerisation of the fees processing system could unlock in terms of running efficiency and resource utilisation.

Methodology

The research will be by means of interviews with the Bursar and personnel in the Levy Office. The following is the list of intended questions

1. **Types of fees charged**
    a. What fees do you charge?
    b. To whom do these fees apply?
    c. Who is responsible for which fees?
    d. Who gives the directive to charge any given student a given fee?
    e. What documentation if any is used in (d) above?
    f. What documentation used in communicating the fees charged against any given student to that student?

2. **Storage of information**
    a. What specific transaction records do you keep?
    b. What documentation is used in this storage?
    c. For how long are these records kept?
    d. What happens to the records when their period of use has expired?

3. **Uses of information**
    a. What specific uses do you put the above specified information?
    b. Who are the consumers of this information and what specific aspects of it do they require?
    c. What problems, if any, have you experienced in storing or accessing this information?
    d. What opportunities for further utilisation of the information would you want explored?

4. **Computer System**
    a. Do you have a computer system within your immediate office?
    b. If you do have a computer system, what are its technical specifications?
    c. How would you appraise your level of computer literacy?

The Current System

The fees charged by Mount Pleasant High School fall into two categories. The first is made up of so called "government fees" and the Bursar administers this group of fees. The second is made up of levy fees and the Levy Office administers it. The two offices are fully autonomous of each other.

The Bursar's Office

Four fee categories fall under the Bursar's office. These are

a) The tuition fee
b) The General Purpose fee
c) Industrial fees.
d) Examination fees

The tuition fee is gazetted by government and communicated to the school head by means of a circular from the Ministry Of Education, Sport And Culture. All students in former "Group A" schools pay the same tuition fee. Government also gazettes the General Purpose fee. The government gazettes a range and then invites school heads to select and apply for a specific figure within this range. Industrial fees are charged for all practical subjects with variations being allowed for across the various academic levels. Examination fees are also gazetted and accepted by the Bursar on behalf of the Zimbabwe Schools Examination Council.

All fees are payable in advance. Students are therefore invoiced upon registration and during the last week of each term if the student wishes to continue with his/her studies in the subsequent term. The invoices used enjoy statutory instrument status and must be obtained from the Government Printers. The invoice is the primary document used when paying fees into the school's "School Services Account" and banks are required to only accept payments accompanied by invoices and to note the invoice number when recording the deposit.

Upon payment, the paying agent is issued with two deposit slips one of which must be presented to the school as proof of payment. The school will, upon deposit slip receipt, issue an admission form from the "School Services Fund" receipt book that is also obtained from the Government Printers. Before issuing the admission form, the Bursar is required to ensure that the student actually paid the amount invoiced. If there is a shortfall, she is not allowed to issue the admission form and the bursar must refund any surplus. These refunds are usually in the form of cheques. No balances maybe carried forward or transferred to other accounts including those administered by the Levy Office. For those students who lose their deposit slips, an admission form maybe issued upon receipt of the school's bank statement if it contains a record of a deposit made against the invoice number issued to the student.

The Bursar has an iMac computer with an Epson bubble jet printer. The computer system is however not used in the processing of fees as the accounting system used also enjoys statutory instrument status and does not allow for computer-aided fees processing.

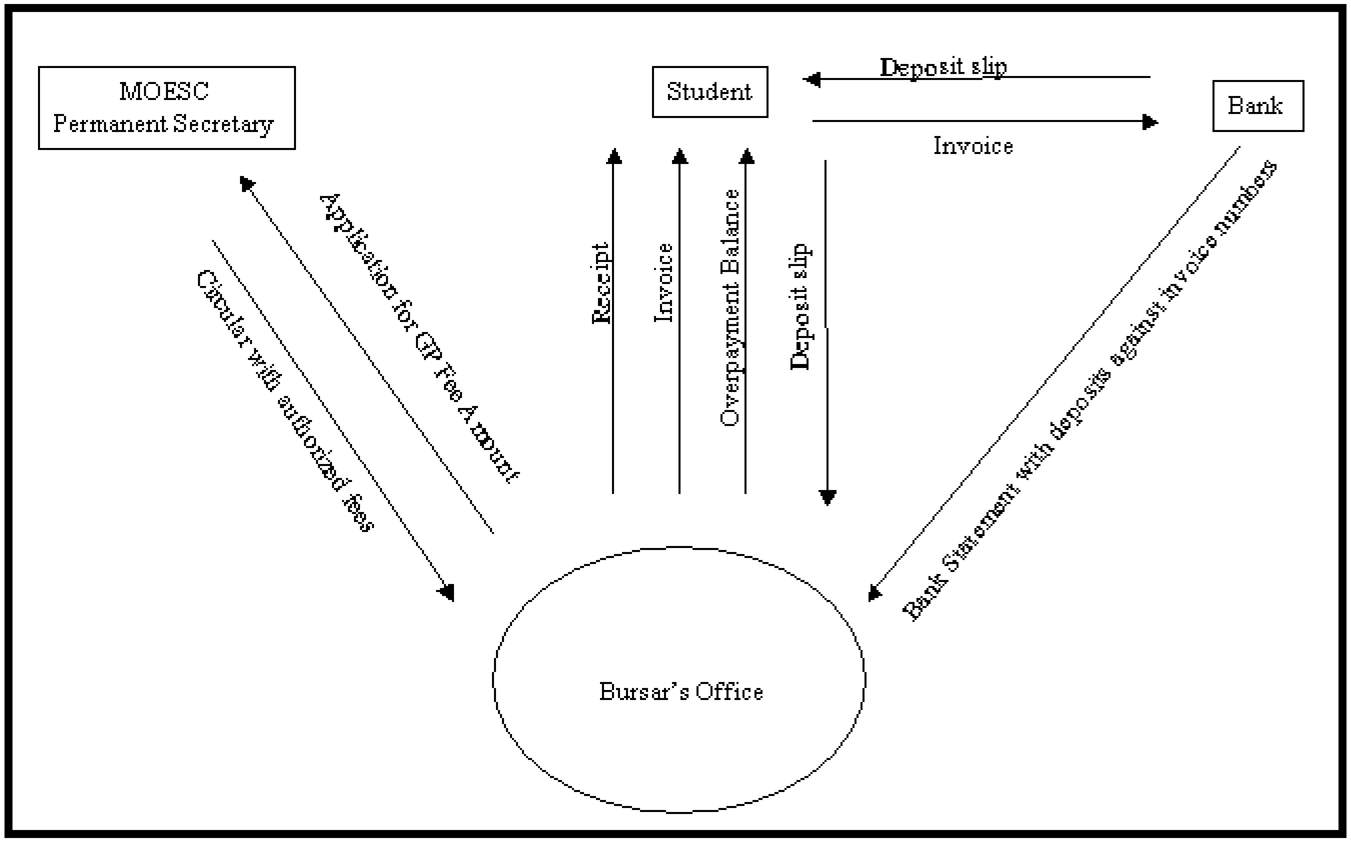

MOESC Permanent Secretary
Student
Bank
Deposit slip
Invoice
Application for GP Fee Amount
Circular with authorized fees
Receipt
Invoice
Overpayment Balance
Deposit slip
Bank Statement with deposits against invoice numbers
Bursar's Office

Figure 1: Bursar's Office Context

The Levy Office

The Levy Office handles the following fees

a) Registration fee
b) General levy
c) Project fund
d) Caution fee deposit
e) School Bus Fund
f) Building Fund
g) Dinner dance levy
h) Science levy
i) Magazine levy
j) Report book levy

The amounts charged under each fee vary with the academic year of the student. The following categories are currently in use

- Forms 1 and 2
- Forms 3 and 4
- Advanced Level Sciences
- Advanced Level Non-Sciences

The School Development Association, subject to ministerial approval, sets the above fees.

A student returning to the school for Advanced Level studies is required to reregister and thus pay a registration fee. The report book fee on the other hand is paid only on initial registration. The caution fee is also payable upon registration and must be maintained at the prescribed minimum level. A student, who incurs any financial liability, for example by loosing a textbook or damaging laboratory equipment, has his/her account debited. Upon clearance for deregistration, the balance on the account is refunded.

For the purpose of accounting, the Levy Office maintains a Total Debtors account with the individual student accounts being kept as memoranda accounts. It also maintains total fees accounts for each fee class. Entries in the Total Debtor accounts and Total fees accounts are based on figures extracted from these memoranda accounts. The individual student accounts are maintained using an Excel 7.0 spreadsheet. For audit purposes, a break down of the total debtors figure into credit balances for those who have prepaid and debit balances for those still owing is required.

Upon registration, a student is added to the corresponding worksheet in the spreadsheet for the given calendar year. His/her account is then automatically debited. Upon payment, the account is credited and a receipt issued. These transactions often occur one after the other. The invoicing of fees at the end of the term is however procedurally different. Before invoices are prepared, class lists are updated using the class attendance registers. In addition, the student's current "levy fees owing" balance also has to be calculated. This is necessitated by the fact that a student who has failed to pay the required levies cannot be denied education and may thus carry a negative balance until legal recourse remedies the situation. Also, the Levy Office does not refund those who pay more than what they will have been invoiced unless they specifically make a request for this to be done. The prepared invoices are then used to make the payments through the School Development Association's bank account using the same procedure as that used when paying "government" fees.

The Levy Office has an IBM compatible personal computer and an HP LaserJet 5L printer. The system runs on the Microsoft Windows 95 platform. The principle officer in the Levy Office is quite proficient with regards to using the system as evidenced by the modifications he has made to the Excel 7.0 spreadsheet he has been using.

Statement of Preliminary Feasibility

After carrying out the above research, the project developer has ascertained that the statutory requirements for the Bursar's office make the computerisation of that office's fees processing system unfeasible. As a result, the project scope has been revised to cover the Levy Office only. In light of this, all references in the remainder of this document to the "Fees Processing System" will be with respect to the "Levy Fees Processing System".

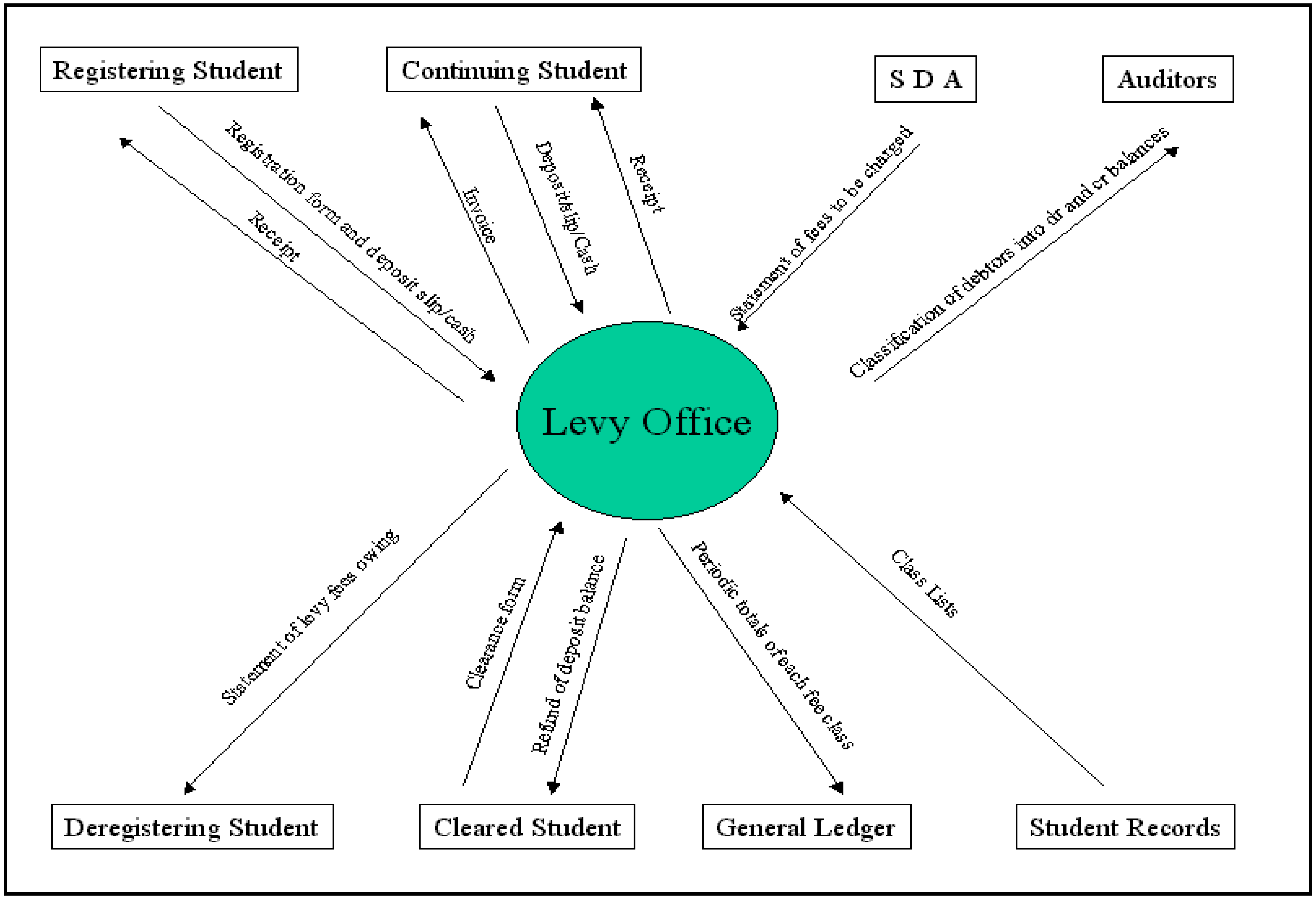

Registering Student
Continuing Student
S D A
Auditors
Registration form and deposit slip/cash
Receipt
Invoice
Depositslip/Cash
Receipt
Statement of fees to be charged
Classification of debtors into dr and cr balances
Levy Office
Statement of levy fees owing
Clearance form
Refund of deposit balance
Periodic totals of each fee class
Class Lists
Deregistering Student
Cleared Student
General Ledger
Student Records

Figure 2: Levy Office Context

Functional Requirements

Requirements Definition

The system must

1. Facilitate the maintenance of student records
2. Facilitate the recording of fees debited to each student account.
3. Facilitate the recording of all payments.
4. Be able to produce statements of account for each student.
5. Be able to extract the totals fees charged under a given fee over a user-defined period.
6. Be able to analyse the student account balances into debit and credit balances.
7. Maintain a "Caution deposit account" for each student.
8. Facilitate the sending of reminders to parents with fees owing.

Requirements Specifications

1. The system must facilitate the maintenance of student records
    a. The system should store the following details with regards to each student
        i. A unique identification code - Student ID
        ii. Surname
        iii. First names
        iv. Class
        v. Sex
        vi. Guardian's
            - Surname
            - Initials
            - Title
            - Postal Address
            - Telephone Number

Rationale for 1.a.i	This data item has been added because the combination of Surname, First names and class cannot be guaranteed to be unique.
Rationale for 1.a.v	See requirement specification 2.c
Rationale for 1.a.vi	Required for generating mailing labels. Telephone number added aid contacting of guardians by telephone. Also see requirement 8.

    b. Facilities should exist for finding all the student details given either of the attributes listed below. Where multiple matches have been found, they should all be made available. The facility should also support pattern matching.
        i. Student ID
        ii. Surname
        iii. First names
        iv. Class
        v. Guardian's surname

Rationale for 1.b	Parents and teachers making enquiries often have partial details.

    c. The results of the process referred to in 1b should be available for output onto
        i. The Visual Display Unit (VDU)
        ii. The Printer.
        iii. They should also be available for exporting into the formats specified in Requirement 9.

Rationale for 1.c.ii	When a student or other designate is sent by administrative staff to collect the details of a given record,

	the results will be required on hard copy.
Rationale for 1.c.iii	Other offices may require or prefer an electronic copy of the results.

d. A facility for batch updating the class field by moving all the students in one class to another should be provided.

Rationale for 1.d	At the year-end, most of the students in a given class will progress to the corresponding class in the next academic level. This is a labour saving measure.

e. The following information should be stored about each class
   i. Name of class
   ii. Name of form teacher
   iii. Minimum Caution Deposit Account Balance

f. A deregistered student's details should continue to be stored within the databases. The record should be the same as that specified in 1a with the following exceptions
   i. The class field may be dropped.
   ii. An additional field specifying the date registered should be maintained.

2. The system must facilitate the recording of fees debited to each student account.
    a. The following should be recorded for each fee debited.
        i. Student ID
        ii. Date charged
        iii. Amount debited
        iv. Description of fee

Rationale for 2.a	This is the recorded information currently maintained in the spreadsheets used. The invoice number will no longer be stored, as the system will no longer process invoices. See Requirement 4.

    b. The system operator should be able to debit a student's account by a combination of the following
        i. Selecting an editable template containing a list of fees along with their associated amounts.
        ii. Specifying a single fee and its associated amount.

Rationale for 2.b	Fees charged on registration are similar and a template would ease entering of these and speed up the process. Reducing the workload will minimise errors.

    c. The operator should be able to charge all the students in a given class
        i. Facilities similar to those specified in 2b should be provided.
        ii. Before debits are made, the system should list the students to be charged sorted first as to sex then as to surname and first names.

Rationale for 2.c.i	The Levy Office updates its class records using attendance registers. The registers use the sorting order stated.

3. The system must facilitate the recording of payments
    a. The following details must be kept
        i. Receipt number
        ii. Date receipted
        iii. Amount paid
        iv. Student ID of student affected

Rationale for 3.a	These are the details currently stored by the Levy Office.

4. The system must be capable of producing statements of account for Levy accounts
    a. The statement should contain the following details
        i. Student ID
        ii. Name of student
        iii. Class
        iv. All transactions made between two user supplied dates
        v. Balance owing and due.

Rationale for 4.a	Details of the school bank account are not required as the paying agent is issued with a custom deposit slip.

    b. A facility for batch printing statements for all students in a given list of classes should be available.

Rationale for 4.b	Required for printing statements of account at the end of the terms for use by students when paying levies into the SDA bank account.

5. The system should be able to extract fees charged under a given fee over a user-defined period. The results should be available for both display and printing

Rationale for 5	Required for making entries into the general ledger.

6. The system should be able to provide an analysis of student account balances into debit and credit balances.
    a. The analysis should contain
        i. A list of all students with debit balances.
        ii. A list of all students with credit balances.
        iii. A list of all students with 0 balances.
        iv. Total debit balances
        v. Total credit balances

Rationale for 6.a.i	Required by administration when pursuing legal action.
6.a.ii & 6.a.iii	Inserted to cater for system evolution. Have no current use in the system
6.a.iv & 6.a.v	Required by auditors. Opportunity for use as an input by the SDA executive committee when planning.

    b. The output referred to in 6.a should be made available
        i. On the Visual Display Unit (VDU)
        ii. In hard copy
        iii. For export as to Requirement 9
    c. The source SQL should also be made available for export to facilitate Mail merge. See requirement 8.

7. The system should maintain a “Caution fee deposit” account for each student
    a. A facility for setting the minimum balance for each class should be made available. See requirement 1.e.

Rationale for 7.a	This is to cater for system evolution. It is not envisaged that a situation will arise whereby students in the same class will have to make different deposit amounts even if the current categories are subdivided. Current categories are • Forms 1 and 2 • Forms 3 and 4 • Advanced Level Sciences • Advanced Level Non-Sciences

    b. Deposit transaction records should contain the following field details.
        i. Date made
        ii. Amount
        iii. Student affected
        iv. Receipt Number

    c. Account debiting records should contain the following details.
        i. Date made
        ii. Reason
        iii. Reference code of the letter of authorisation from the disciplinary committee.
        iv. Amount debited

    d. A facility for producing a statement of account for any user defined period should be provided. The statement should contain a record of all deposits and credits along with the amount owing if any.

    e. A facility for batch producing the statements for all students whose balances are
        i. Below the minimum balance.
        ii. At the prescribed level
        iii. Above the minimum balance

Rationale for 7.e	It is not anticipated that the system will ever have to batch process the balances of those not owing but it has been decided that it would be prudent to include these facilities separately just in case the system evolves along an unexpected path.

8. A mail merge facility for sending reminders to parents with fees owing should be provided
    a. The user should specify the records to be used in the mail merge through Standard Query Language.
    b. The user should be able to specify the body of the message indicating where personalised details will be inserted.
    c. The facility should provide the following editor specific services
        i. Document preparation in the Microsoft Rich Text Format.
        ii. Saving and retrieval of documents in RTF.
        iii. Printing of RTF documents in their native style.
        iv. Search and replace function.
        v. Undo system with a capacity of undoing at least the last 10 changes made.
        vi. Clipboard functions (Cut, Copy & Paste)
        vii. The following RTF formatting styles should be provided
            - Bold
            - Underline
            - Italics
            - Multiple font sizes within a document
            - Multiple fonts within a document
            - Alignment
                - Left
                - Right
                - Centre
            - Indention
            - Image imbedding
        viii. A toolbar function for accessing all the functions listed in 8.c
    d. A facility for previewing the Mail Merge job should be provided.
    e. A facility for inserting the following into the current document
        i. Calendar Dates
        ii. A picture file.
        iii. A text or RTF file
    f. The mail merge job should run fully on facilities provided by the system.

Rationale for 8.a	This will allow the capacity of more reports to be made available by writing new SQL files. It will also provide an opportunity for the reuse of a system developed by the current developer in 1999.
Rationale for 8.b	This is a standard mail merge function.
Rationale for 8.c	The personnel are already comfortable with Microsoft products. The requirement will try to tap into this by specifying an interface that is inline with the MS Word interface that they are already used to.
Rationale for 8.d	To cut down on paper wastage.
Rationale for 8.e	Using facilities provided by external software would expose the school to the possibility of having to acquire

	third party software that may not be economically feasible.

9. A facility for exporting data in electronic form should be provided for use by the system
    a. The facility should support the following formats
        i. Generic comma delaminated format as implemented in Microsoft® CSV.
        ii. Microsoft Excel for Windows 1995

Rationale for 9.a.i	Will allow exporting of the data to the Macintosh computers and any other spreadsheet package.
Rationale for 9.a.ii	Will facilitate exporting of the data to the current version of Microsoft Excel used within the levy office.

10. To complement and extend requirement 1.b, the system should provide an SQL based interface for querying the system.
    a. The user should be able to Enter, Save, Retrieve and Print out the source SQL.
    b. The user should be able to export the results as to the formats specified in Requirement 9.

Non-Functional Requirements

1. The student ID codes should contain checksum data.

Rationale	To minimise data entry errors.

2. The system should restrict access by means of an editable password. It should also provide a facility for specifying access levels.

Rationale	Other administrative personnel sometimes use the computer when printing office documents.

3. The system should generate all printouts on A4 sized paper.

Rationale	This is the paper size in use on all the printers available in the office.

4. The implementation should run on Microsoft Windows 95.

Rationale	This is the operating system in use on the Personal Computer in the Levy Office

5. The system should not provide facilities for deleting transaction records of students currently registered or owing the Levy Office.

Rationale	This is an accounting principle. Errors should be corrected through offsetting entries as opposed to deletions.

6. After a student has been cleared, a facility for retaining the registration number previously used should be provided. All other transaction records may then be deleted.

Rationale	The primary rationale is to limit the size of the database whilst maintaining a record of the registration number in case the student returns to the institution.

7. The project should be completed by the 20th of July 2003.

Rationale	This is the working deadline for the handing in of the project to Department of Computing Science.

System Evolution

The system's requirements have been designed to accommodate any system evolution along probable paths. In defining the boundary between that which is probable and that which is not, the following key assumptions have been made

1. Hardware
   i. The system will remain on an IBM compatible personal computer. This is a critical and very big assumption given the availability of Macintosh computers with the administration office.
   ii. The levy office will not attempt to change to a printer without a capability for printing on A4 sized paper.
   iii. The system will continue to operate in a single user environment.
2. Software
   i. The system will continue to run on The Microsoft Windows platform.
   ii. Future version of the Microsoft Windows platform will continue to support the Visual Basic 6.0 runtime libraries.
   iii. Future version of Microsoft Excel will continue to support the Excel 95 format.
3. Users and their needs
   i. The school will not set a policy that will see students within the same class having different required minimum balances on their deposit accounts.
   ii. The user will not require to batch print statements using any criteria other than the class of the student and the nature of the balance i.e. Settled, Accrued or Prepaid.
   iii. The system will not require more than one individual with administrative rights.
   iv. The user will have a basic appreciation of SQL.

Feasibility Study

Technical Feasibility

The system development will be in Microsoft Visual Basic. The developer has 7 years of experience in the language and has accumulated a vast library of code that should facilitate rapid application development and deployment. It is therefore anticipated that the project is technically feasible. For a full justification of the choice of the Development Environment please consult the design documentation.

Economic Feasibility

Being part of an academic research, Mount Pleasant High School will not be required to pay the developer. Also, the target deployment environment is already available and since the system will not require third party software, it is not anticipated that the Mount Pleasant High School will have to make any financial contribution to the project. The primary running cost will be in the form of printing costs in the production of statements of accounts. The impact of this cost is expected to be offset by the savings in the invoice books that will no longer be required. In the event that the offsetting is not complete, a levy may be negotiated with the parents but it is not envisaged that this will become necessary within the foreseeable future.

Social Feasibility

The system will not results in the cutting of jobs yet it will substantial reduce the workload in the Levy Office. It is therefore expected that it will be welcome by the personnel within the Levy Office. The parents are also unlikely to reject a system that will result in substantial improvements in their ability to access transaction records held by the school. The only concern has been with ability of unscrupulous students to forge the statements of accounts. It has however been concluded that any student who can forge a statement authenticated with a school stamp will not have any difficult in forging a Levy Office invoice. The system will therefore not increase the risk associated with the process. With these issues in mind, it is the opinion of the developer that the system should be certified as socially feasible.

Statement of Overall Feasibility

It is the professional opinion of the developer that the system is feasible within the current context and any changes that may occur over the 3-month development period. This opinion is subject to review in the event of unforeseeable developments.

Design
An Overview

Design Philosophy

The system designer will adopt a top-down approach as proposed by Somerville (1995). The only variation to this method will be in the designing of the data structures first. This approach has been adopted because it is the opinion of the designer that the system is data heavy. It should also be noted that this approach does not contradict the Somerville philosophy, as it was actually proposed by Somerville in his critique of his proposed method.

Data modelling will be followed by Architectural Design, which will detail how the system will be broken down into subsystems. The Architectural Design will however not include the subsystem interfaces, as all interfacing will be at component level. The design will be based on structure diagrams.

Component Design will follow the Architectural Design. The design of each component will include details of

- i The rationale behind the component's inclusion in the System.
- ii Position in system hierarchy
    - Parent.
    - Descendants.
- iii Listing of components known to have dependencies on the component.
- iv User and/or programming interface interface.
- v Algorithms for any specialised task.

The design will not include a detailed description of all the algorithms used by the component. This is because such algorithms are often irrelevant by the time the component is actually implemented due to the emergence of new knowledge. Such a philosophy would also constrain creativity during implementation. The guiding philosophy will therefore be to have component designs that specify **what** a component should do as opposed to **how**. This level of specification should be adequate in ensuring elimination of subsystem conflict.

The designs will be tested for completeness and accuracy through walkthroughs. A suitable method for documentation these has however not been found in the literature available to the designer.

Choice of Development Environment

It is the opinion of the designer that given the specialised nature of programming languages, a decision on the development environment must be made before the design so as to ensure that the produced design will be implementable. This will also allow the use of specialised features that are only available in one development environment.

Given the hybrid nature of the target environment, it is acknowledged by the developer that a platform independent development environment would probably be the most natural choice. Such a view would propose Java based development environment. I is however the opinion of the developer that such a choice would seriously risk the technical feasibility of the project. This decision has been arrived at after considering that

i Java is still a relatively new language with few readily available high-level customisable components. This makes the language unsuitable for rapid application development that is a prerequisite for a project being carried out under such a tight development time frame.

ii The designer, who also happens to be the implementer, has little knowledge of the Java development environment. It is unlikely that he could learn the language and come up with an implementable and accurate design within the given development time. The design would also be quite inferior to one in an alternative language as the designer has little knowledge of the features available in the Java system.

iii Java applications are relatively slow and resource hungry due to the fact that they are interpreted instead of being compiled. This problem is exacerbated by the overhead brought on by the Java runtime environment. The primary target deployment unit is a modest Windows 95 machine. This platform is relatively unsuitable for a large Java application.

The C++ development environment that is also available to the developer would adequately address the problems of speed. However, the problem of the unavailability of high-level components would still be a stumbling block. Data modelling is also not supported directly within C++ making it unsuitable for a data heavy application that required rapid application development.

This leaves the Visual Basic as a possible development environment. Whilst the developer is quick to acknowledge his bias in favour of Visual Basic being his 'native' environment, the decision on the development is one that has been made after consideration of the facts on hand. The following reason have been considered and found to be in favour of a Visual Basic development environment

i The availability of high-level user interface components that conform to the Windows look and feel that the personnel in the Levy Office are familiar with. This is crucial if the project is to be Operationally feasible given the short training period available. It also aids rapid application development.

ii The possibility of code reuse using the implementer's vast library of Visual Basic code. This would contribute towards rapid application development.

The use of tried and tested code would also minimise implementation errors.

iii Visual Basic supports integrated database development that is suitable for the data heavy applications.

iv To increase productivity of the application, it must be in a position to communicate and if possible, integrate with other applications that may provide useful and advanced facilities. For example, requirement R9 states that the system must be able to export data to Excel for Windows 95. Visual Basic has built in facilities for integration with Microsoft Office applications.

v The available version of Visual Basic, VB 6.0, represents a levelling off of the development environment. Most of the features have reached their maturing stage and can safely be assumed to be not subject to further correction unless absolutely necessary. This is important since the backward compatible versions of runtime libraries that are often supplied with improvements in a given language tend to be inadequately compatible with the primary versions.

Choice of DBMS

Having decided on the programming language, the next step is to decide on the database management system. Visual Basic 6.0 has three database development libraries available namely

i Data Access Objects (DAO)
ii Remote Data Objects (RDO)
iii ActiveX Data Objects (ADO)

RDO is not being considered since it has been superseded by ADO and is provided primarily for backward compatibility. The guidance on making a decision between ADO and DOA is provided in the Microsoft Knowledge Base Article 225048 - INFO: Issues Migrating from DAO/Jet to ADO/Jet. According to this article, the decision on the library to use should be based on the extend to which the target operating environment is distributed. The designer has decided to use DAO for because of the following reasons

i The intended environment is not distributed. In the event that it does become distributed, it is unlikely that there will be more than 3 clients. Even in the unlikely event that the system does become distributed; the traffic will remain concentrated on the Levy Office client.

ii The OLE DB provider for ADO has limited functionality in the Jet environment.

iii Performance tests indicate that DAO can be up to 10 times faster than ADO.

iv Requirement R1b requires pattern matching. DAO has a broader pattern matching mechanism.

Data Modelling

Overview

The system will maintain a single Microsoft Jet 4.0 Database based on the Visual Basic Database Access Objects (DAO) 3.6 library. The entities listed below have been identified along with their associated attributes and are presented in Relational Database Structure Diagram notation. Please note that all the dates will be unique, as they will contain time information. The system will however validate the uniqueness of the receipt numbers entered though it will not use them as primary keys. Please refer to the ER Diagram for the relationships among these entities and to the Data Dictionary for an explanation of the attribute specifications.

1. Students
   (Student ID, Surname, First Names, Sex, Guardian Title, Guardian Surname, Guardian Initials, Guardian Address, Class)

2. Deregistered Students
   (Student ID, Surname, First Names, Date Deregistered, Guardian Title, Guardian Surname, Guardian Initials, Guardian Address)

3. Classes
   (Class, Form Teacher, Minimum Balance)

4. Registered Fees
   (Fee code, Fee Name)

5. Levy Account Debits
   (Transaction Date, Amount, Fee Code, Student ID)

6. Levy Account Credits
   (Transaction Date, Amount, Receipt Number, Student ID)

7. Fees Templates
   (Template Name, Template Data)

8. Caution Fee Account Debits
   (Transaction Date, Reason, Amount, Reference Code, Student ID)

9. Caution Fee Account Credits
   (Transaction Date, Amount, Receipt Number, Student ID)

10. Users
    (User Name, User Password, User Rights)

Relationships Among Entities

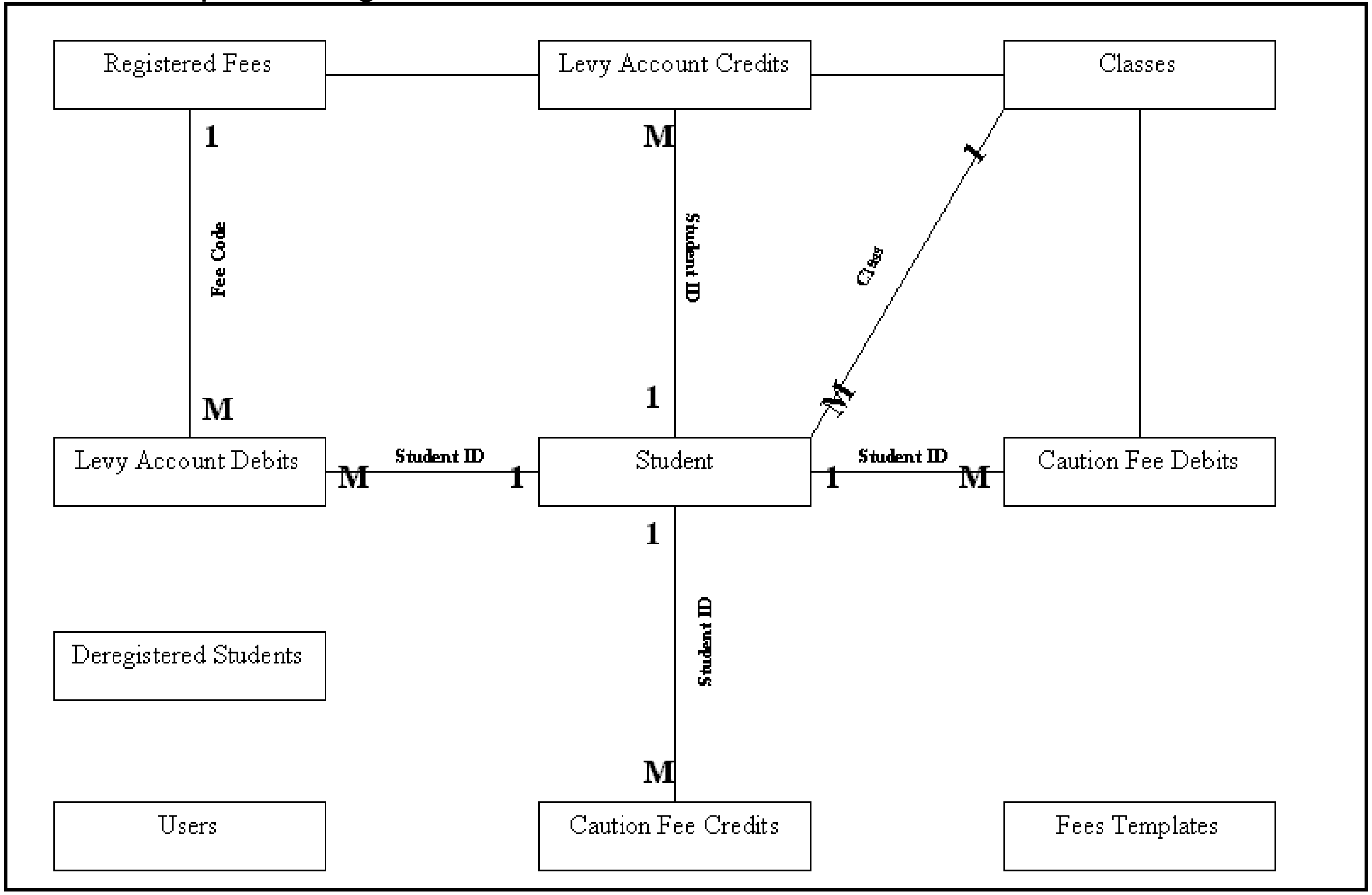
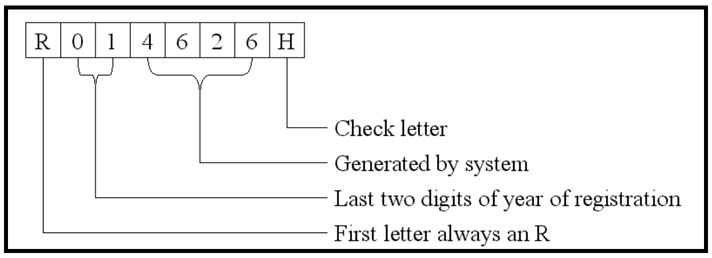


Figure 3: Relationship Among Entities

Data Modelling Dictionary

Field Name	Type	Remarks
Amount	Currency	
Class	Text – 6cc	Unique class name.
Date Deregistered	Time	Contains both date and time information.
Fee Code	Integer	Indexes fee table. Generated by system.
Fee Name	Text – 30cc	Official title of a Levy Office fee as given on the approving circular.
First Names	Text – 30cc	
Form Teacher	Text – 50cc	Full name of the teacher who maintains the class list for the given class.
Guardian's Address	Text – 150cc	The contact address of the guardian. May contain several lines of text.
Guardian's Initials	Text – 6cc	The guardian's initials as the should be printed e.g. J.K.
Guardian's Surname	Text – 15cc	
Guardian's Telephone Number	Text – 20cc	
Guardian's Title	Text – 6cc	The guardian's official title e.g. Prof
Minimum Balance	Currency	Class' minimum balance for the Caution Fee Deposit Account.
Reason	Text – 30cc	Brief description of reason for debit. e.g. Lost text book
Receipt Number	Text – 15cc	
Reference Code	Text – 15cc	Reference code of letter from disciplinary committee authorising debit.
Sex	Text – 1cc	Stores the sex of the given student in the following format [F | M] F – Female; M – Male
Student ID	Text – 8cc	Unique student identification code structured as follows.  R 0 1 4 6 2 6 H Check letter Generated by system Last two digits of year of registration First letter always an R **Figure 4: Student ID Format**
Surname	Text – 20cc	
Template Data	Memo	A string of fee codes and amounts in the format {Fee Code, Fee Amount;}

Field Name	Type	Remarks
Template Name	Text – 30cc	A unique descriptive name for each template. e.g. F1 & F2 Registration Fees
Transaction Date	Time	Contains time when transaction was transacted. Includes date information.
User Name	Text – 30cc	Must be unique
User Password	Text – 20cc	Stored in encrypted form
User Rights	Text – 30cc	A string in which each character is either a one or a zero with the following meaning 1 – Access to facility approved 2 – Access to facility **not** approved

Architectural Design

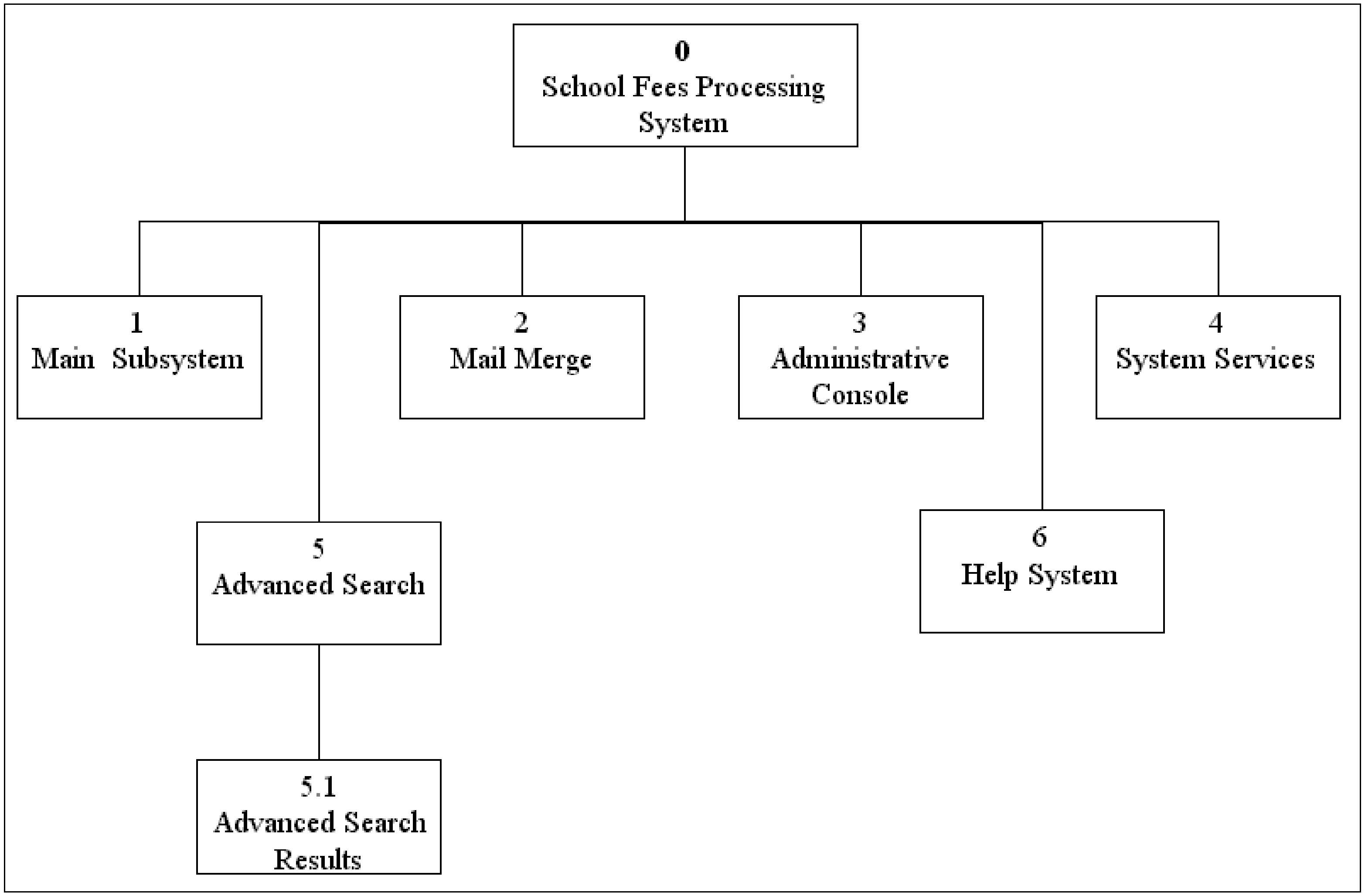


Figure 5: S0 Level 0

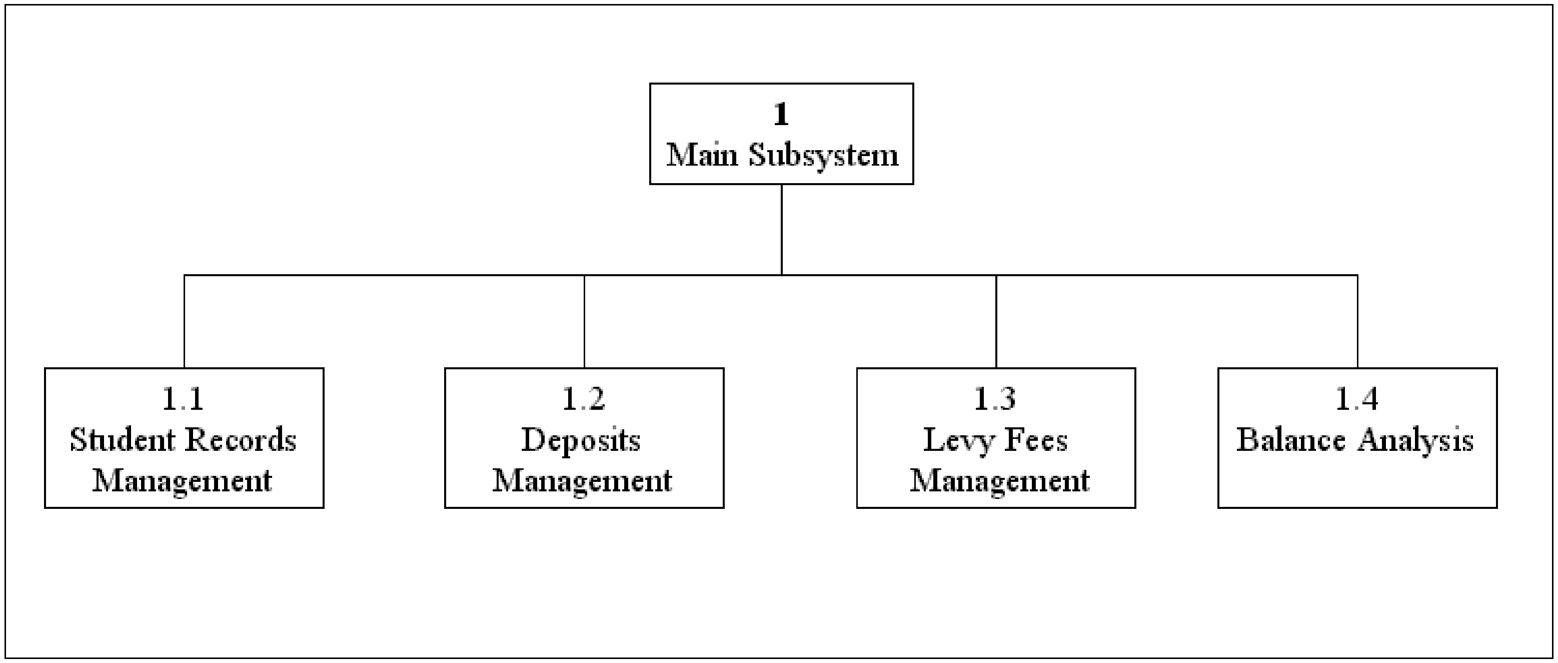


Figure 6: S1 Main Subsystem

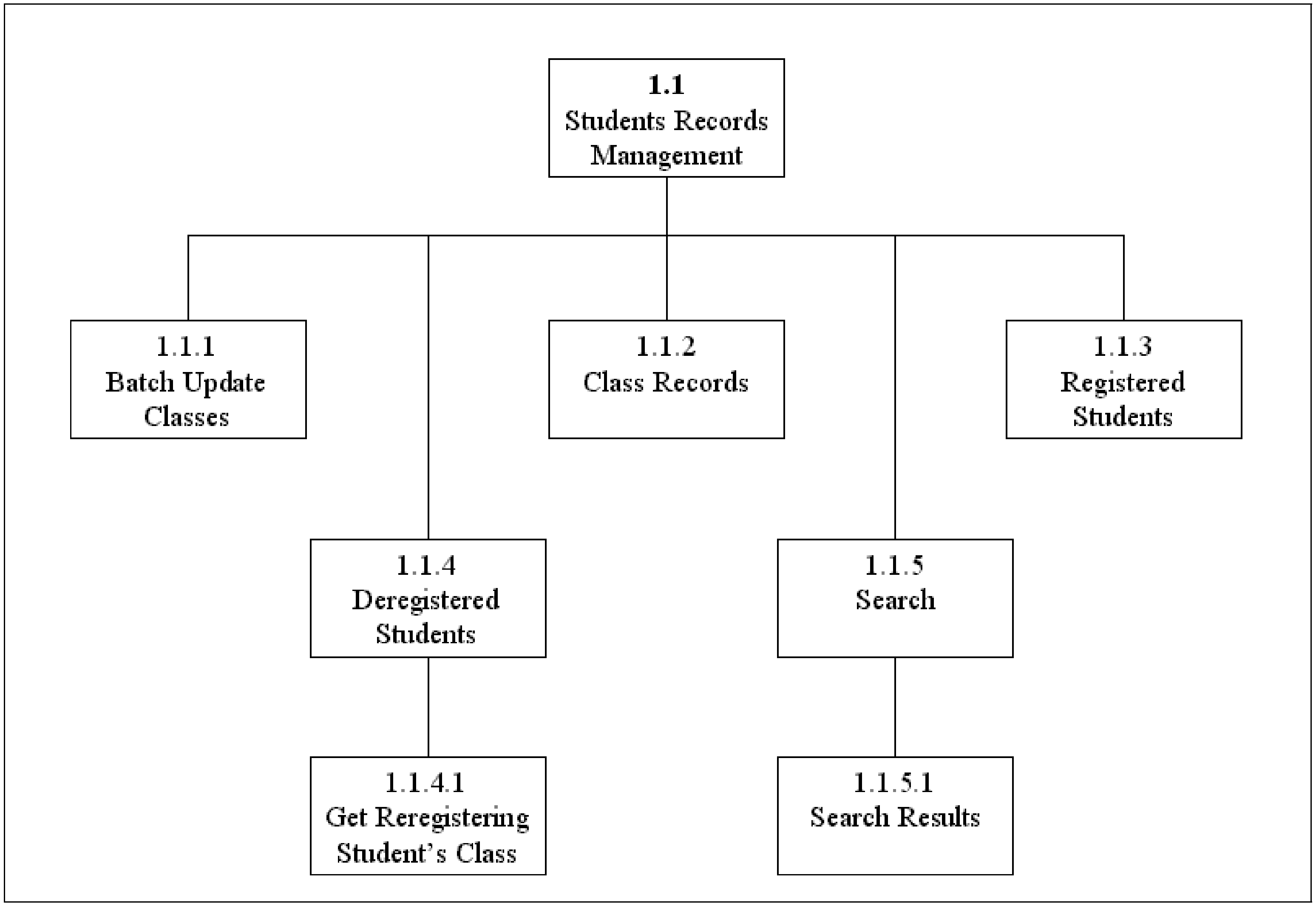


Figure 7: S1.1 - Students Records Management

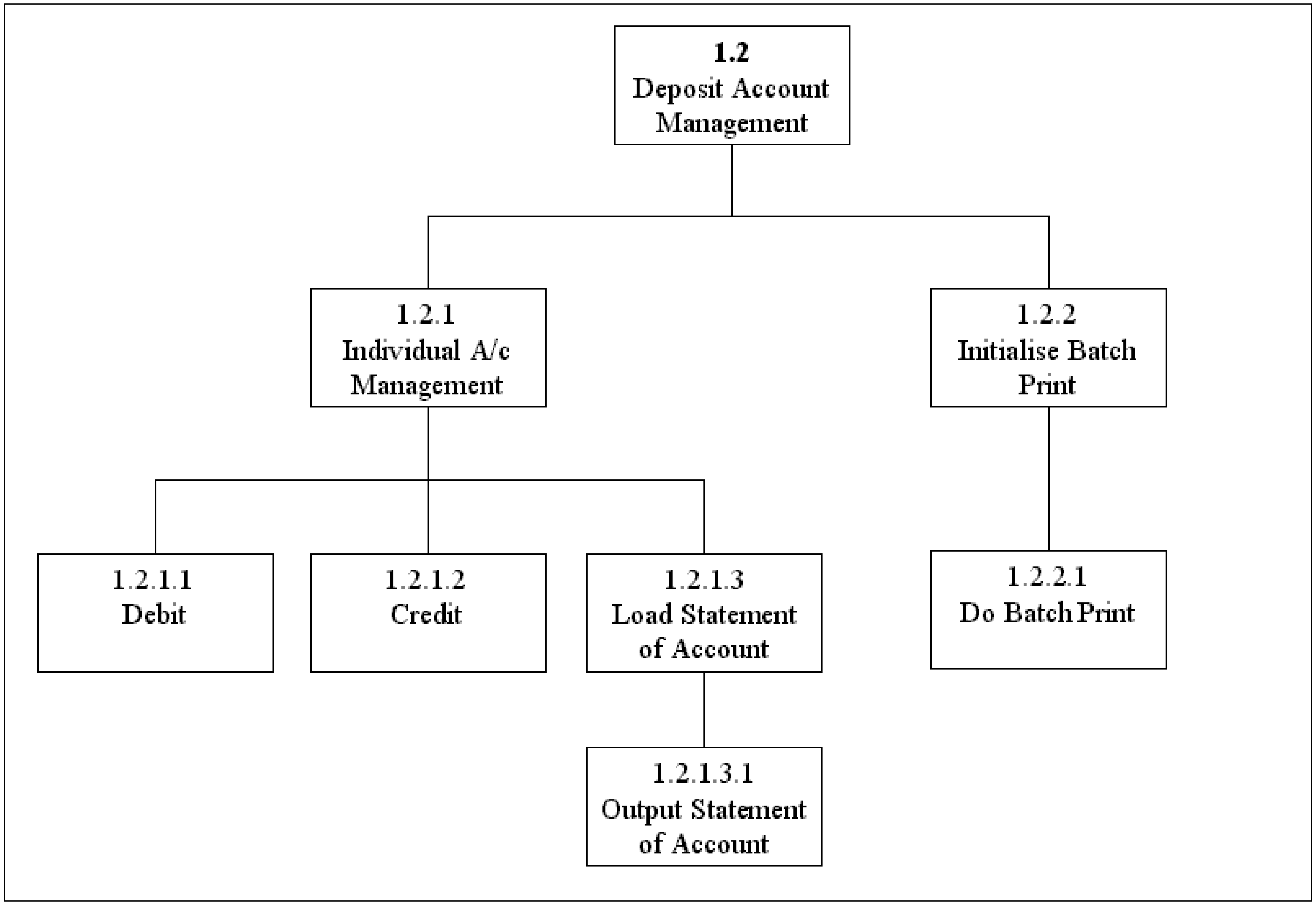

1.2
Deposit Account Management
1.2.1
Individual A/c Management
1.2.2
Initialise Batch Print
1.2.1.1
Debit
1.2.1.2
Credit
1.2.1.3
Load Statement of Account
1.2.2.1
Do Batch Print
1.2.1.3.1
Output Statement of Account

Figure 8: S1.2 Deposit Accounts Management

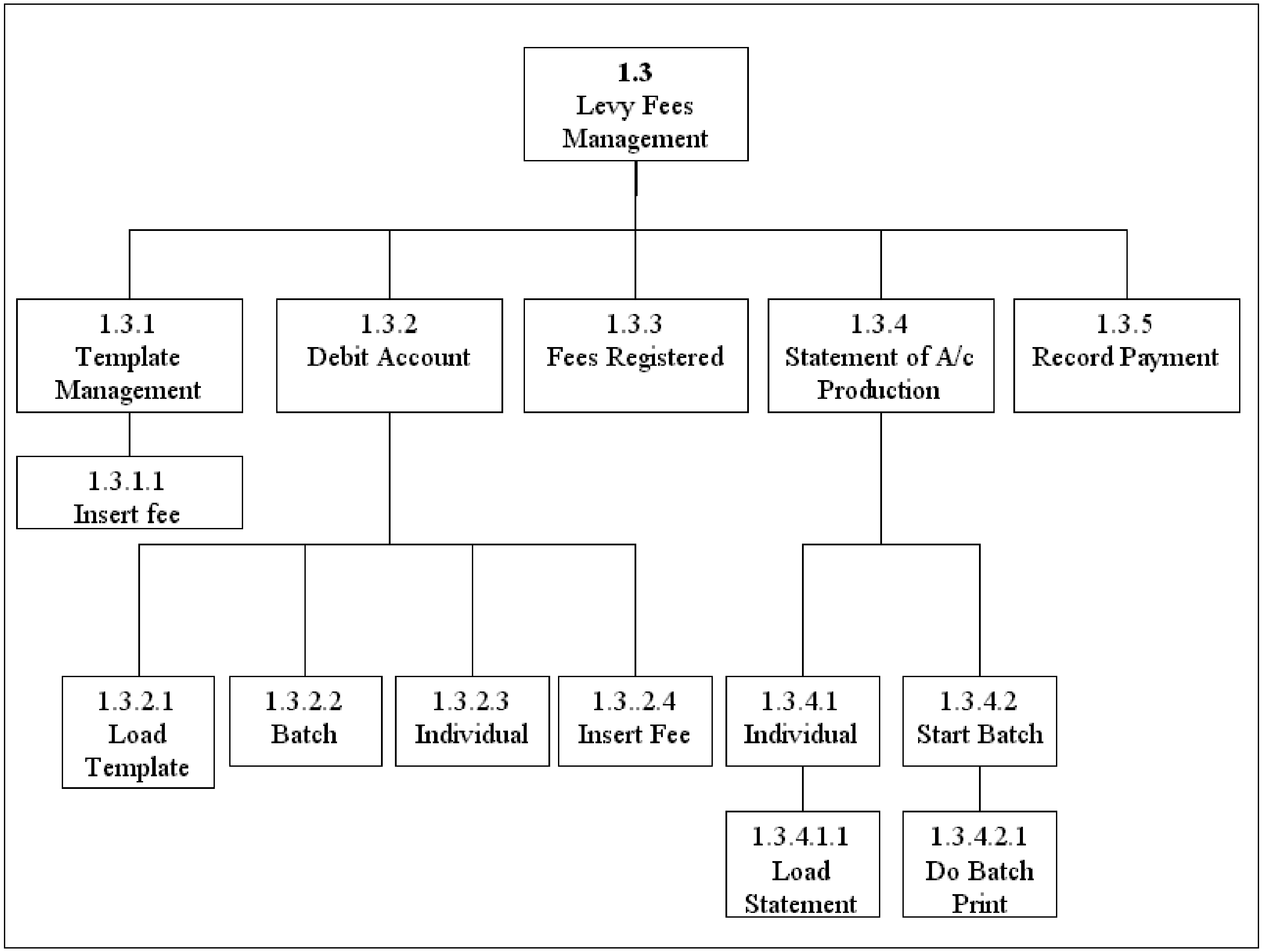


Figure 9: S1.3 - Levy Fees Management

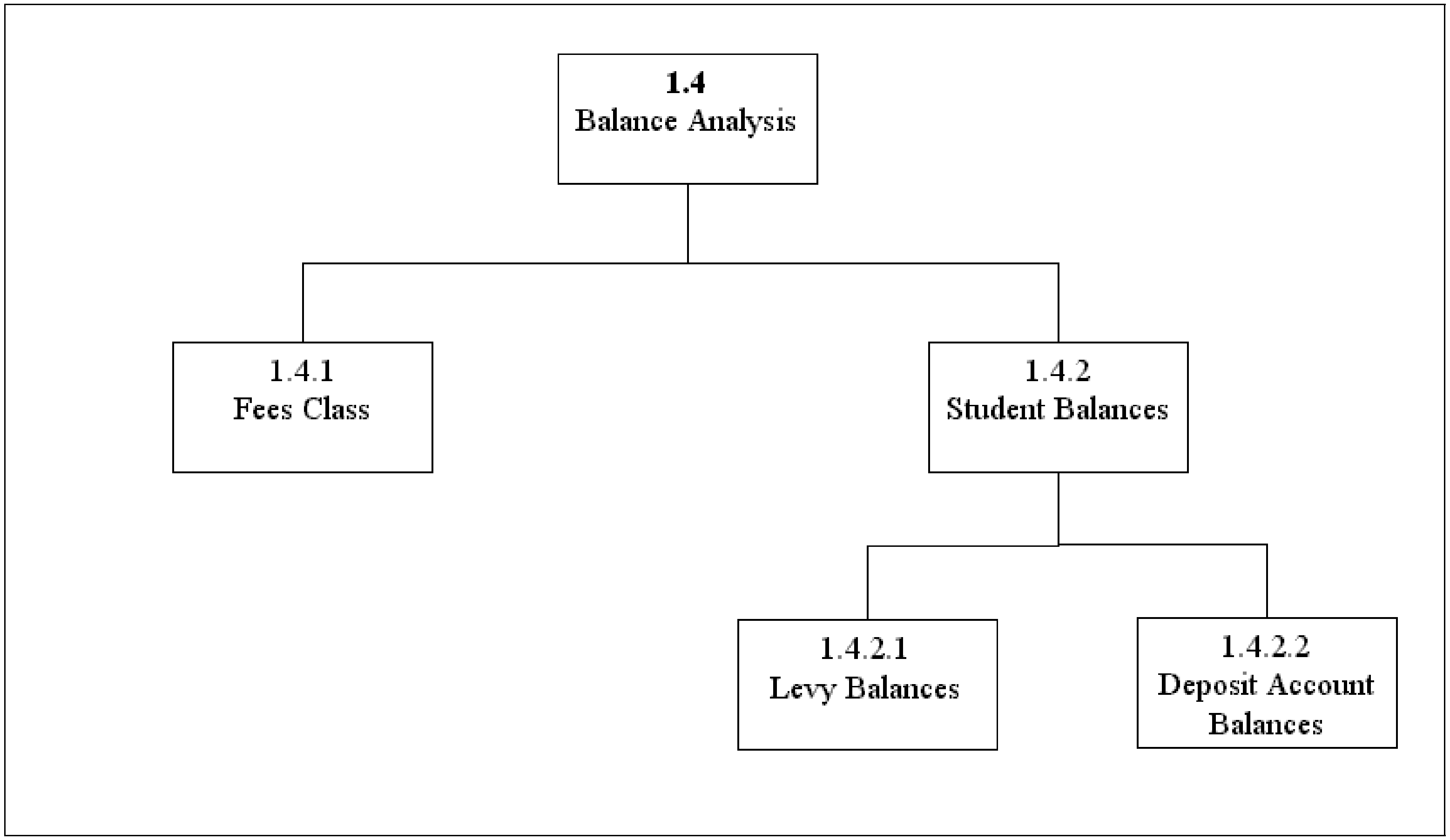


Figure 10: S1.4 Balance Analysis

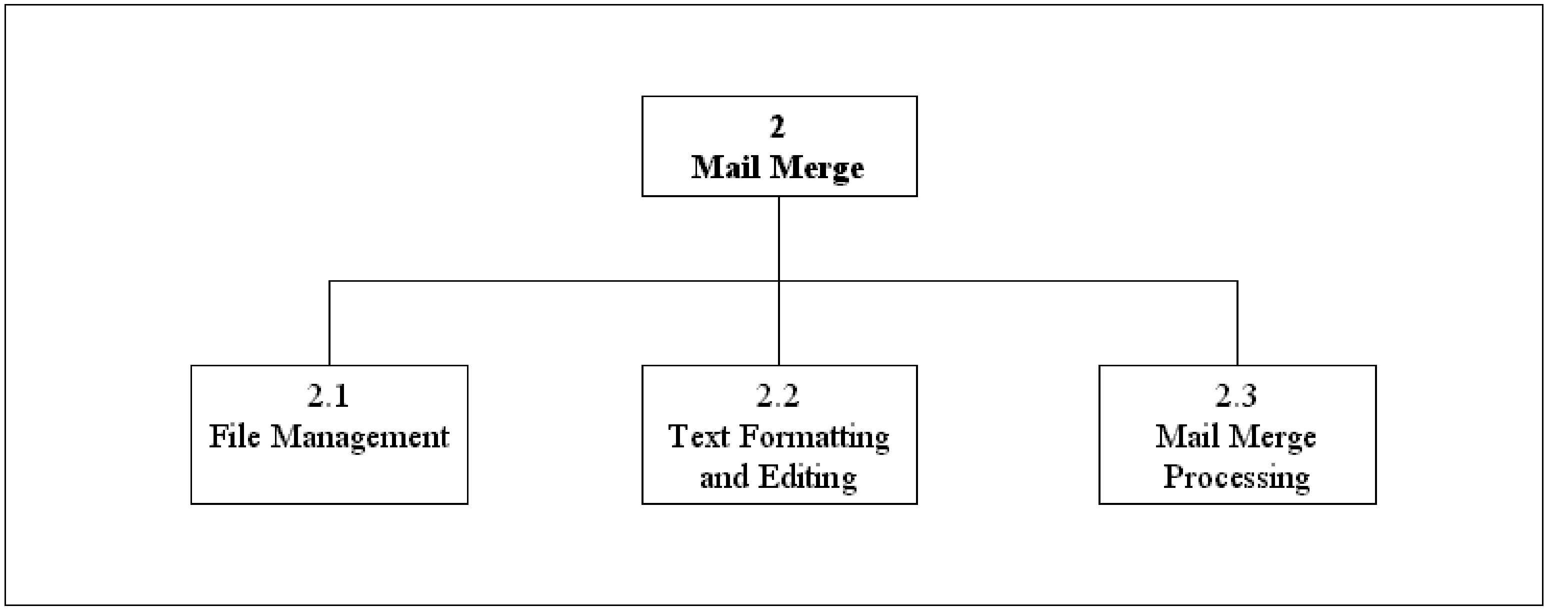


Figure 11: S2 Mail Merge

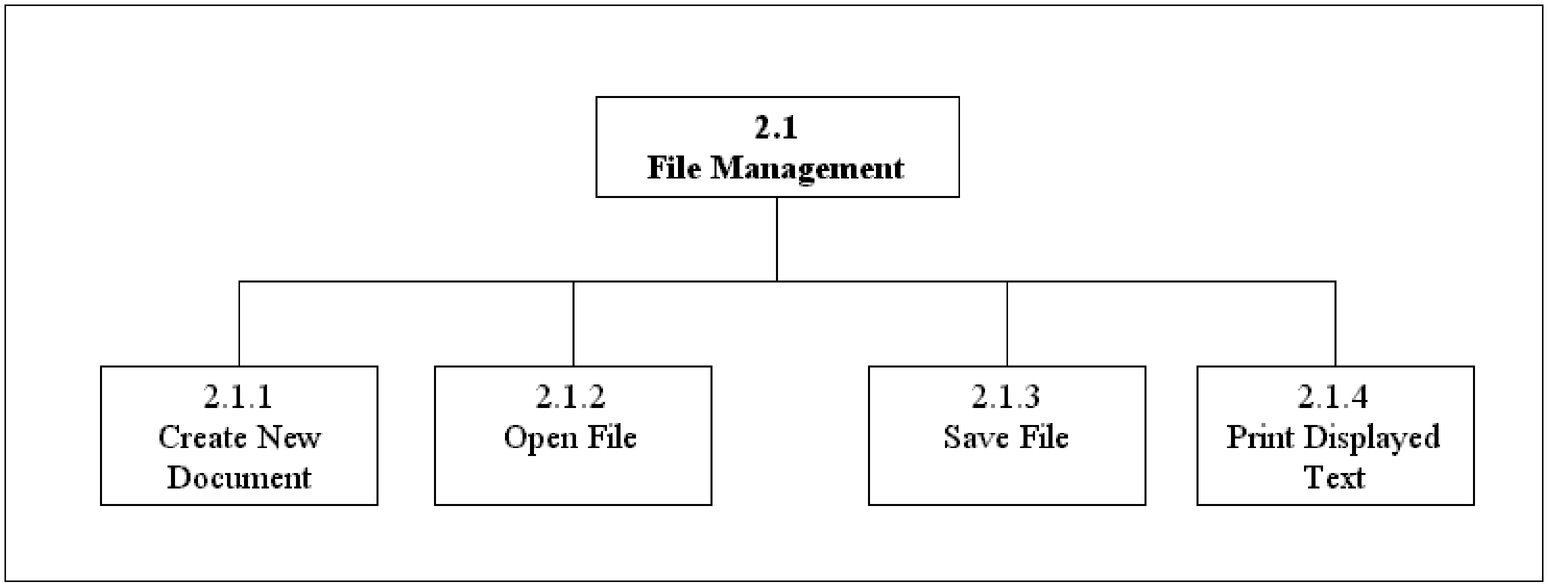


Figure 12: S2.1 File Management

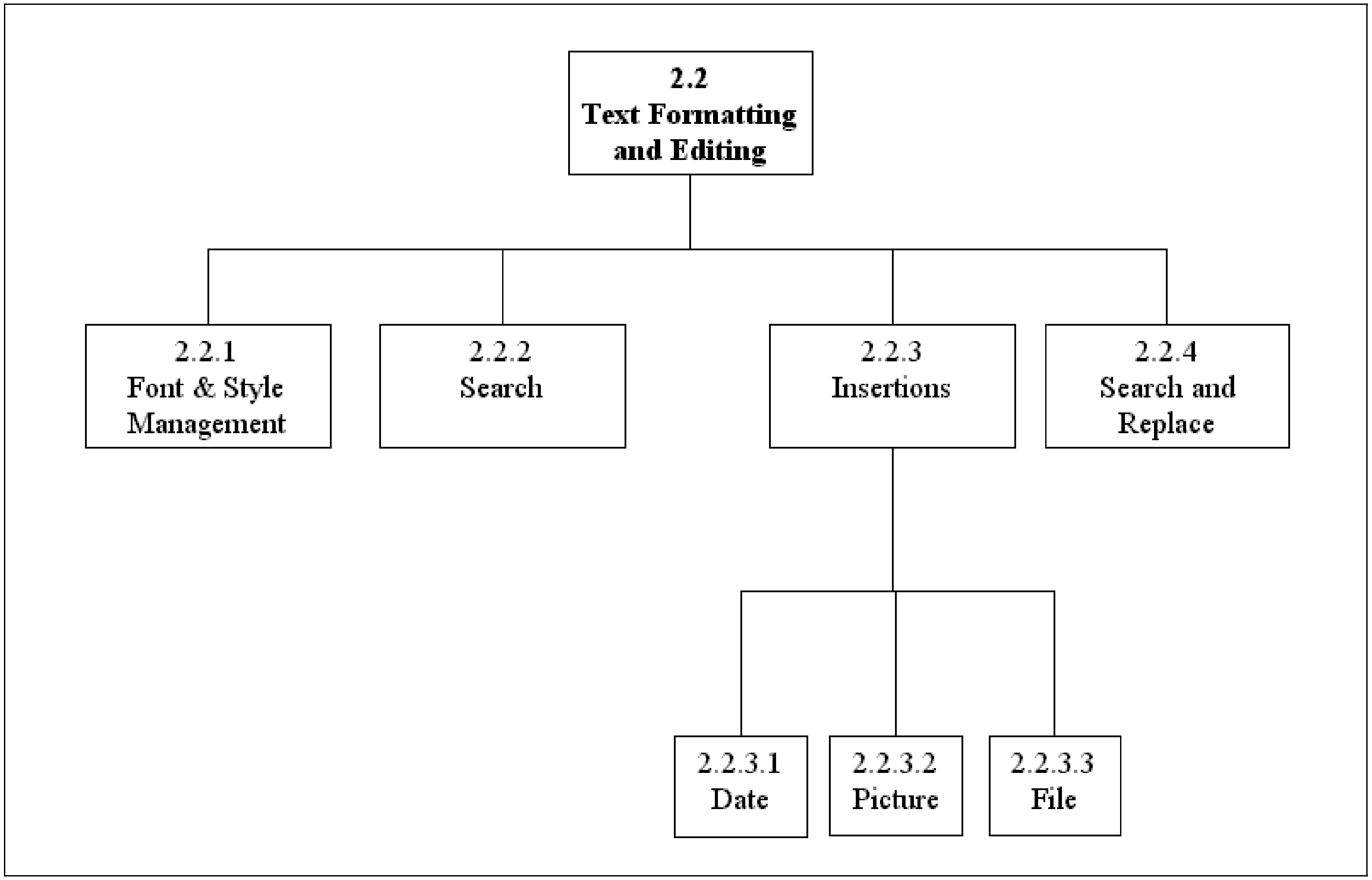


Figure 13: S2.2 Text Formatting and Editing

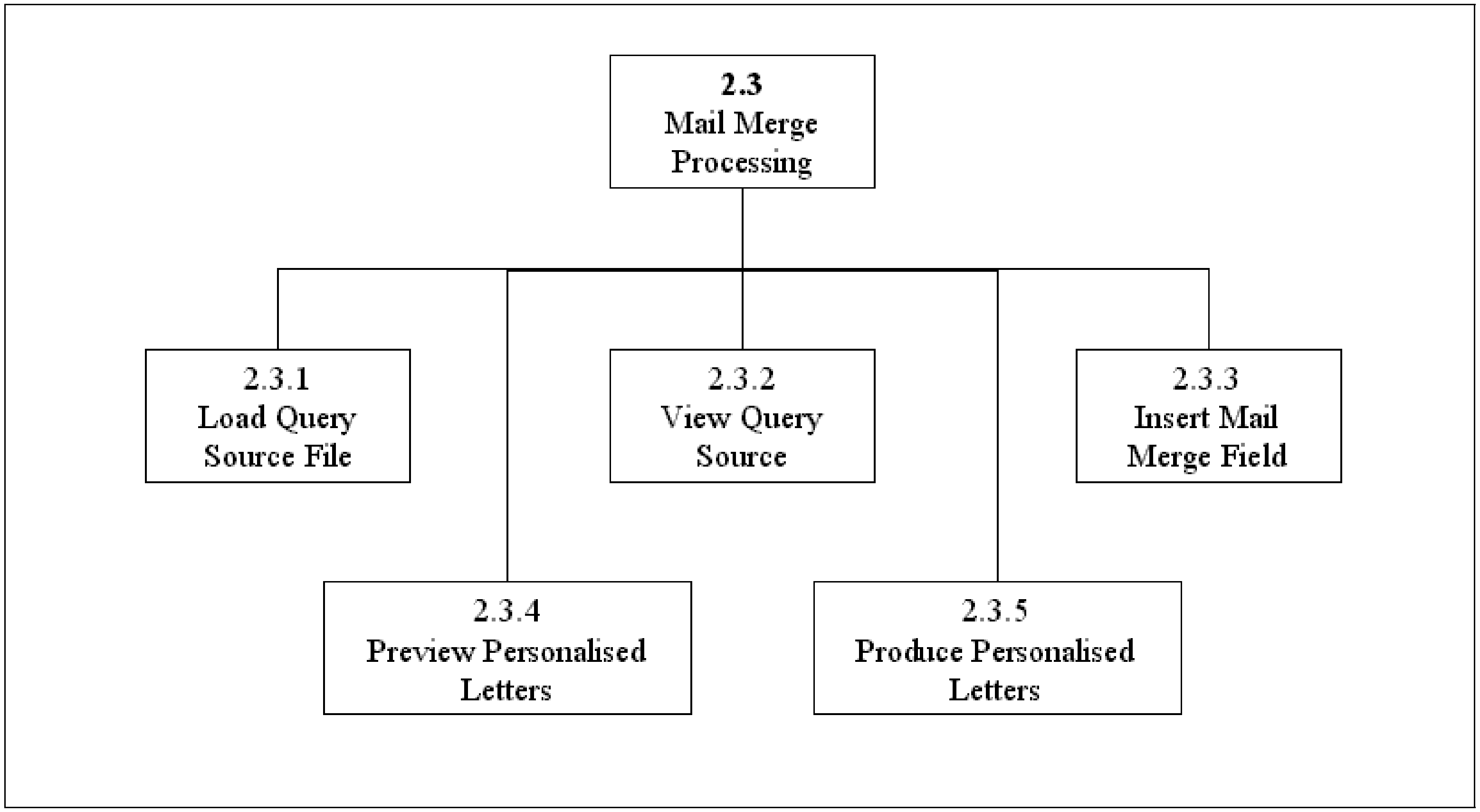


Figure 14: S2.3 Mail Merge Processing

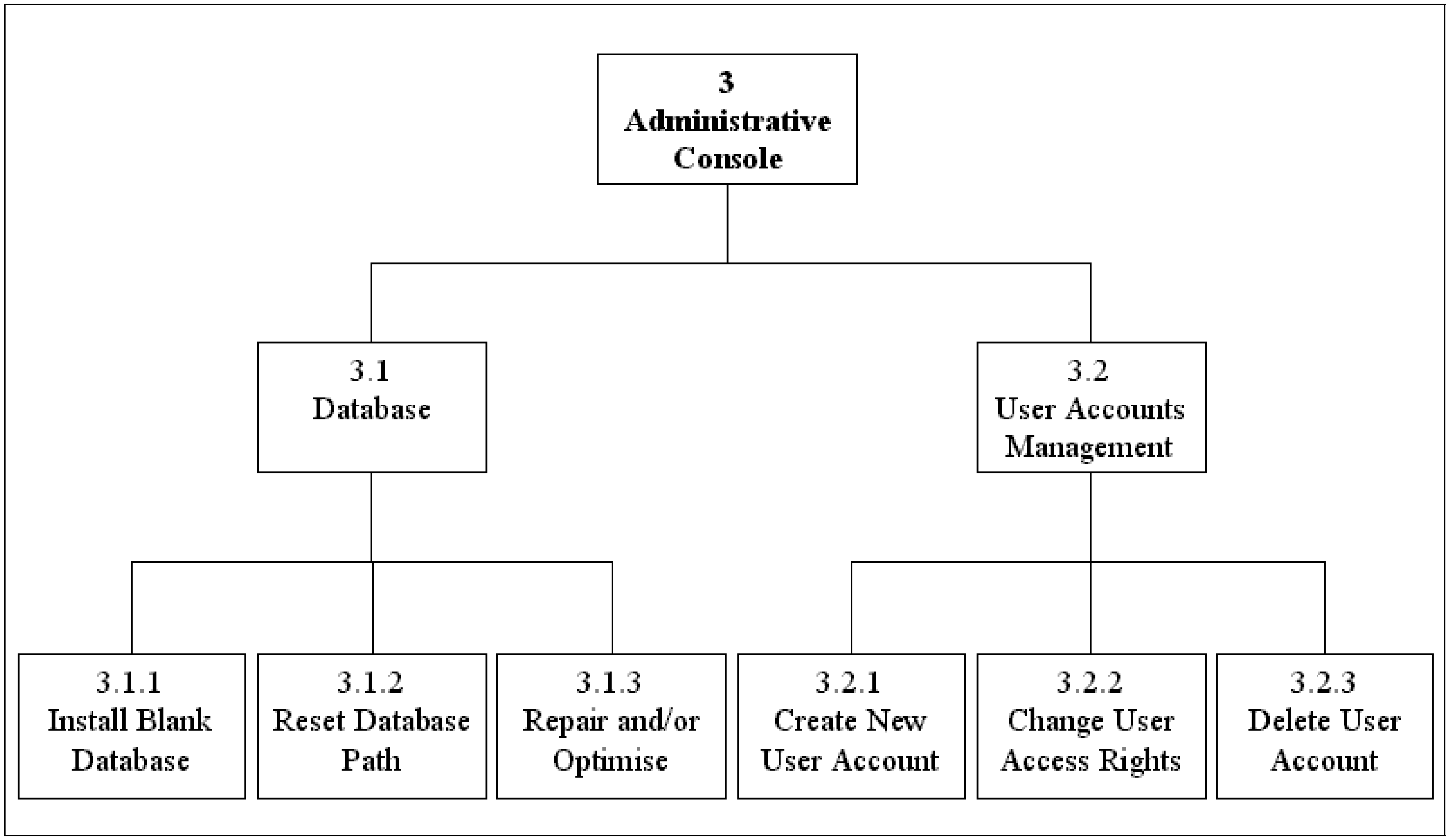


Figure 15: S3 Administrative Console

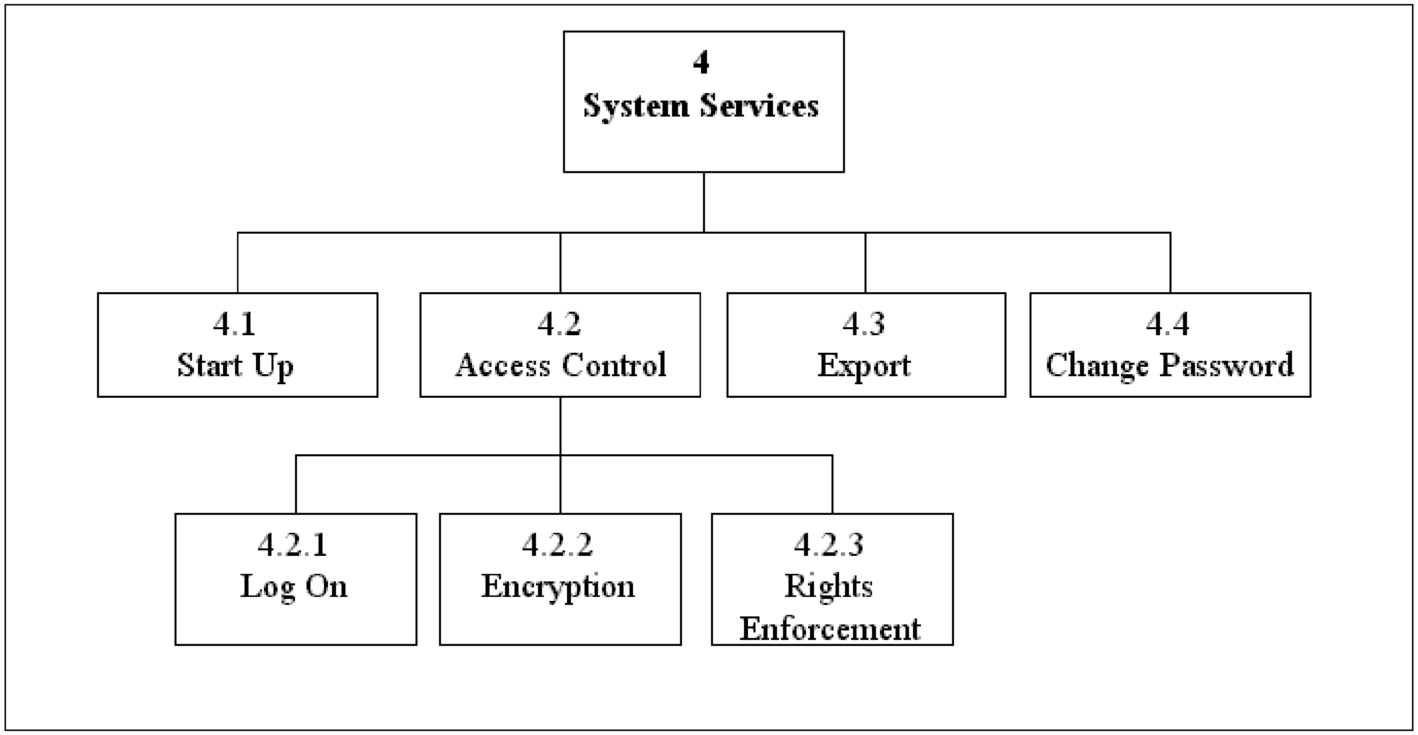


Figure 16: S4 System Services

Component Design

C0 – School Fees Processing System

Rationale
Abstract top-level object to represent program group in Operating System.

Parent
Operating System

Descendants

C1	-	Main Subsystem
C2	-	Mail Merge
C3	-	Administrative Console
C4	-	System Services
C5	-	Advanced Search
C6	-	Help System

Data Sources Accessed And Modified
None

User Interface
Program Group as defined by Operating System. System Services not represented on program group

C1 – Main Subsystem

Rationale
Provides a logical grouping and access to the most commonly used components.

Parent

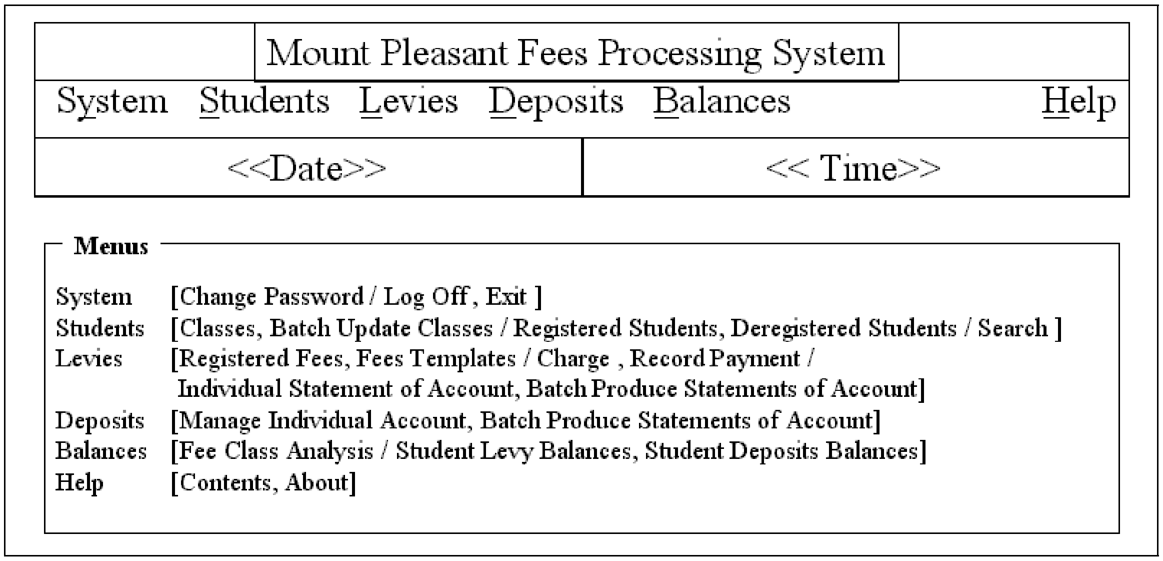
C0	-	School Fees Processing System

Descendants

C1.1	-	Students Records Management
C1.2	-	Deposits Management
C1.3	-	Levy Fees Management
C1.4	-	Balance Analysis

Data Sources Accessed And Modified
<None>

User Interface



System	[Change Password / Log Off, Exit]
Students	[Classes, Batch Update Classes / Registered Students, Deregistered Students / Search]
Levies	[Registered Fees, Fees Templates / Charge , Record Payment / Individual Statement of Account, Batch Produce Statements of Account]
Deposits	[Manage Individual Account, Batch Produce Statements of Account]
Balances	[Fee Class Analysis / Student Levy Balances, Student Deposits Balances]
Help	[Contents, About]

C1.1 Student Records Management

Rationale
Logical grouping of components related to student records management.

Parent

C1	Main Subsystem

Descendants

C1.1.1 -	Batch Update Classes
C1.1.2 -	Class Records
C1.1.3 -	Registered Students
C1.1.4 -	Deregistered Students
C1.1.5 -	Search

Data Sources Accessed And Modified
<<None>>

User Interface
<<None>>

C1.1.1 – Batch Update Classes

Rationale
Requirement R1.d

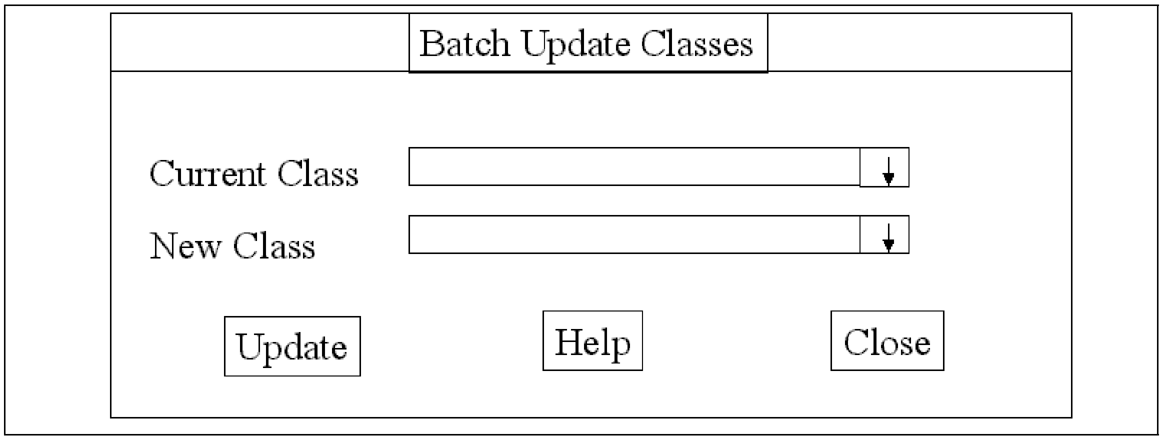
Parent
C1.1 - Students Records Management

Descendants
<<None>>

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes table	No	Used to obtain list of available classes
Students Table	Yes	Modifies the Class column using an Update query

User Interface



Input Name	Input Type	Remarks
Current Class	Drop Down List	• Must contain a list of all available classes
New Class	Drop Down List	• Must contain a list of all the available classes
Update	Command Button	• Should Ensure that Current and New Class have been specified and are not the same • Runs update query moving all the students in “Current Class” to “New Class”
Help	Command Button	• Calls context sensitive help
Close	Command Button	• Unloads form

C1.1.2 Class Records

Rationale
Requirement 1.e

Parent
C1.1 - Student Records Management

Descendants
<<None>>

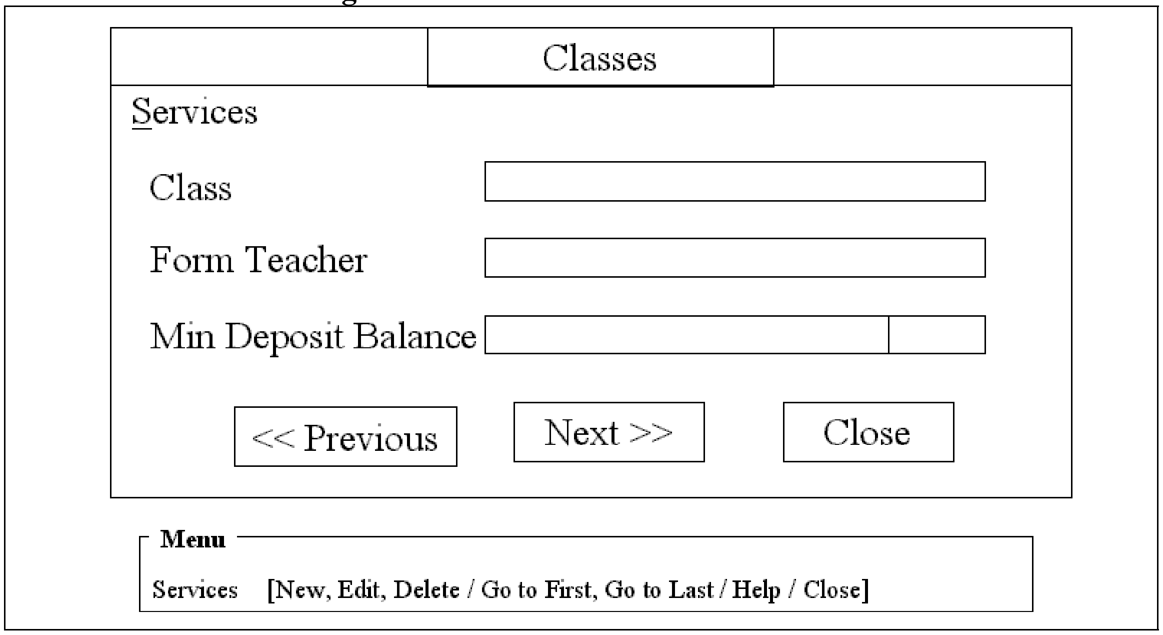
Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes Table	Yes	Read, Edit, Add New and Delete at record level
Students table	No	Used to calculate the number of references to the target class before it is deleted. Class cannot be deleted if it is still in use.

Remarks
Loaded as modal
The following is a known dependency

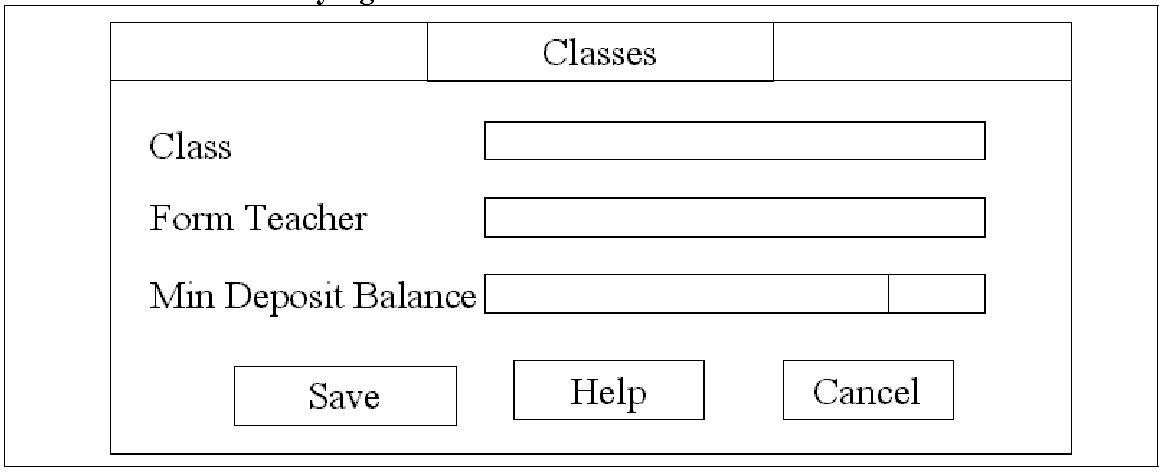
C1.1.5 Search	Used to display list of students in a given class if deletion of class has been denied due to it being still in use.

User Interface - Browsing



Input Name	Input Type	Remarks
New	Menu	• Load interface for modifying records
Edit	Menu	• Load interface for modifying records • There must be a current record
Delete	Menu	• Deletes current record after confirmation. • There must be a current record
Go to first	Menu	• Display First record if present
Go to last	Menu	• Display last record if present
Help	Menu	• Display context sensitive help
Close	Menu	• Unload form
Next	Button	• Display next record if present • If current record is the last, give option to display first record
Previous		• Display previous record if present. • If current record is the first then give option to display the last record.
Close		• Unload form

User Interface – Modifying Records



Input Name	Input Type	Remarks
Class	Text Box	• Maps onto Class field • Maximum input length of 6 characters • Must be unique in table • Must be entered
Form Teacher	Text Box	• Maps onto 'Form Teacher' Field • Maximum input length of 50 characters • Must be entered
Min Deposit Balance	Text Box	• Maps onto 'Minimum balance' field • Must be greater than 0
Save	Button	• Commits changes • Reloads browsing interface
Help	Button	• Loads context sensitive help
Cancel	Button	• Cancels current operation • Reloads browse mode

C1.1.3 Registered Students

Rationale
Requirement 1a

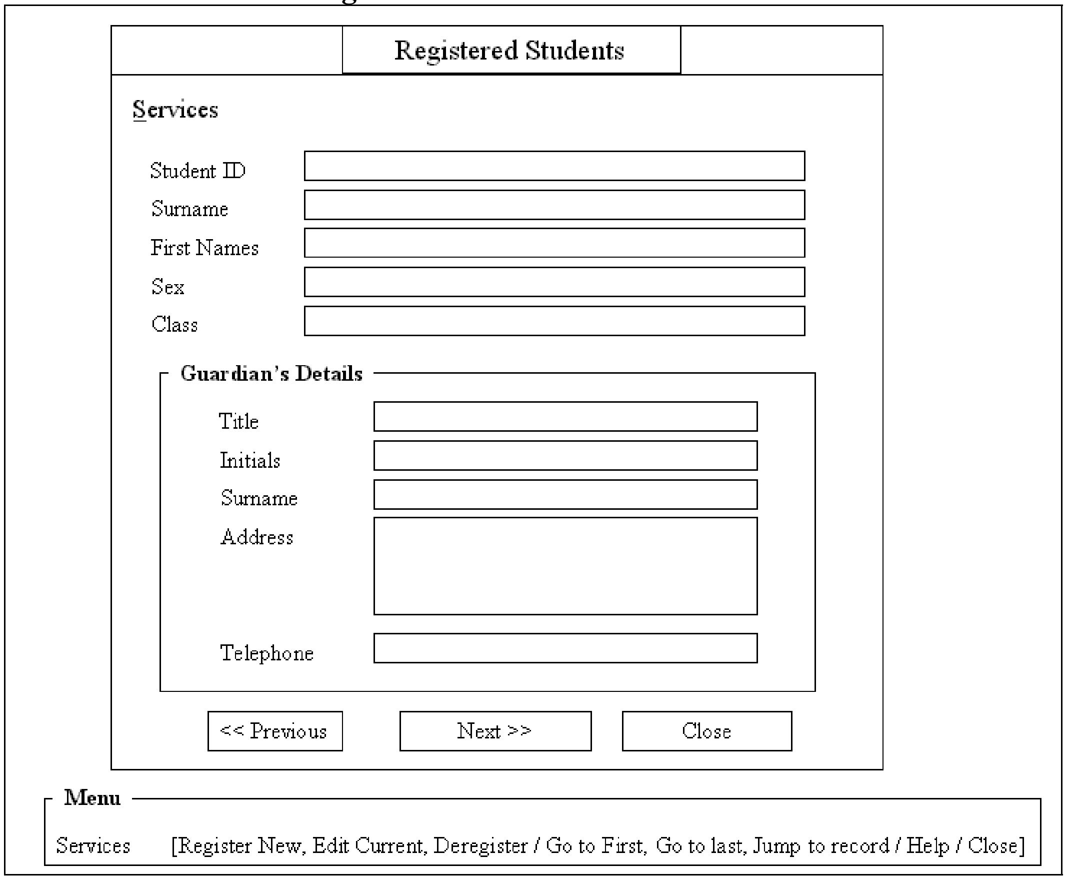
Parent
C1.1 - Students Records Management

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes table	No	Provides list of available classes during record modification
Students Table	Yes	Read, edit, add new and delete at record level.
Deregistered Students table	Yes	Add new record during deregistration.
Caution Fee Account Credits table	No	Used to calculate balance on Caution fee Account during deregistration.
Caution Fee Account Debits table	No	Used to calculate balance on Caution fee Account during deregistration.
Levy Account Credits table	No	Used to calculate balance on Levy fees Account during deregistration.
Levy Account Debits table	No	Used to calculate balance on Levy fees Account during deregistration.

Remarks
1. This component should not be loaded unless there are registered classes on the system
2. Loaded modally

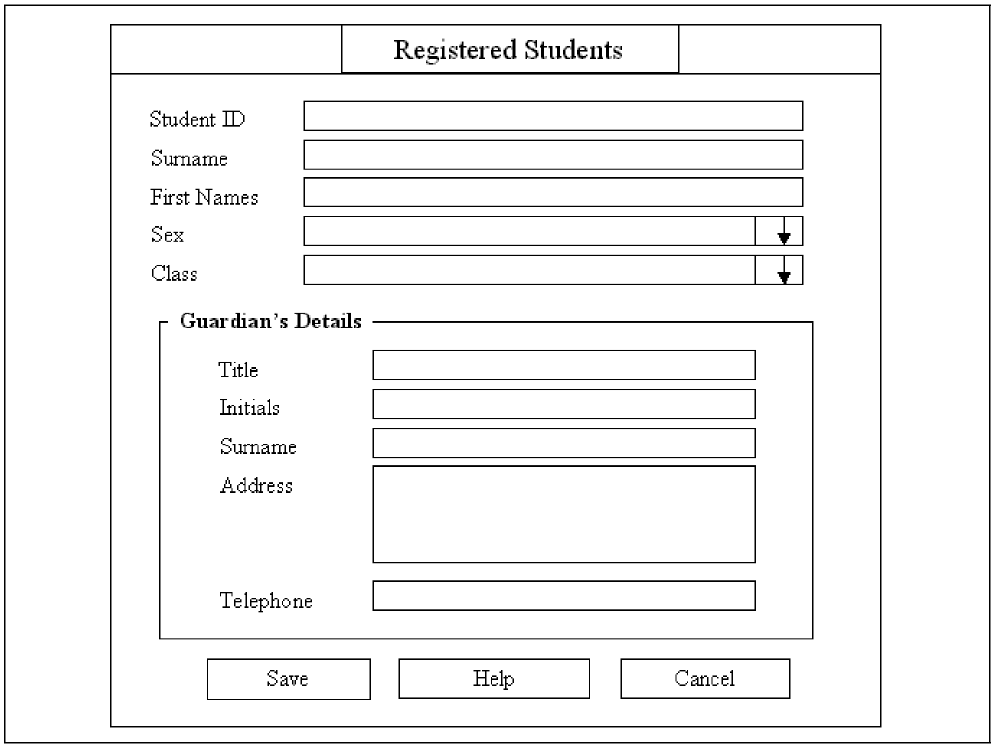
User Interface - Browsing



Input Name	Input Type	Remarks
Register New	Menu	• Load record modification interface.
Edit	Menu	• There must be a current record • Load record modification interface
Deregister	Menu	• There must be current record • Student must have a balance of $0.00 on both the Deposit and Levy accounts. • Action requires confirmation • Delete current record and place it in the Deregistered Students table
Go to First	Menu	• Jump to the first record if present
Go to last	Menu	• Jump to the last record if present
Jump to record	Menu	• Request the Student ID of the record to jump to • Use an Input Box • On no match, report error and maintain record that was being displayed as current.
Help	Menu	• Load context sensitive help
Close	Menu	• Unload form

Input Name	Input Type	Remarks
Next	Button	• Go to the next record if present. • If current record is the last then give option to go to the first record.
Previous	Button	• Go to the previous record if present. • If current record is the first then give option to go to the last record.
Close	Button	• Unload form
Sex	Text Box	• Display as Male/Female
Address	Text Box	• Display as multi-line

User Interface – Record Modification



Input Name	Input Type	Remarks
Student ID	Text box	• Automatically generated as to specification in Data Modelling • Must be 8 characters
Surname	Text box	• Required input • Must range between 1 and 20 characters inclusive
First names	Text Box	• Required input • Must range between 1 and 30 characters.

Input Name	Input Type	Remarks
Sex	Drop Down List	• [Male, Female] • Required input • Store as [M|F]
Class	Drop Down List	• Must list all the available classes • Required input
Guardian's Title	Text box	• Required Input • Maximum length of 6 characters
Guardian's Initials	Text Box	• Required input • Maximum length of 6 characters
Guardian's Surname	Text Box	• Required input • Maximum length of 20 characters
Guardian's Address	Text Box	• Multi-line • Required input • Maximum length of 150 characters
Guardian's Telephone Number	Text Box	• Required input • Maximum length of 20 characters
Save	Button	• Commit changes
Help	Button	• Display context sensitive Help
Cancel	Button	• Abort record modification operation • Reload browsing interface

C1.1.4 Deregistered Students

Rationale
Requirement R1f

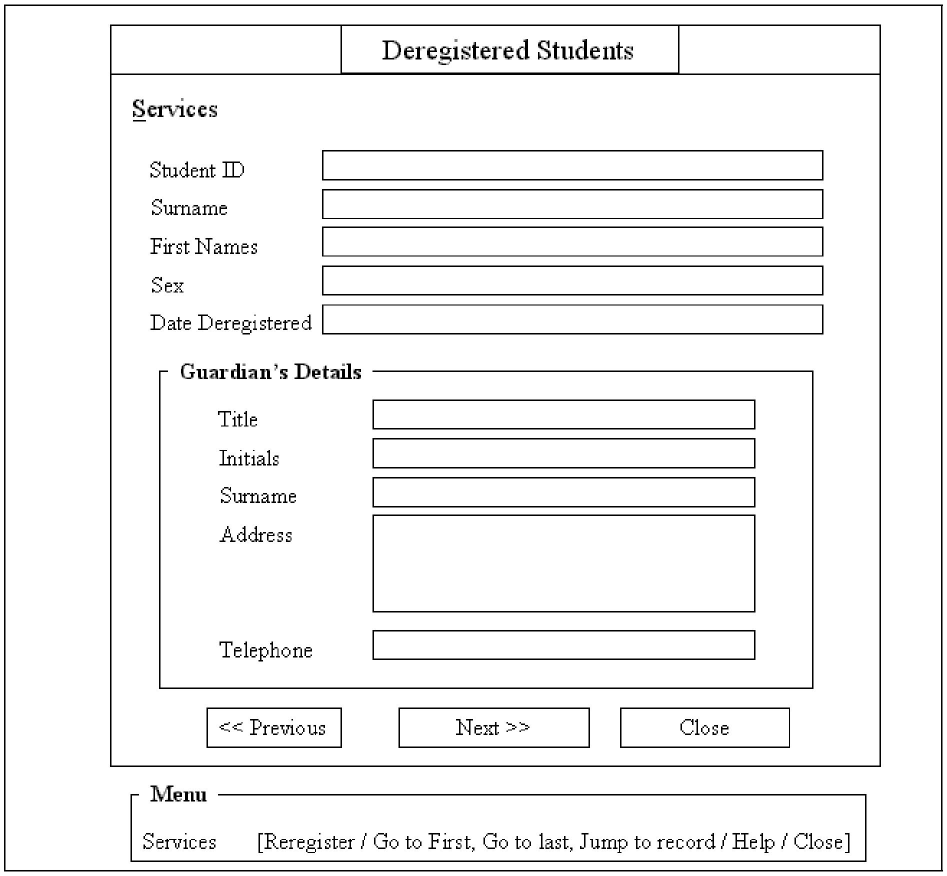
Parent
C1.1 - Student Records Management

Descendants
C1.1.4.1 Get reregistering student's class

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Deregistered Students table	Yes	Read and delete records
Students Table	Yes	Add new

User Interface



Deregistered Students

Services

Student ID
Surname
First Names
Sex
Date Deregistered

Guardian's Details

Title
Initials
Surname
Address
Telephone

<< Previous | Next >> | Close

Menu

Services [Reregister / Go to First, Go to last, Jump to record / Help / Close]

Input Name	Input Type	Remarks
Previous	Button	• Go to previous record if present

		• If current record is first then give option to go to the last record
Next	Button	• Go to the next record if present. • If the current record is the last then give option to go to the first record
Close	Button	• Unload form.
Reregister	Menu	• Get class using component C1.1.4.1 • Move the current record to the Students table
Go to First	Menu	• Jump to the first record if present
Go to Last	Menu	• Jump to the last record if present
Jump to record	Menu	• Request the Student ID of the record to jump to • Use an Input Box • On no match, report error and maintain record that was being displayed as current.
Help	Menu	• Load context sensitive help
Close	Menu	• Unload form

C1.1.4.1 Get Reregistering Student's Class

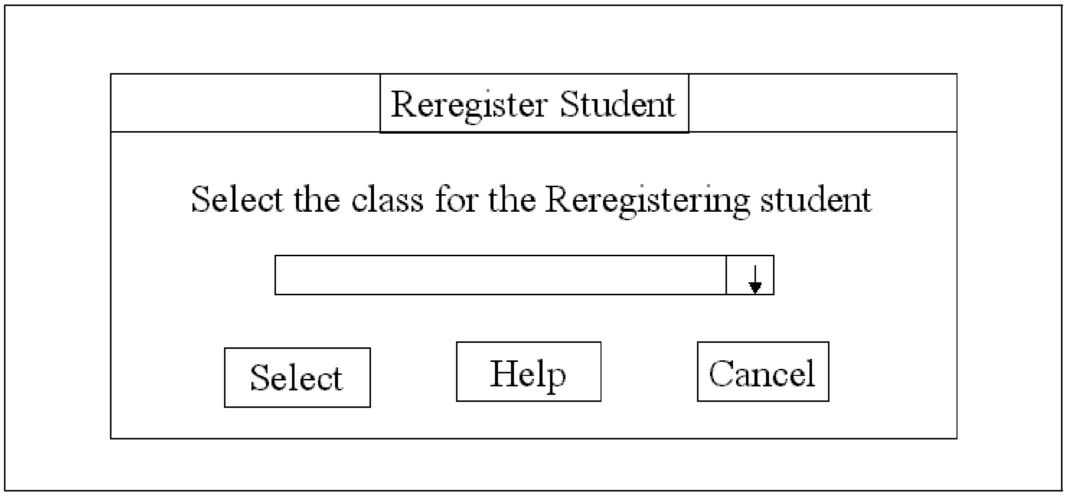
Rationale
To minimise input errors when registering students, it is necessary that the user be given a list of valid inputs to choose from.

Parent
C1.1.4 Deregistered Students

Descendants
<<None>>

Data Sources Accessed And Modified
<<None>

User Interface



Input Name	Input Type	Remarks
Classes	Drop Down List	• Must contain a list of the available classes
Select	Button	• Hide form and leave parent to retrieve selection details
Help	Button	• Load Context sensitive help.
Cancel	Button	• Hide form and ensure that nothing is selected in the Class list

C1.1.5 Search

Rationale
Requirement 1.b

Parent

C1.1	Students Records Management

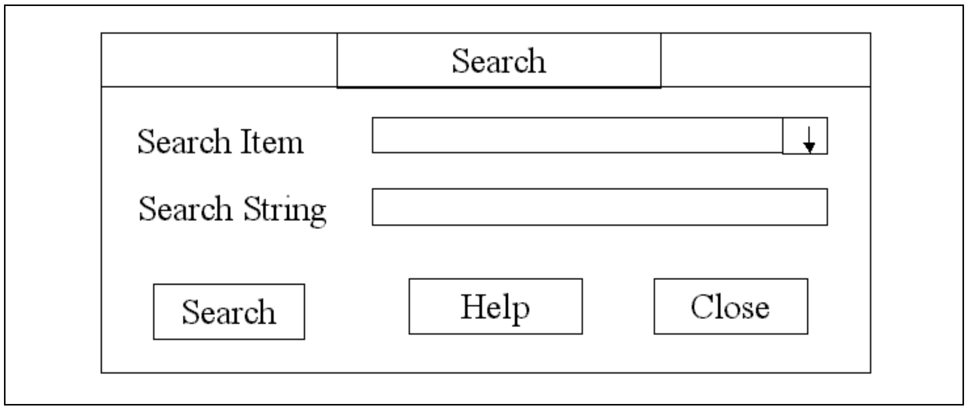
Descendants

C1.1.5.1	Search results

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students table	No	Queried

User Interface



Input Name	Input Type	Remarks
Search Item	Drop Down List	[Student ID, Surname, First names, Class, Guardian's Surname]
Search string	Text Box	String input including wildcard characters.
Search	Button	Build and execute a query and if it returns some records, load Component 1.1.5.1 to display the results.
Help	Button	Load context sensitive help.
Close	Button	Unload form

Remarks
The following are known dependencies

C1.1.2	Class Records	Uses form to load list of students in a given class if deleting it has been denied due to it being in use.
C1.3.2	Debit Levy Accounts	Uses form to show the list of students that will be charged.

C1.1.5.1 Search Results

Rationale
Requirement 1c

Parent

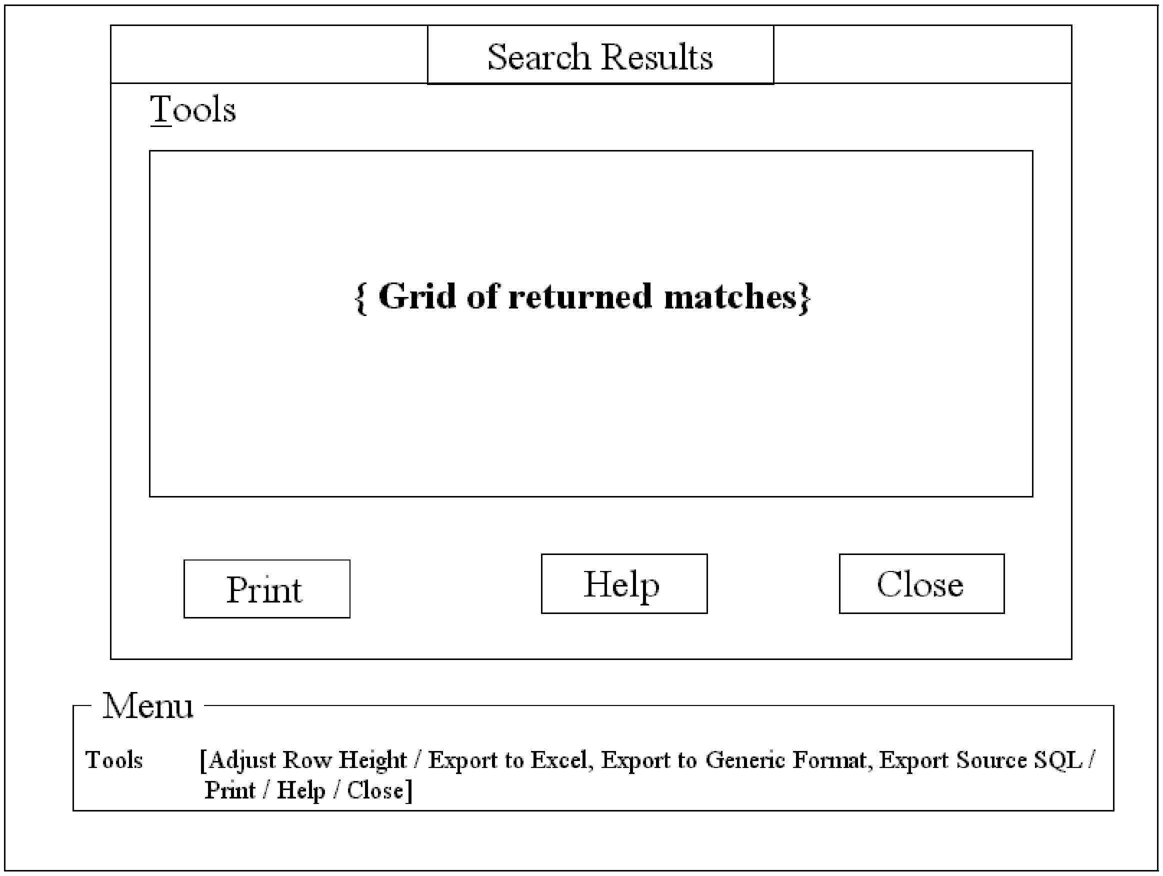
C1.1.5	Search

Descendants
<<None>>

Known Dependencies
<<None>>

Data Sources Accessed And Modified
<<None>>

User Interface


Search Results
Tools
{ Grid of returned matches}
Print
Help
Close
Menu
Tools [Adjust Row Height / Export to Excel, Export to Generic Format, Export Source SQL / Print / Help / Close]

Input Name	Input Type	Remarks
Print	Button	Send search results to printer
Help	Button	Load context sensitive help
Close	Button	Unload search results form
Adjust Row Height	Menu	Use an input box control to get the number of rows of text to display per multi-line field
Export to Excel	Menu	Use component C4.3 to export the search results
Export to generic format	Menu	Use component C4.3 to export the search results
Export Source SQL	Menu	Save the query used to generate the result list to a text file.
Print	Menu	Send search results to printer
Help	Menu	Load context sensitive help
Close	Menu	Unload form

C1.2 Deposit Account Management

Rationale

A logical grouping of the components making up the Deposit Account Management subsystem.

Parent

C1	Main Subsystem

Descendants

C1.2.1	Individual Account Management
C1.2.2	Initialise Batch Printing of Deposit Statements.

Data Sources Accessed And Modified

<<None>>

User Interface

<< None>>

C1.2.1 Individual Deposit Account Management

Rationale

A logical grouping of the components making up the Individual Deposit Account Management subsystem.

Parent

C1.2	Deposit Account Management

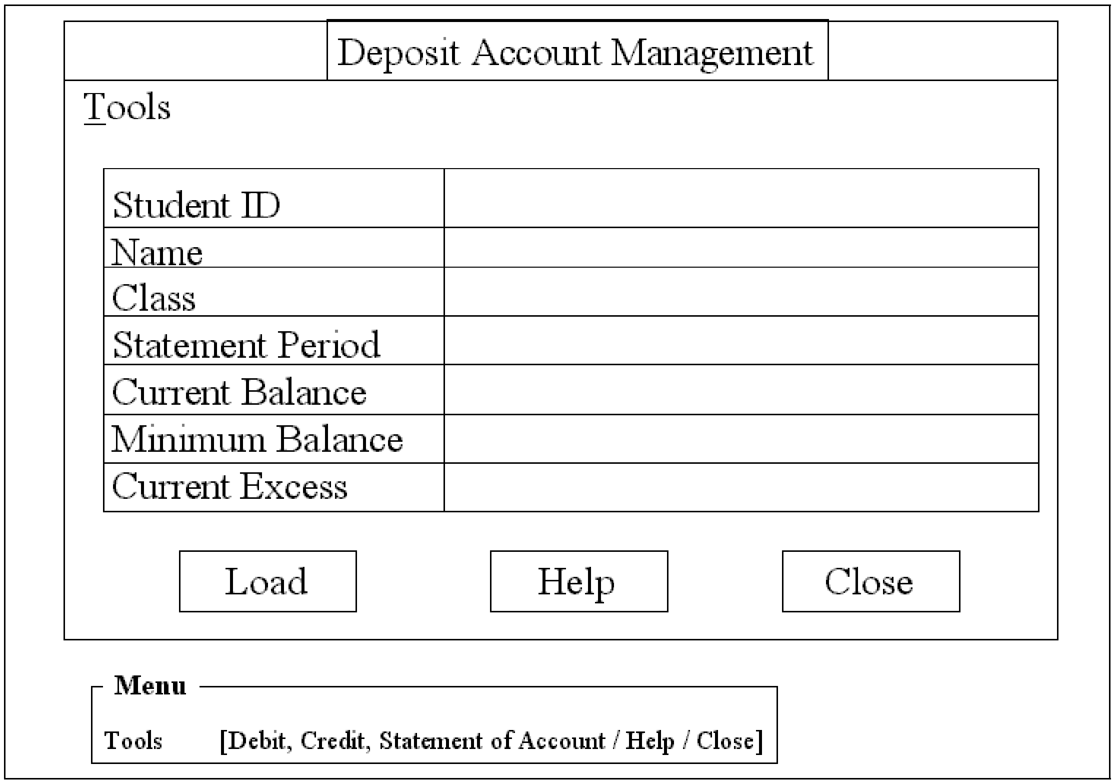
Descendants

C1.2.1.1	Debit Individual Deposit Account
C1.2.1.2	Credit Individual Deposit Account
C1.2.1.3	Individual Deposit Statement of Account

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students table	No	Read to obtain Student's ID, Name and Class during account load.
Caution Fee Account Debits Table	No	Read by query when calculating balance during account load
Caution Fee Account Credits Table	No	Read by query when calculating balance during account load
Classes table	No	Read to obtain class' minimum balance during account load.

User Interface



Input Name	Input Type	Remarks
Debit	Menu	• Launch component C1.2.1.1
Credit	Menu	• Launch component C1.2.1.2
Statement of Account	Menu	• Launch component C1.2.1.3
Help	Menu	• Display context sensitive help
Close	Menu	• Unload form
Load	Button	• Load individual account using input box
Help	Button	• Display context sensitive help
Close	Button	• Unload form

C1.2.1.1 Debit Individual Account

Rationale
Requirement 7c

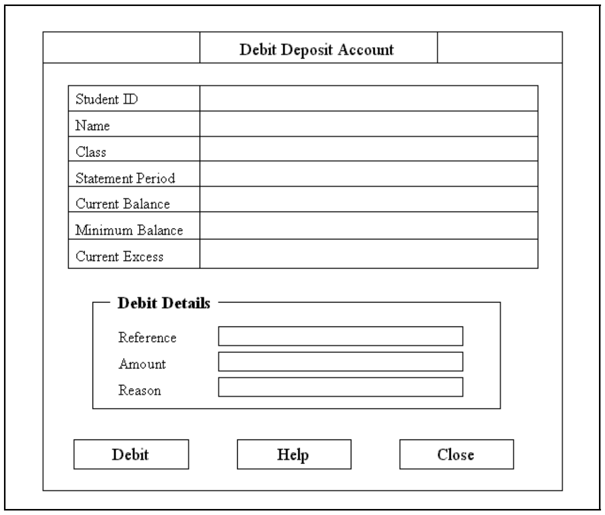
Parent
C1.2.1 Manage Individual Deposit Account

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students Table	No	Used to verify Student ID before committing debit
Caution Fee Account Debits	Yes	New Record added

User Interface



Input Name	Input Type	Remarks
Reference	Text Box	• Required Input • Up to 15 characters
Amount	Text Box	• Required input • Value must be translated into currency format • Must be greater than $0.00
Reason	Text Box	• Required input • Up to 30 characters

C1.2.1.2 Credit Individual Account

Rationale
Requirement 7b

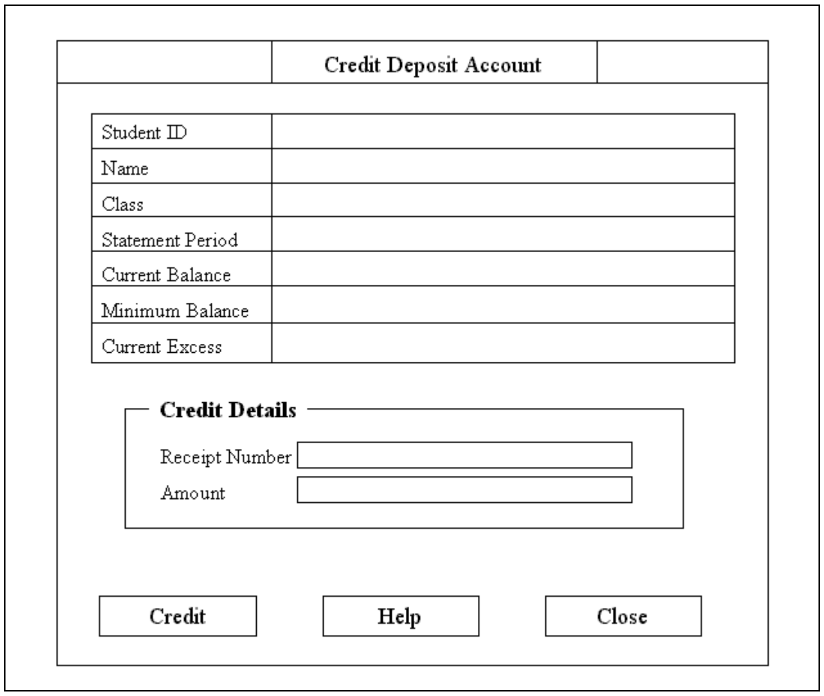
Parent
C1.2.1 Manage Individual Deposit Account

Descendants
<<None>>

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students Table	No	Read to verify that Student ID is valid
Caution Fee Account Credits table	Yes	Add new record

User Interface



Input Name	Input Type	Remarks
Receipt Number	Text Box	• Required input • Maximum length of 15 characters
Amount	Text Box	• Required input • Must be converted to currency • Must be greater than $0.00

C1.2.1.3 Load Individual Deposit Statement

Rationale
Requirement 7d

Parent

C1.2.1	Manage Individual Deposit Account

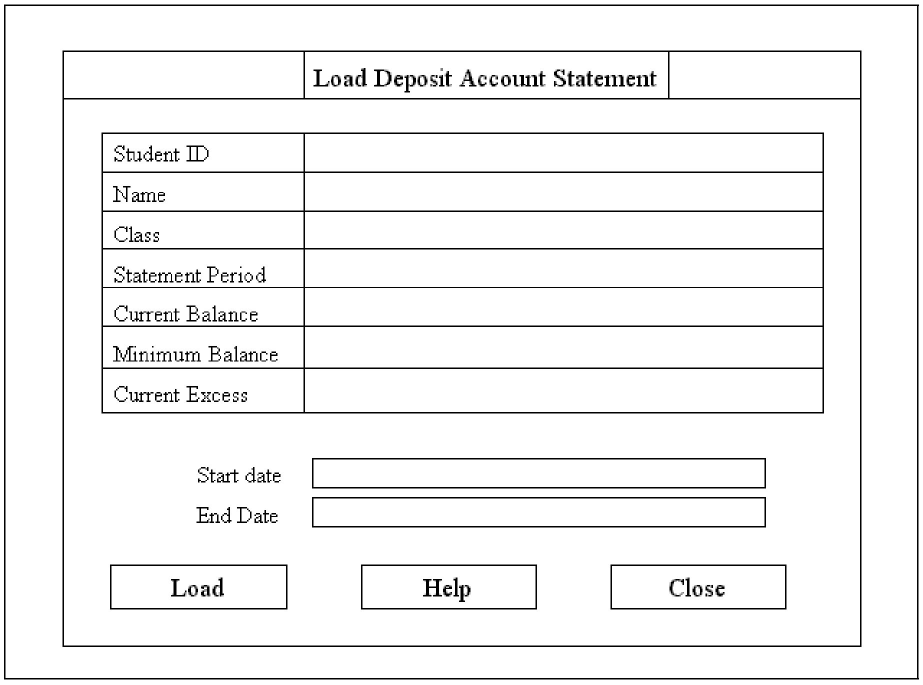
Descendants

C1.2.1.3.1	Output Individual Deposit Statement

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Caution Fee Account Debits	No	Read by query generating statement
Caution Fee Account Credits	No	Read by query generating statement

User Interface



Load Deposit Account Statement

Student ID	
Name	
Class	
Statement Period	
Current Balance	
Minimum Balance	
Current Excess	

Start date
End Date

Input Name	Input Type	Remarks
Start Date	Date Picker	• Required input • Default value is the first of January of the current year
End Date	Date Picker	• Requited input • Default value is today.
Load	Button	• Generate statement of account and load it into C1.2.1.3.1 • Show C1.2.1.3.1 modally
Help	Button	• Display context sensitive help.
Close	Button	• Unload form

C1.2.1.3.1 Output Deposit Statement

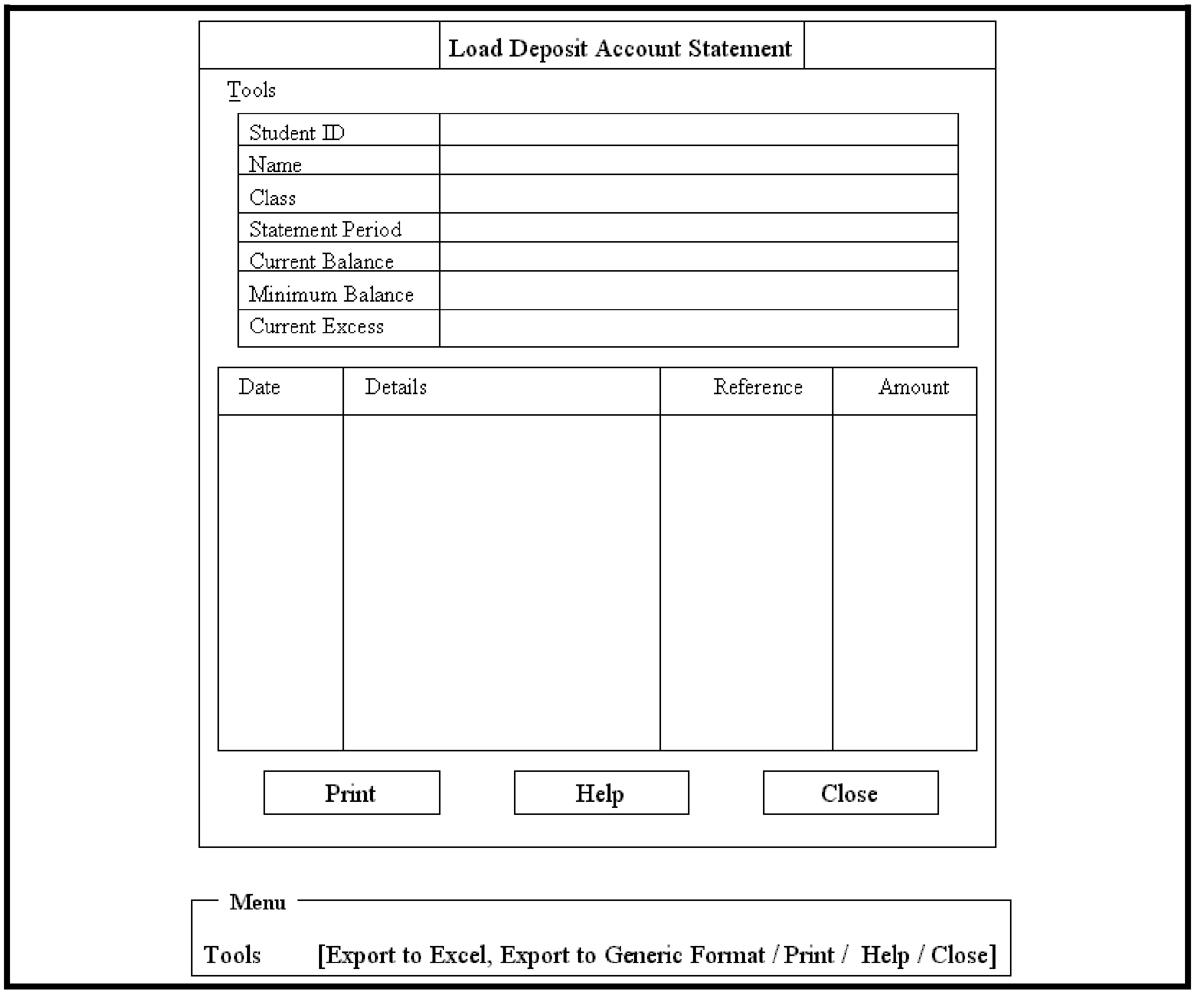
Rationale
Requirement 7d

Parent
C1.2.1.3 Load Individual Deposit Statement

Descendants
None

Data Sources Accessed And Modified
None

User Interface



Load Deposit Account Statement

Tools

Student ID	
Name	
Class	
Statement Period	
Current Balance	
Minimum Balance	
Current Excess	

Date	Details	Reference	Amount

Print
Help
Close
Menu
Tools [Export to Excel, Export to Generic Format / Print / Help / Close]

C1.2.2 Initialise Deposit Statements Printing

Rationale
Requirement 7e

Parent

C1.2	Manage Deposit Accounts

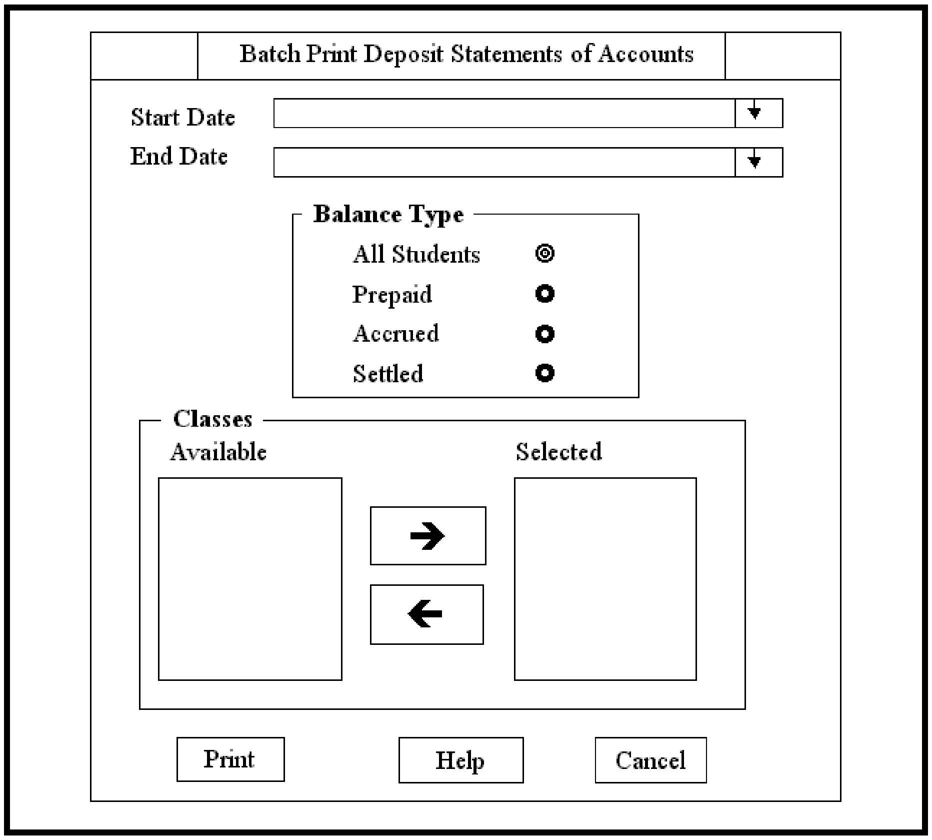
Descendants

C1.2.2.1	Do Deposit Statements Batch Print

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes	No	Populate list boxes in Classes frame

User Interface



Input Name	Input Type	Remarks
Start date	Date Picker	• Defaults to 1 January of current year • Must fall on or before End Date to be valid
End date	Date Picker	• Defaults to today
Balance Type	Option Group	• Defaults to All students
Available Classes	Drop Down List	• Populate with all the available classes
Selected Classes	Drop Down List	• Initialised to empty • Must contain a class before job can proceed
→	Button	• Moves selected class from Available to Selected list
←	Button	• Moves selected class from Selected to Available list
Print	Button	• Start batch job by loading component C1.2.2.1
Help	Button	• Loads context sensitive help
Close	Button	• Unload form

C1.2.2.1 Do Deposit Statements Batch Print

Rationale
Requirement 7e. Displaying progress bar gives the application a responsive look and feel.

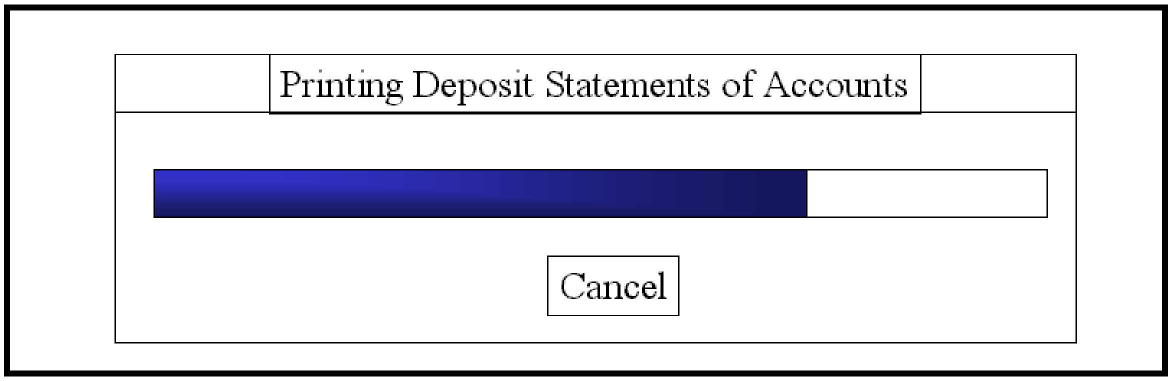
Parent
C1.2.2 Initialise Deposit Statement Printing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students Table	No	Supplies details of students that match the specified criteria. Used in statement header.
Classes table	No	Supplies Minimum Balance details
Caution Fee Account Credits	No	Used to generate body of statement and calculate shortfalls/excesses.
Caution Fee Account Debits	No	Used to generate body of statement and calculate shortfalls/excesses.

User Interface



Input Name	Input Type	Remarks
Cancel	Button	• DoEvents backed button for halting a job in progress.

C1.3 Levy Fees Management

Rationale
Logical grouping of components dealing with the management of Levy Fees.

Parent

C1	Main Subsystem

Descendants

C1.3.1	Template Management
C1.3.2	Debit Levy Account
C1.3.3	Registered Levy Fees
C1.3.4	Levy Statements Production
C1.3.5	Record Levy Payment

Data Sources Accessed And Modified
None

User Interface
None

C1.3.1 Template Management

Rationale
Requirement 2b

Parent
C1.3 Levy Fees Management

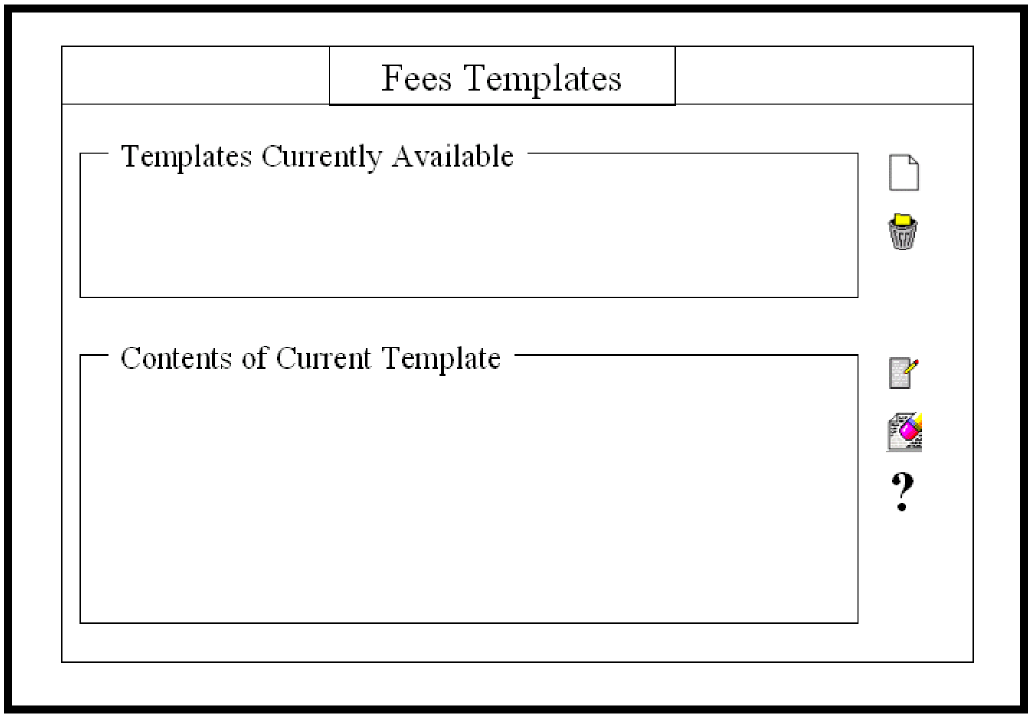
Descendants
C1.3.1.1 Insert Fee

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Fees Templates table	Yes	Read, Modify, Add new and Delete
Registered Fees	No	Fee Code to Fee Name translation

User Interface



Input Name	Input Type	Remarks
🗋	Button	• Create New Template • New Template Name must be unique and can be up to 30 characters
🗑	Button	• Deletes current template
📝	Button	• Use component C1.3.1.1 to handle the fee insertion • When C1.3.1.1 unloads, refresh the display to incorporate any changes made.
🗎	Button	• Delete selected fee from selected template
?	Button	• Load context sensitive help.

C1.3.1.1 Insert Fee Into Template

Rationale
Aids C1.3.1 in meeting requirement 2b

Parent
C1.3.1 Template Management

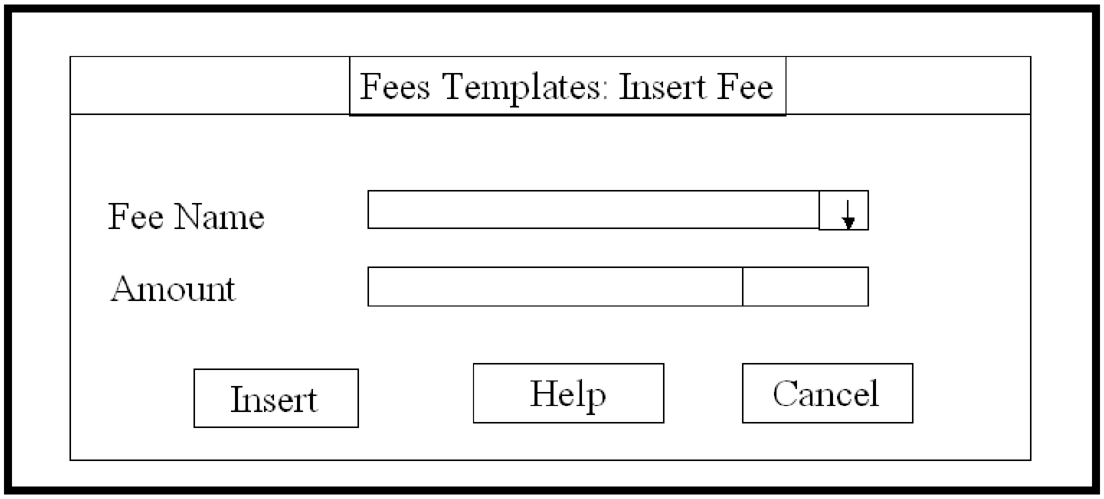
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Fees Templates table	Yes	Template record modified to include new fee.
Registered Fees	No	Supplies list of all the registered fees Fee Name to Fee code translation

User Interface



Input Name	Input Type	Remarks
Fee Name	Drop Down List	• Must contain a list of all the registered fees
Amount	Text Box	• Used to specify the amount • Must be greater than $0.00
Insert	Button	• Modifies template record to include specified fee • For fee to be accepted, it must not already be in the template.

C1.3.2 Debit Levy Account

Rationale
Requirement 2a

Parent

C1.3	Levy Fees Management

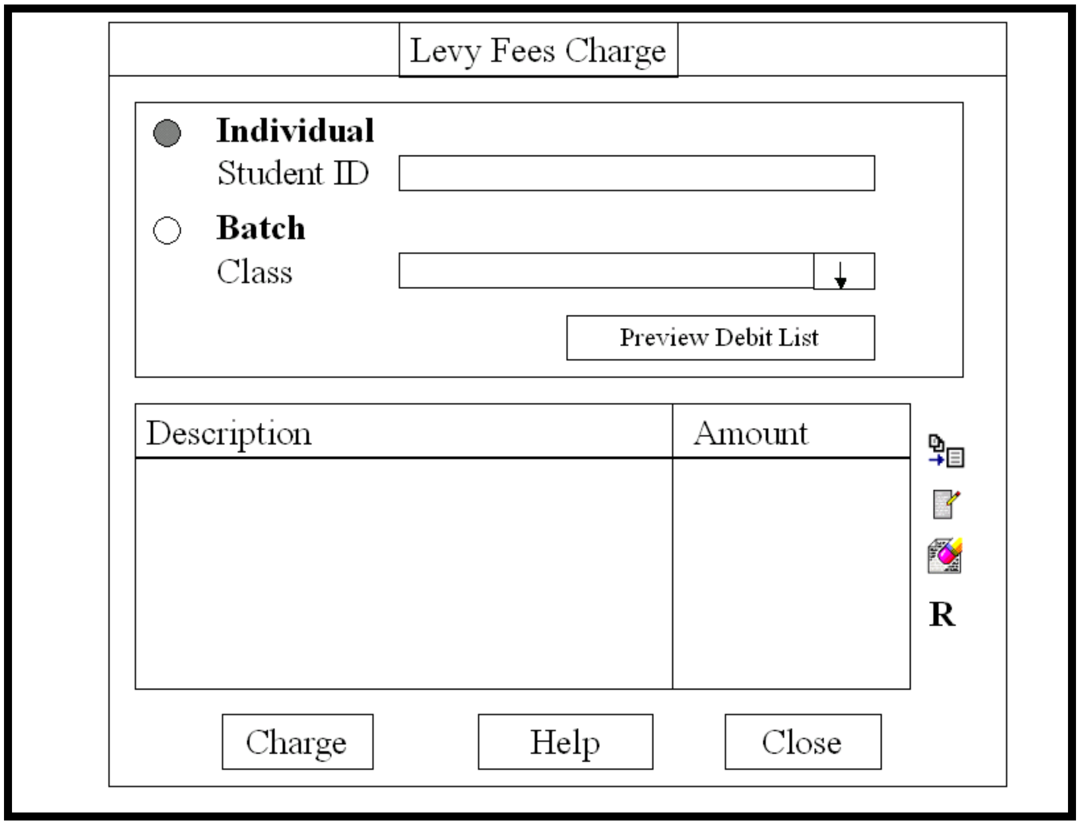
Descendants

C1.3.2.1	Load template
C1.3.2.2	Batch Charge
C1.3.2.3	Individual Charge
C1.3.2.4	Insert Fee

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes	No	Provides list of available classes.
Students	No	Provides student details for confirmation.
Levy Account Debits	Yes	Add new entries for individual charge

User Interface



Input Name	Input Type	Remarks
Individual / Batch	Option Group	• One must be selected • Default to individual
Student ID	Text box	• Required if opt Individual is selected. • Must be a valid student ID
Class	Drop down list	• Populate with list of all the available classes • One of the list items must be selected if opt batch is selected
Preview Debit List	Button	• Launch C1.1.5 to display a list of all the students in the selected class.
[Description, Amount]	Grid Control	• List of all the fees to be charged along with their associated amounts • Description must be a fee name.
📋	Button	• Call component C1.3.2.1 to load a template
📝	Button	• Load component C1.3.2.4 to insert a fee
🧽	Button	• Remove the selected fee from the list
R	Button	• Reset (Clear) the list of fees

C1.3.2.1 Load Fee Template

Rationale
Requirement 2b

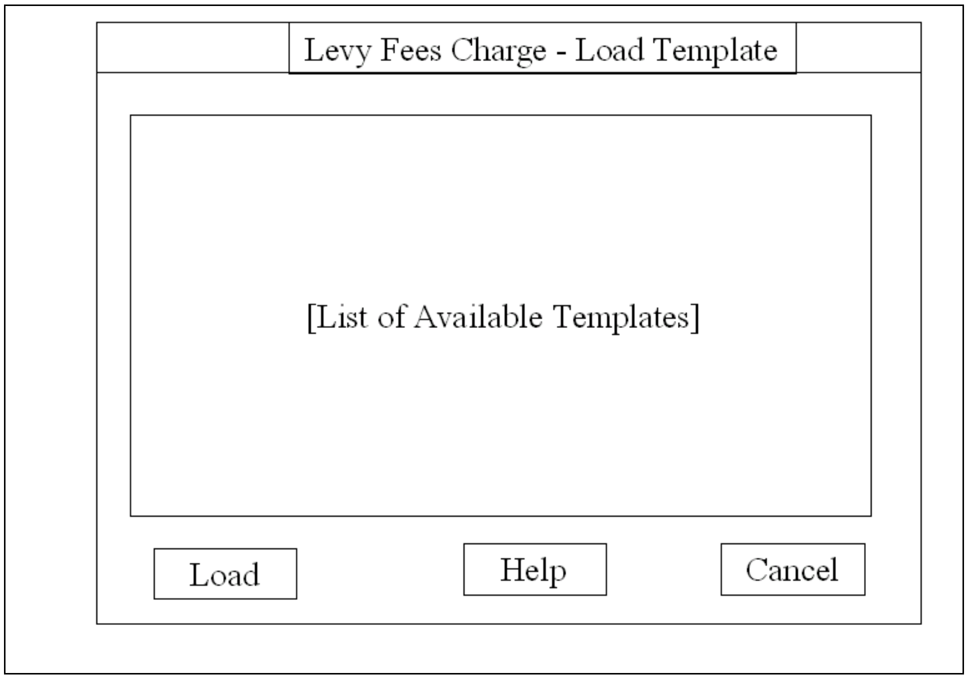
Parent
C1.3.2 Debit Levy Account

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Fees Templates	No	Read to provide list of available templates and also their contents.
Registered Fees	No	Fee code to Fee name translation.

User Interface


Levy Fees Charge - Load Template
[List of Available Templates]
Load
Help
Cancel

Input Name	Input Type	Remarks
Load	Button	• Insert the contents of the selected template into the parent form and then Unload component • Template insertion must not create fee duplicates
Help	Button	• Load context sensitive help
Cancel	Button	• Unload component

C1.3.2.2 Batch Debit Levy Accounts

Rationale
Requirement 2c

Parent
C1.3.2 Debit Levy Accounts

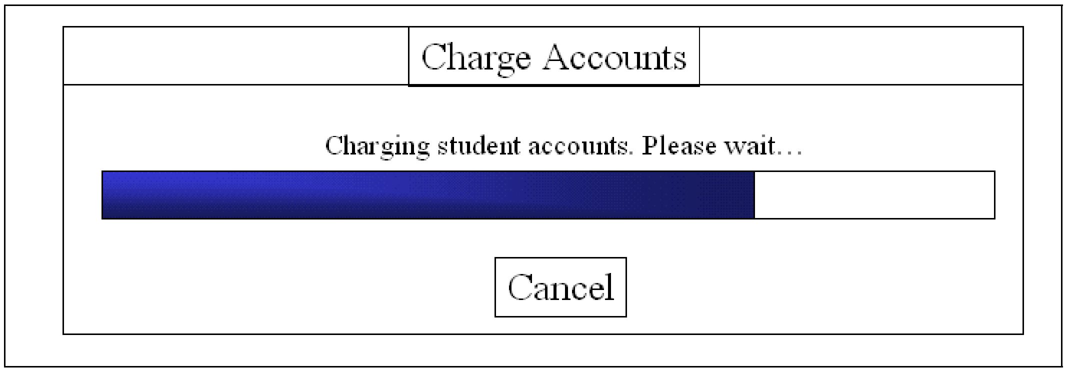
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students table	No	Provides list of students in the class selected as a target.
Levy Account Debits	Yes	Adds new records

User Interface



Charging student accounts. Please wait…

Input Name	Input Type	Remarks
Cancel	Button	• DoEvents backed process interruption control.

C1.3.2.3 Charge Individual Levy Account

Rationale
Requirement 2a

Parent
C1.3.2 Debit Levy Account

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students table	No	Supplies student details for use in confirming transaction.
Levy Account Debits	Yes	Add new records

User Interface
None

C1.3.2.4 Insert Individual Fee

Rationale
Requirement 2a

Parent
C1.3.2 Debit Levy Accounts

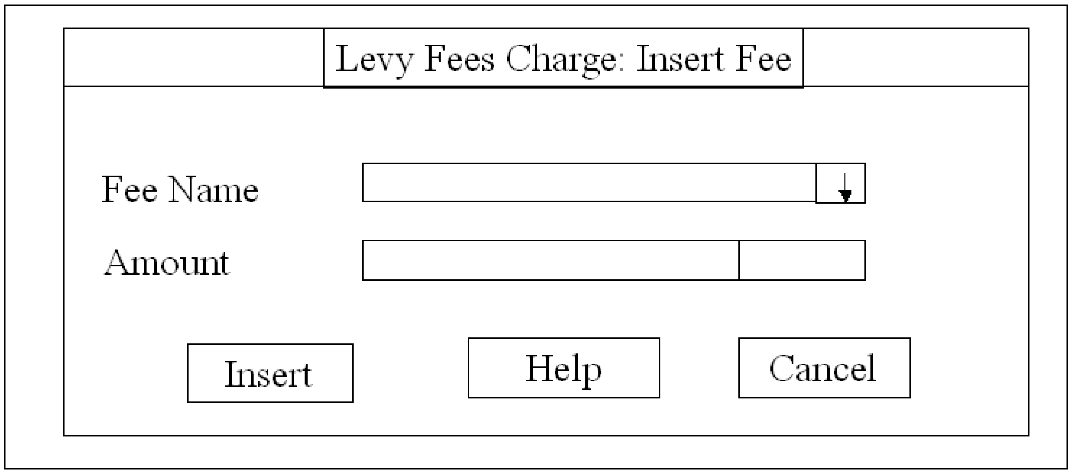
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Registered Fees	No	Supplies list of registered fees for the Fee Name drop down list

User Interface



Input Name	Input Type	Remarks
Fee Name	Drop down list	• Populated with all the registered fees • Required input
Amount	Text Box	• Required currency format input • Must be greater than $0.00
Insert	Button	• Insert the selected fee into the grid provided it does not result in duplicates
Help	Button	• Display context sensitive help
Cancel	Button	• Unload form

C1.3.3 Registered Fees

Rationale
Required for managing the Registered Fees table specified in the data modelling.

Parent

C1.3	Levy Fees Management

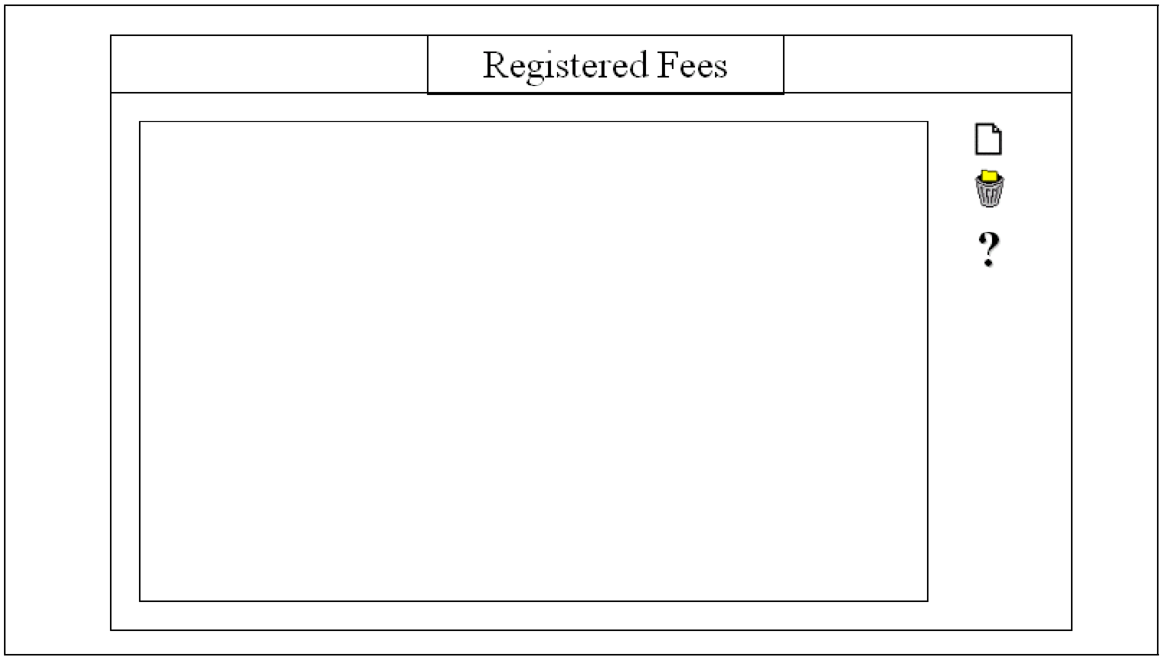
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Registered Fees	Yes	Add new and delete
Levy Account Debits	No	Used to obtain details on fee usage before it is deleted. A fee referenced by the Levy Debits Table may not be deleted.

User Interface



Input Name	Input Type	Remarks
🗋	Button	• Add a new fee • Name must be unique
🗑	Button	• Delete fee • Fee must not be in use.
?	Button	• Load context sensitive help.

C1.3.4 Statement of Account Production

Rationale
Abstract logical grouping

Parent

C1.3	Levy Fees Management

Descendants

1.3.4.1	Individual
1.3.4.2	Start Batch

Known Dependencies
None

Data Sources Accessed And Modified
None

User Interface
None

C1.3.4.1 Individual Statement of Account

Rationale
Requirement 4a

Parent

C1.3.4	Statement of Account Production

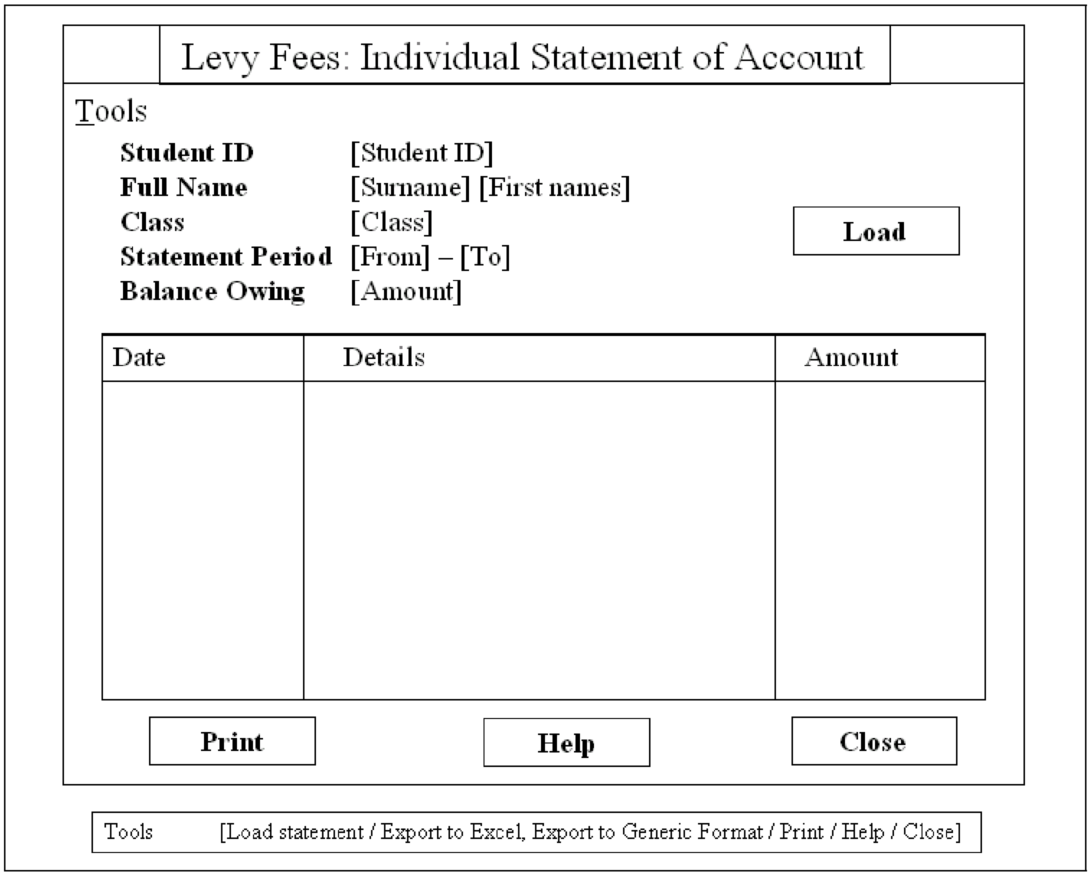
Descendants

1.3.4.1.1	Load Statement of Account

Known Dependencies
None

Data Sources Accessed And Modified
None

User Interface


Levy Fees: Individual Statement of Account

Tools

Student ID	[Student ID]
Full Name	[Surname] [First names]
Class	[Class]
Statement Period	[From] – [To]
Balance Owing	[Amount]

Date	Details	Amount

Tools [Load statement / Export to Excel, Export to Generic Format / Print / Help / Close]

Input Name	Input Type	Remarks

Load	Button	• Use component 1.3.4.1.1 to load a statement
Print	Button	• Print the displayed statement
Help	Button	• Display context sensitive help
Close	Button	• Unload form.
Load statement	Menu	• Use component 1.3.4.1.1 to load a statement
Export to Excel	Menu	• Use component 4.3 to export the statement
Export to Generic format	Menu	• Use component 4.3 to export the statement
Print	Menu	• Print statement
Help	Menu	• Load context sensitive help.
Close	Menu	• Unload form

C1.3.4.1.1 Load Statement

Rationale
Requirement 4a

Parent

C1.3.4.1	Individual Statement of Account

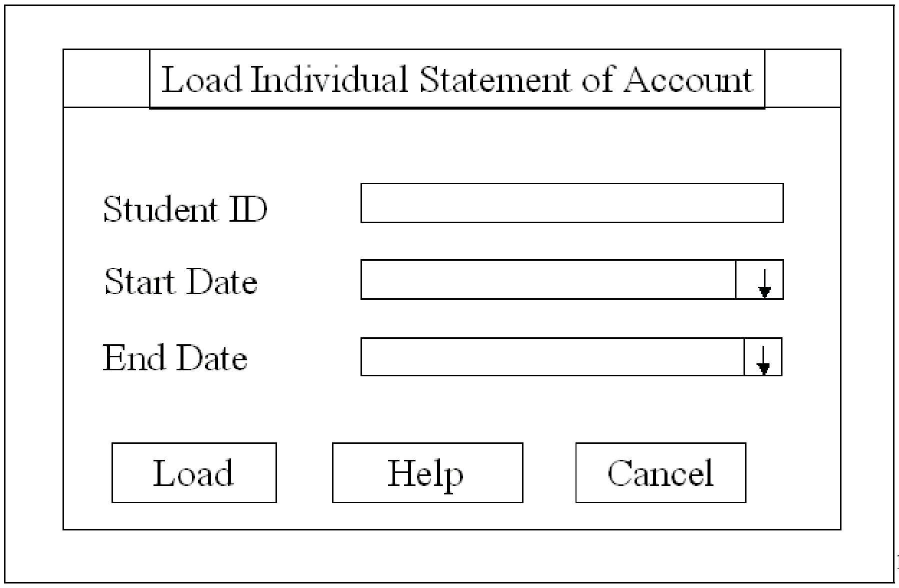
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students	No	Get student details
Levy Account Debits	No	Used in calculating balances and getting the list of transaction s occurring between two user specified dates.
Levy Account Credits	No	Used in calculating balances and getting the list of transaction s occurring between two user specified dates.

User Interface



[1]

Input Name	Input Type	Remarks

[1]

Student ID	Text box	• Valid student ID
Start Date	Date Picker	• Default to 1 January of current year
End date	Date Picker	• Default to Today
Load	Button	• Generate statement and place in parent form
Help	Button	• Load context sensitive help
Cancel	Button	• Unload form

C1.3.4.2 Start Batch Printing Levy Statement

Rationale
Requirement 4b

Parent
C1.3.4 Statement of Account Production

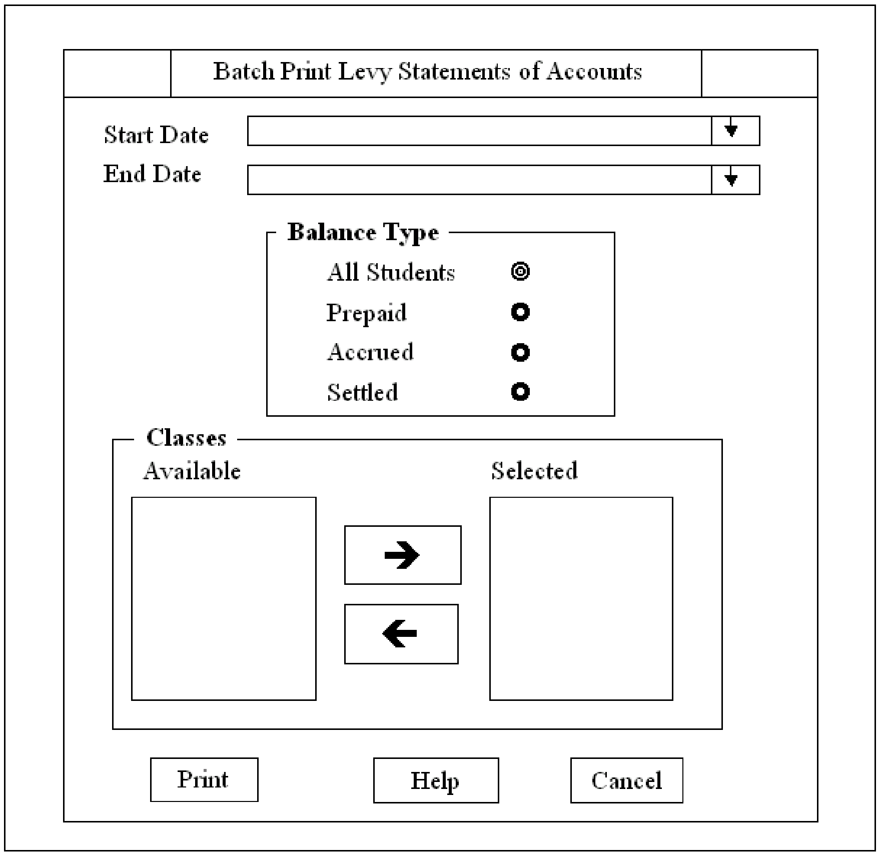
Descendants
1.3.4.2.1 Do Batch Print

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Classes	No	Populate list boxes in Classes frame

User Interface



Input Name	Input Type	Remarks
Start date	Date Picker	• Defaults to 1 January of current year

		• Must fall on or before End Date to be valid
End date	Date Picker	• Defaults to today
Balance Type	Option Group	• Defaults to All students
Available Classes	Drop Down List	• Populate with all the available classes
Selected Classes	Drop Down List	• Initialised to empty • Must contain a class before job can proceed
→	Button	• Moves selected class from Available to Selected list
←	Button	• Moves selected class from Selected to Available list
Print	Button	• Start batch job by loading component C1.3.4.2.1
Help	Button	• Loads context sensitive help
Close	Button	• Unload form

C1.3.4.2.1 Batch Print Levy Statements

Rationale
Requirement 4b. Displaying progress bar gives the application a responsive look and feel.

Parent

C1.3.4.2	Start Batch Printing Levy Statements

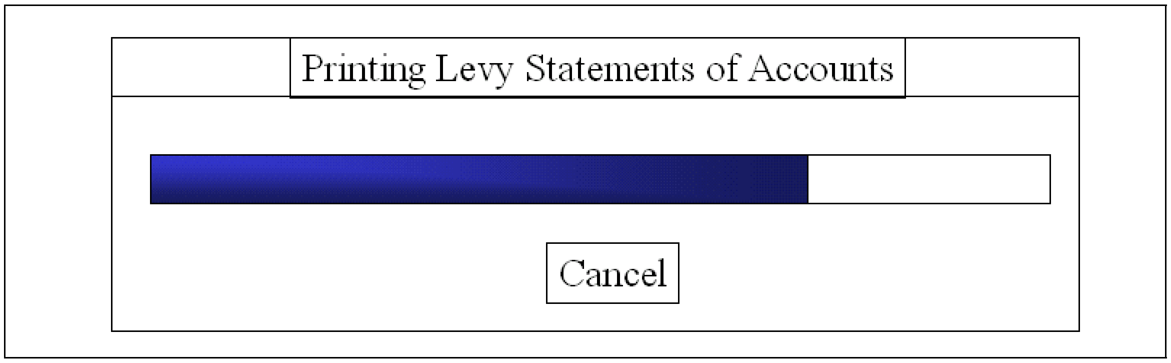
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Students Table	No	Supplies details of students that match the specified criteria. Used in statement header.
Levy Account Credits	No	Used to generate body of statement and calculate shortfalls/excesses.
Levy Account Debits	No	Used to generate body of statement and calculate shortfalls/excesses.

User Interface



Input Name	Input Type	Remarks
Cancel	Button	• DoEvents backed button for halting a job in progress.

C1.3.5 Record Payment

Rationale
Requirement 3

Parent

C1.3	Levy Fees Management

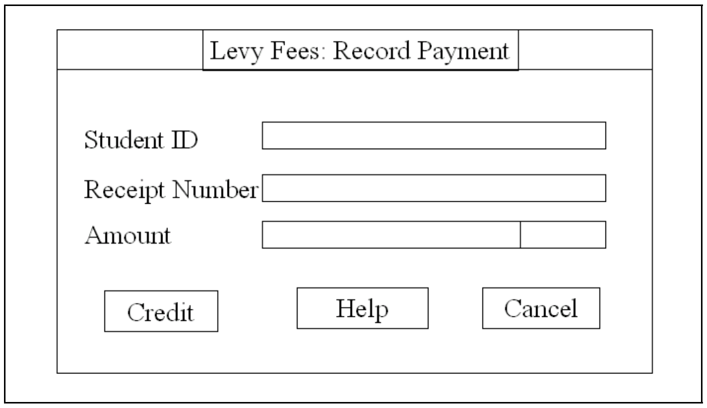
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Levy Account Credits	Yes	Add new record Checking if receipt number is already in use.
Students	No	Verifying Student ID Getting details of student for confirmation of transaction.

User Interface



Input Name	Input Type	Remarks
Student Id	Text box	• Must be a valid student ID • Required input
Receipt number	Text box	• Must be unique in table • Required input of up to 15 characters
Amount	Text box	• Required currency input. • Must be greater than $0.00
Credit	Button	• Enter the credit transaction
Help	Button	• Load context sensitive help
Cancel	Button	• Unload form

C1.4 Balance Analysis

Rationale
Abstract logical grouping

Parent

C1	Main Subsystem

Descendants

C1.4.1	Fee Class
C1.4.2	Student Balances

Data Sources Accessed And Modified
None

User Interface
None

C1.4.1 Fee Class

Rationale
Requirement 5

Parent

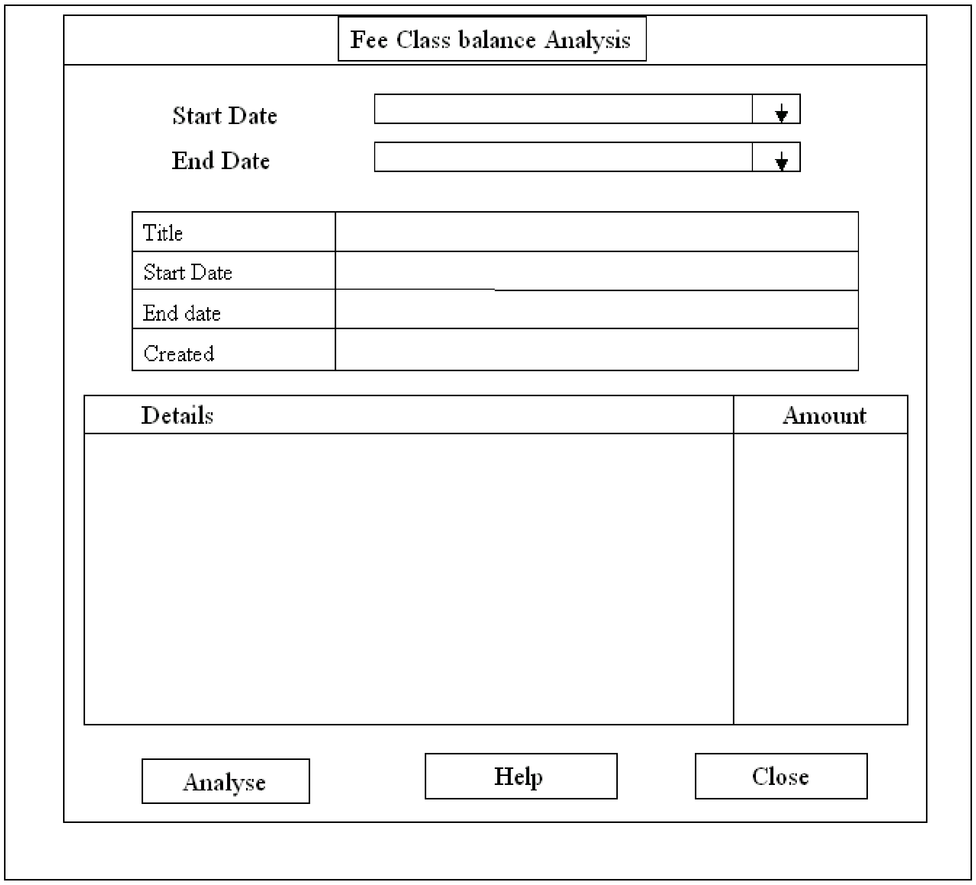
C1.4	Balance Analysis

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Remarks
Students	No	Grouping of students so as to calculate account balances.
Registered Fees	No	Grouping of fees so as to get each fee total
Levy Account Debits	No	Calculation of levy account balances
Levy Account Credits	No	Calculation of levy account balances
Caution Fee Account Credits	No	Calculation of caution deposit fee account balances
Caution Fee Account Credits	No	Calculation of caution deposit fee account balances
Classes	No	Retrieval of minimum balances to determine shortfalls and excesses of Caution Fee Accounts

User Interface



Input Name	Input Type	Remarks
Start Date	Date Picker	• Default to 1 January of current year
End Date	Date Picker	• Default to today • Must fall on or after Start Date
Analyse	Button	• Carry out the analysis
Help	Button	• Display context sensitive help.
Close	Button	• Unload form

Output Specification

Section	Subsection	Contents
Header	Title	Fee Class Balance Analysis
	Start Date	Specified start date in long date format
	End Date	Specified end date in long date format
	Created	Time at which analysis finished
Details	Deposit Analysis	Current shortfall Current excess Current net [excess | shortfall] Current total credits Current total debits Current Total Minimum Balances Duration Credits Duration Debits
	Levy Payments	Total payments over period Current Accruals Current Prepayments
	Levy Fees Charged	For each fee, give total

C1.4.2 Student Balances

Rationale
Abstract logical grouping

Parent

C1.4	Balance Analysis

Descendants

C1.4.2.1	Levy Balances
C1.4.2.2	Deposit Balances

Data Sources Accessed And Modified
None

User Interface
None

C1.4.2.1 Levy Student Balance Analysis

Rationale
Requirement 6

Parent
C1.4.1 Student Balances

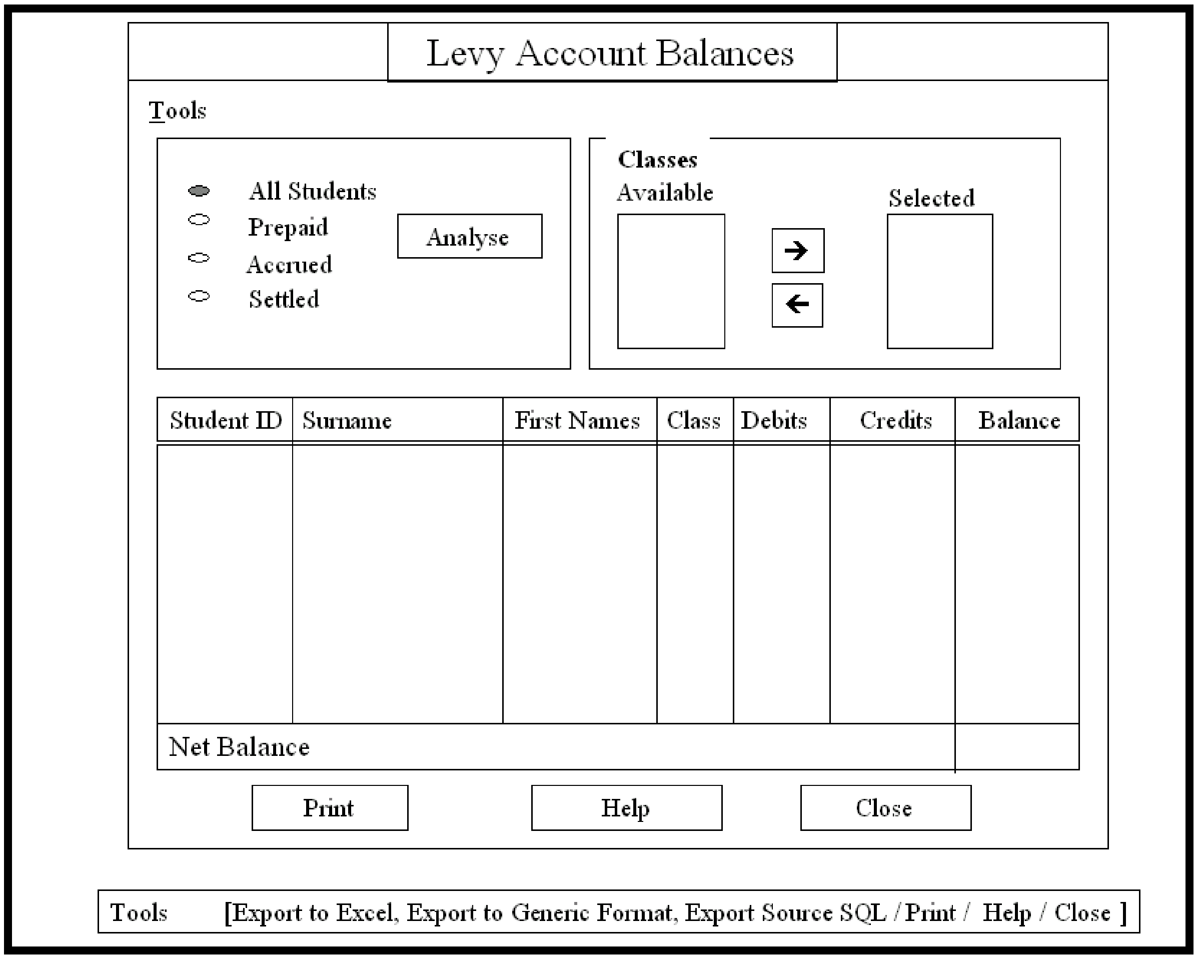
Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Levy Account Debits table	No	Supplies total of debit entries for each student
Levy Account Credits table	No	Supplies total of credit entries for each student
Students table	No	Used in grouping
Classes table	No	Used to populate class list boxes.

User Interface



Student ID	Surname	First Names	Class	Debits	Credits	Balance
Net Balance						

Tools [Export to Excel, Export to Generic Format, Export Source SQL / Print / Help / Close]

Input Name	Input Type	Remarks
Balance Type	Option group	• Default to all students
Classes Available	List	• Populate with classes table
Classes Selected	List	• Contains list of classes to be include in the analysis • Must not be empty
Analyse	Button	• Generate analysis details
➔	Button	• Move the class selected in the Available list to the Selected list
🡐		• Move the class selected in the Selected list to the Available list
Print	Button	• Send analysis to printer
Help	Button	• Display context sensitive help.
Close	Button	• Unload form
Export to Excel	Menu	• Use component 4.3 to export the results
Export to Generic format	Menu	• Use component 4.3 to export the results
Export Source SQL	Menu	• Use component 4.3 to export the source SQL
Print	Menu	• Send results to printer
Help	Menu	• Display context sensitive help
Close	Menu	• Unload form

C1.4.2.2 Deposit A/c Student Balance Analysis

Rationale
Requirement 6

Parent

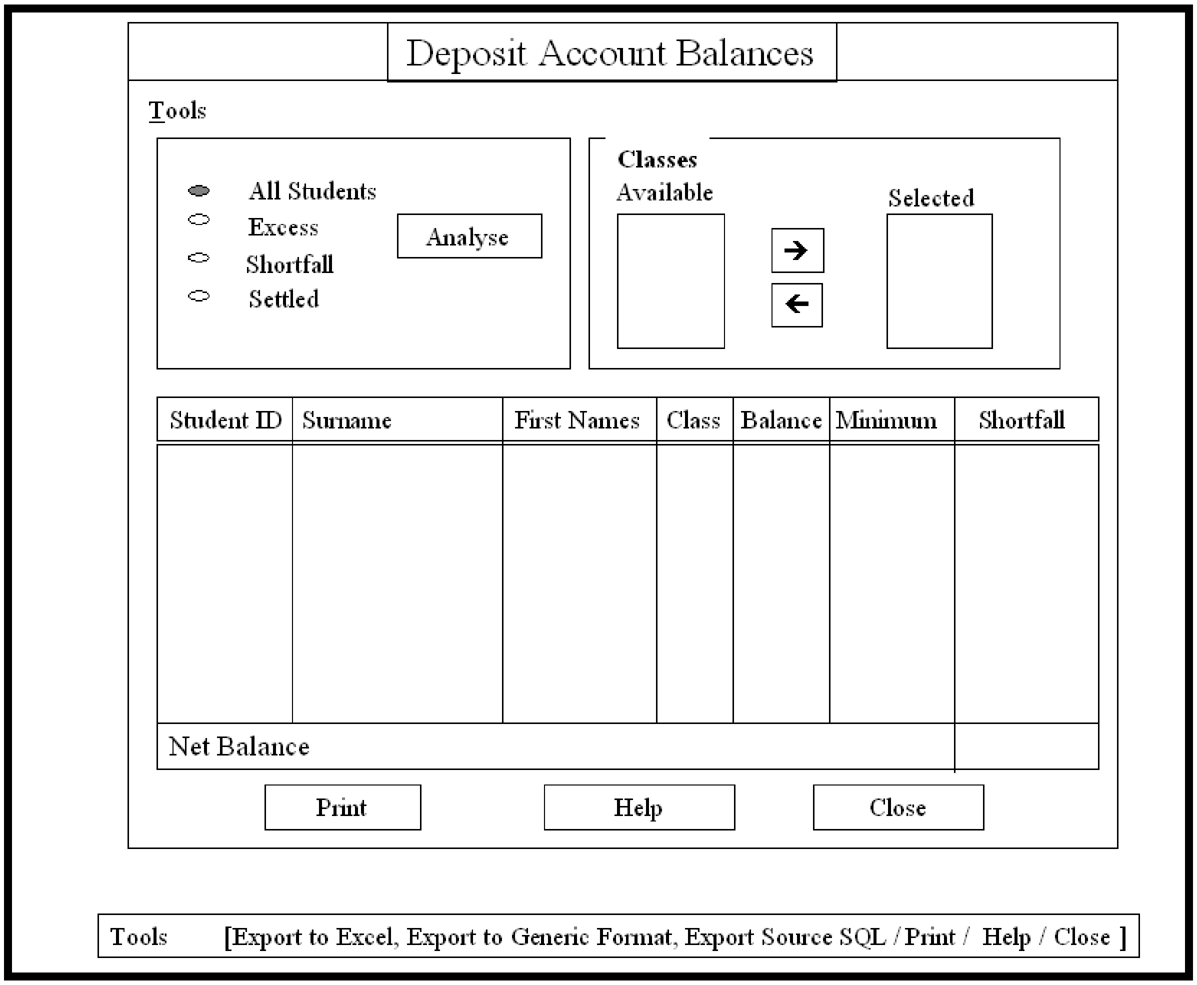
C1.4.2	Student Balances

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Caution Fee Account Debits table	No	Supplies total of debit entries for each student
Caution Fee Account Credits table	No	Supplies total of credit entries for each student
Students table	No	Used in grouping
Classes table	No	Supplied minimum balances and used to populate class list boxes.

User Interface



Deposit Account Balances

Tools

- [● All Students]
- Excess
- Shortfall
- Settled

Analyse

Classes

Available

Selected

Student ID	Surname	First Names	Class	Balance	Minimum	Shortfall
Net Balance						

Print Help Close

Tools [Export to Excel, Export to Generic Format, Export Source SQL / Print / Help / Close]

Input Name	Input Type	Remarks
Balance Type	Option group	• Default to all students
Classes Available	List	• Populate with classes table
Classes Selected	List	• Contains list of classes to be include in the analysis • Must not be empty
Analyse	Button	• Generate analysis details
➔	Button	• Move the class selected in the Available list to the Selected list
🡐		• Move the class selected in the Selected list to the Available list
Print	Button	• Send analysis to printer
Help	Button	• Display context sensitive help.
Close	Button	• Unload form
Export to Excel	Menu	• Use component 4.3 to export the results
Export to Generic format	Menu	• Use component 4.3 to export the results
Export Source SQL	Menu	• Use component 4.3 to export the source SQL
Print	Menu	• Send results to printer
Help	Menu	• Display context sensitive help
Close	Menu	• Unload form

C2 Mail Merge

Rationale
Provides user interface for accessing the mail merge components.

Parent

C0	Level 0

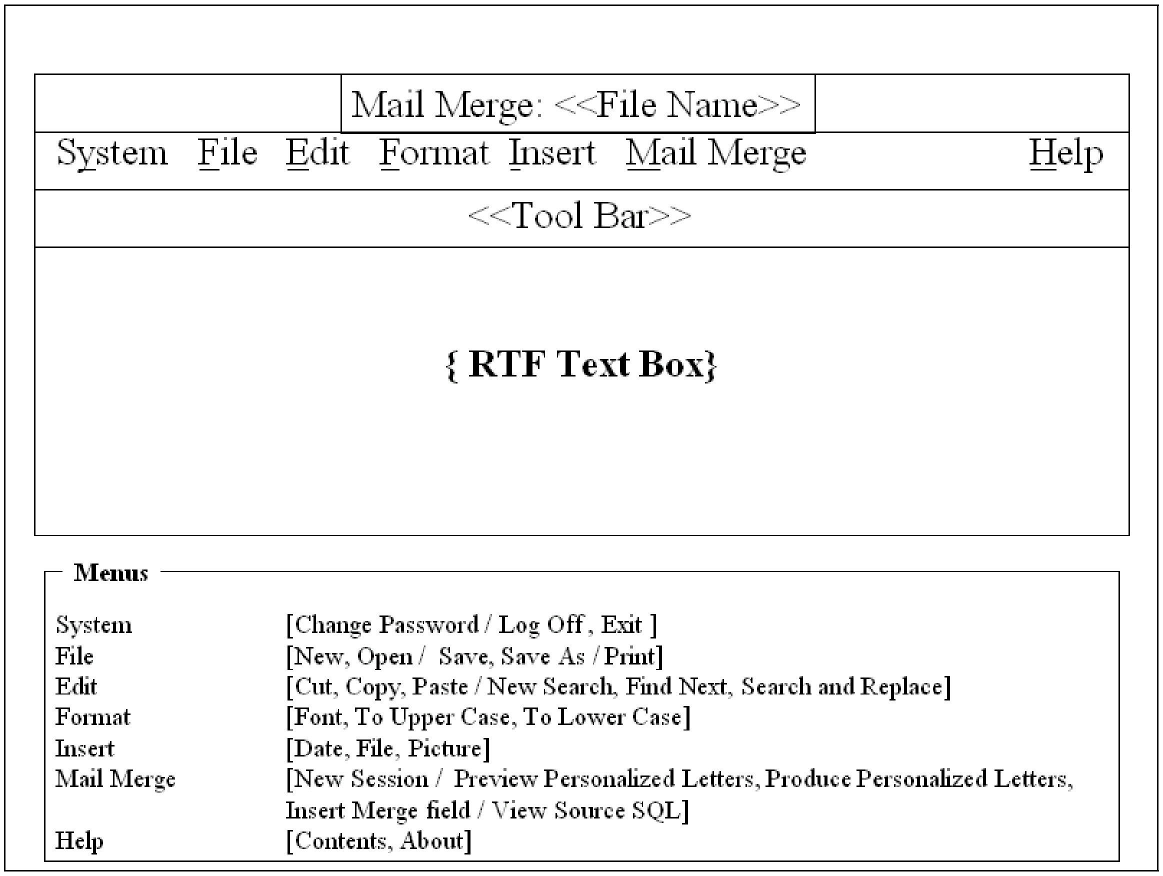
Descendants

C2.1	File Management
C2.2	Text formatting and editing
C2.3	Mail Merge processing

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Reports Query	Yes	Sets SQL property of object to the contents of an SQL text file. Query executed and used to create recordset.

User Interface


Mail Merge: <<File Name>>
System File Edit Format Insert Mail Merge Help
<<Tool Bar>>
{ RTF Text Box}
Menus

System	[Change Password / Log Off , Exit]
File	[New, Open / Save, Save As / Print]
Edit	[Cut, Copy, Paste / New Search, Find Next, Search and Replace]
Format	[Font, To Upper Case, To Lower Case]
Insert	[Date, File, Picture]
Mail Merge	[New Session / Preview Personalized Letters, Produce Personalized Letters, Insert Merge field / View Source SQL]
Help	[Contents, About]

Input Name	Input Type	Remarks
Change Password	Menu	• Load C4.4
Log Off	Menu	• Reload component C4.1
Exit	Menu	• Shut down system
New File	Menu	• Load C2.1.1
Open	Menu	• Load C2.1.2
Save	Menu	• Load C2.1.3
Save As	Menu	• Load C2.1.3
Print	Menu	• Load C2.1.4
Cut	Menu	• Place selected test onto clipboard. Remove text in current selection from document
Copy	Menu	• Copy Selected text onto clipboard
Paste	Menu	• Insert clipboard text into current document at current cursor position. If there is selected text, overwrite it.
New Search	Menu	• Reinitialise component C2.2.3 and then load it.
Search	Menu	• Load C2.2.3
Search and Replace	Menu	• Load C.2.2.4
Font	Menu	• Load C2.2.1
To Upper case	Menu	• Convert selected text o upper case
To Lower Case	Menu	• Convert selected text o upper case
Insert date	Menu	• Load C2.2.3.1
Insert Picture	Menu	• Load C2.2.3.2
Insert File	Menu	• Load C2.2.3.3
New Session	Menu	• Load C2.3.1
Preview Personalised Letters	Menu	• Load C2.3.4
Produce Personalised Letters	Menu	• Load C2.3.5
Insert Merge Field	Menu	• Load C2.3.3
View Query Source	Menu	• Load C2.3.2
Help Contents	Menu	• Load context sensitive help
Help About	Menu	• Display credits

C2.1 File Management

Rationale
Abstract logical grouping of file management components

Parent

C2	Mail merge

Descendants

C2.1.1	Create New File
C2.1.2	Open file
C2.1.3	Save existing file
C2.1.4	Print Current File

Data Sources Accessed And Modified
None

User Interface
None

C2.1.1 Create new Document

Rationale
Requirement 8.c.i

Parent

C2.1	File Management

Descendants
None

Known Dependencies
None

Data Sources Accessed And Modified
None

User Interface
None

Algorithm
If the currently loads file has changed since being last saved, give option to save.
Set default font settings to

- Times New Roman 12pt
- Left justified
- Not bold
- Not italic
- Not underlined
- Not struck through
- Not in bullet style
- Black text fore colour
- No indent

Blank out the document area

C2.1.2 Open Document

Rationale
Requirement 8.c.ii

Parent

C2.1	File Management

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System open dialogue

Algorithm
If the currently loads file has changed since being last saved, give option to save.
Blank out the document area
Load the specified document using the **LoadFile** method of the RTF text box.

C2.1.3 Save Document

Rationale
Requirement 8.c.ii

Parent

C2.1	File Management

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System save dialogue

Remarks
Use RTF text box's **SaveFile** method. If the loaded file already has a file name associated with it, then proceed to use this file name without prompting unless the user has explicitly requested that this not be done.

C2.1.4 Print File

Rationale
Requirement 8.c.iii

Parent

C2.1	File Management

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System Print dialogue

Remarks
Use Pierre-Emmanuel Gross' routine for printing RTF files. Do not use the default **PrintRTF** method because it does not support margins.

Code source at http://www.codeguru.com/vb/openfaq/comments/159.shtml

C2.2 Text Formatting and Editing

Rationale
Abstract logical grouping of file management components

Parent

C2	Mail merge

Descendants

C2.2.1	Font and Style Management
C2.2.2	Search
C2.2.3	Insertions
C2.2.4	Search and Replace

Data Sources Accessed And Modified
None

User Interface
None

C2.2.1 Font and Style Management

Rationale
Requirement 8.c.vii

Parent

C2.2	Text Formatting and Editing

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System Font dialogue

Remarks
The Font dialogue should be loaded with the current style before being displayed. Appropriate treatment of nulls should be provided.

C2.2.2 Search

Rationale
Requirement 8.c.iv

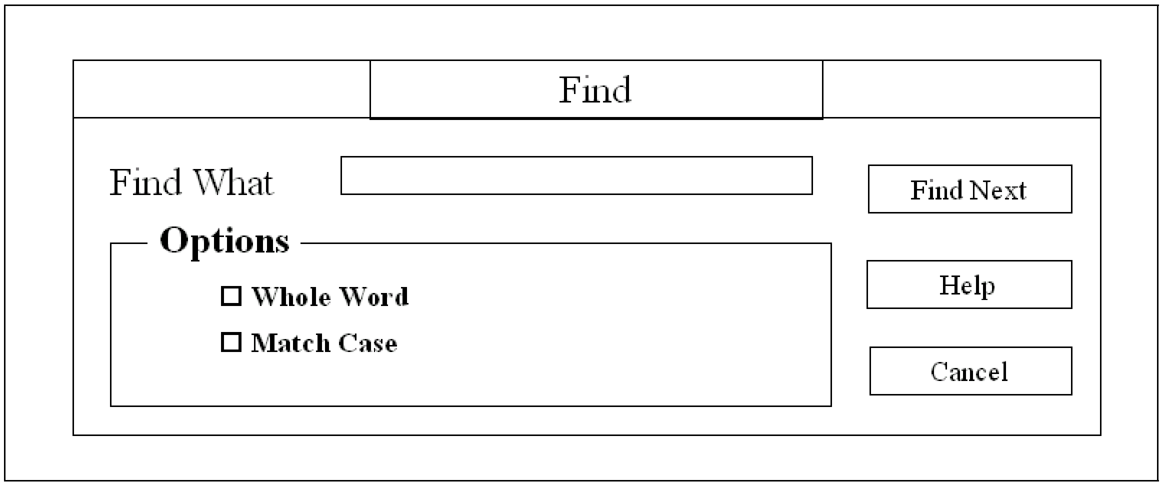
Parent

C2.2	Text Formatting and Editing

Descendants
None

Data Sources Accessed And Modified
None

User Interface



Remarks
Should carry a reset option to enable new searches. On finding a match, the focus should be returned to the document with the match highlighted.

C2.2.3 Insertions

Rationale
Abstract grouping of components used in insertions

Parent

C2.2	Text Formatting and Editing

Descendants

C2.2.3.1	Date Insertion
C2.2.3.2	Picture Insertion
C2.2.3.3	File Insertion

Data Sources Accessed And Modified
None

User Interface
None

C2.2.3.1 Insert Date

Rationale
Requirement 8.e.i

Parent

C2.2.3	Insertions

Descendants
None

Data Sources Accessed And Modified
None

User Interface
Month view control for the current year. Input should be by double clicking a date.

C2.2.3.2 Insert Picture

Rationale
Requirement 8.e.ii

Parent
C2.2.3 Insertions

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System's open dialogue with a filter for the following formats

- Bitmap (Default)
- JPEG Graphic Interchange Format
- Graphic Interchange Format

C2.2.3.3 Insert File

Rationale
Requirement 8.e.iii

Parent

C2.2.3	Insertions

Descendants
None

Data Sources Accessed And Modified
None

User Interface
System's open dialogue with a filter for the following formats
- Rich Text Format (Default)
- Standard Text

Remarks
Use a working area form with an RTF control to load the file and the copy the contents of this control and paste them into the current document at the current insertion position.

C2.2.4 Search and Replace

Rationale
Requirement 8.c.iv

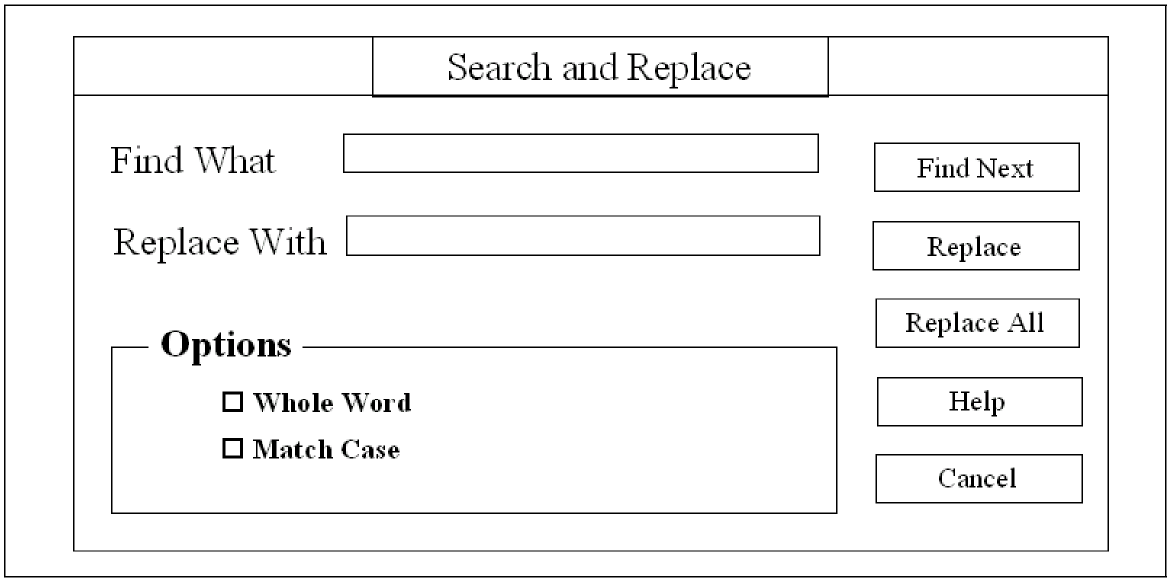
Parent

C2.2	Text Formatting and Editing

Descendants
None

Data Sources Accessed And Modified
None

User Interface



C2.3 Mail Merge Processing

Rationale
Abstract logical grouping of Mail merge processing routines.

Parent

C2	Mail Merge

Descendants

C2.3.1	Load Query Source File
C2.3.2	View Query Source
C2.3.3	Insert Mail Merge Field
C2.3.4	Preview Personalised Letters
C2.3.5	Produce Personalised Letters

Data Sources Accessed And Modified
None

User Interface
None

C2.3.1 Load Query Source File

Rationale
Requirement 8a

Parent

C2.3	Mail Merge Processing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Report Query Object	Yes	SQL property set to contents of SQL text file

User Interface
None

Algorithms
1. Use the System Open File dialogue box to get the file name of the source SQL file
2. Assign query file's contents to the SQL property of the Report QueryDef object. If assignment fails because of a syntax error, give the user an option to try loading a different file.
3. Check that the specified query does not modify records. If it does, give the option to try another query file.
4. Execute the stored query and assign the results to a global recordset object.

C2.3.2 View Query Source

Rationale
Requirement 8b

Parent
C2.3 Mail Merge Processing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Report Query Object	No	Retrieves the SQL property of the QueryDef object.

User Interface
A form containing a text box with both scroll bars.

Algorithms
Retrieve SQL property of QueryDef object and assigns contents to the text box on the form.

C2.3.3 Insert Merge Field

Rationale
Requirement 8b

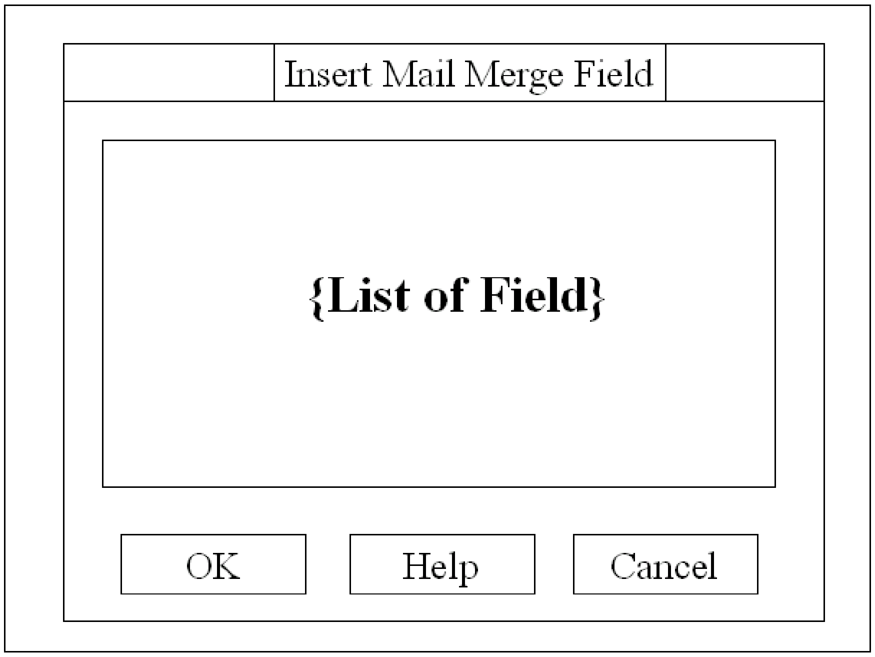
Parent
C2.3 Mail Merge Processing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Query results recordset	No	Iterates through field objects

User Interface



Algorithms

Component Load
Retrieve all the fields in the query result and place them in the List of Fields

OK Chosen
Paste the caption of the selected list item into the document surrounded by start and end of field characters.

C2.3.4 Preview Personalised Letters

Rationale
Requirement 8

Parent
C2.3 Mail Merge Processing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Query Result Recordset	No	Supplies list of personalised details

User Interface
Form containing an RTF component with both scroll bars.

Algorithm
```
Check if every field specified exists in the query result
If an invalid field is encountered
    Inform user and abort operation
Else
    For each record in the query result
        Make a copy of the input document in a working area
        For each field in the source query
            Replace field name in working area copy with personalised details
        Next
        Send result to Output form
    Next
    Display output form
End if
```

C2.3.5 Produce Personalised Letters

Rationale
Requirement 8

Parent

C2.3	Mail Merge Processing

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Query Result Recordset	No	Supplies list of personalised details

User Interface
System print dialogue

Algorithm

```
Check if every field specified exists in the query result
If an invalid field is encountered
    Inform user and abort operation
Else
    For each record in the query result
        Make a copy of the input document in a working area
        For each field in the source query
            Replace field name in working area copy with personalised details
        Next
        Send result to printer starting on a new page
    Next
End if
```

C3 Administrative Console

Rationale
Provide support for system configuration

Parent

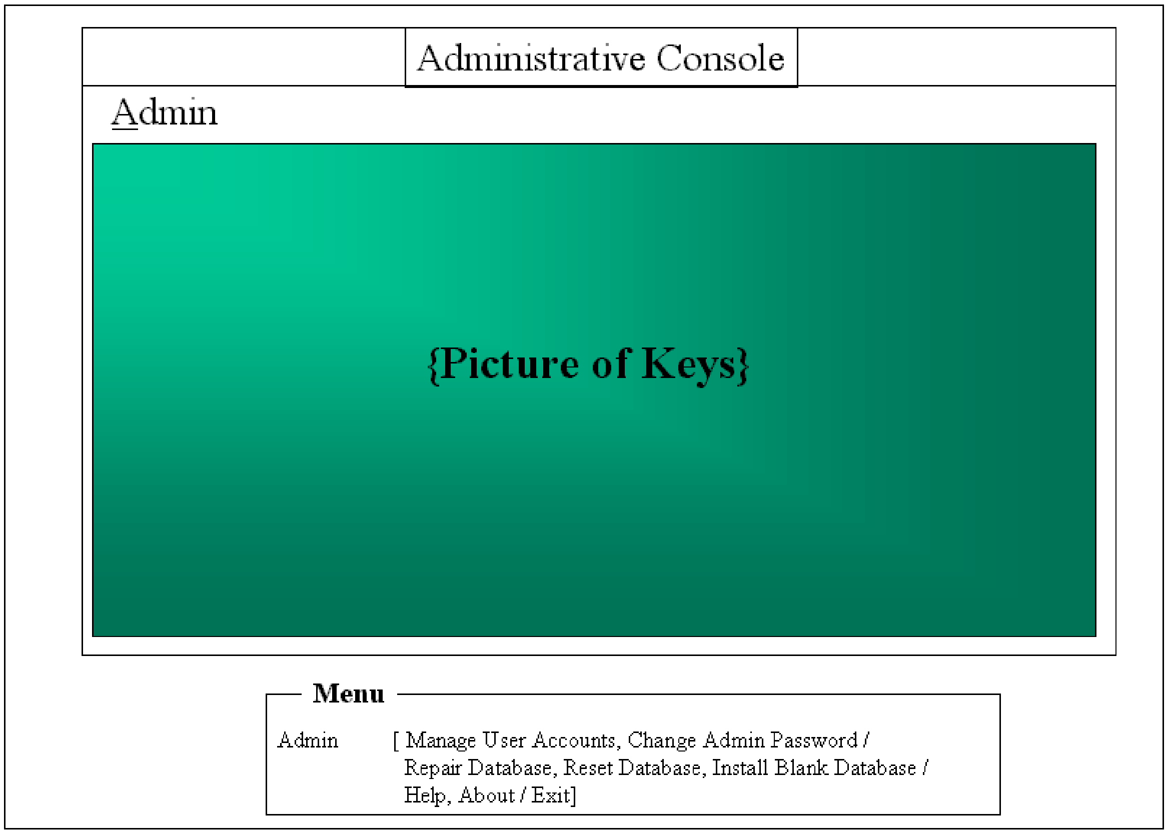
C0	Level 0

Descendants

C3.1	Database Management
C3.2	User Accounts Management

Data Sources Accessed And Modified
None

User Interface


Administrative Console
Admin
{Picture of Keys}
Menu
Admin
[Manage User Accounts, Change Admin Password / Repair Database, Reset Database, Install Blank Database / Help, About / Exit]

Input Name	Input Type	Remarks
Manage User Accounts	Menu	• Load component C3.2
Change Admin Password	Menu	• Load component C4.4
Repair Database	Menu	• Load component C3.1.3
Reset Database Path	Menu	• Load C3.1.2
Install Blank Database	Menu	• Load C3.1.1
Help	Menu	• Load context sensitive help
About	Menu	• Display credits
Exit	Menu	• Unload component

C3.1 Database Management

Rationale
Abstract logical grouping

Parent

C3	Administrative Console

Descendants

C3.1.1	Install Blank Database
C3.1.2	Reset Database Path
C3.1.3	Repair/Compact Database

Data Sources Accessed And Modified
None

User Interface
None

C3.1.1 Install Blank Database

Rationale
Required for application deployment

Parent

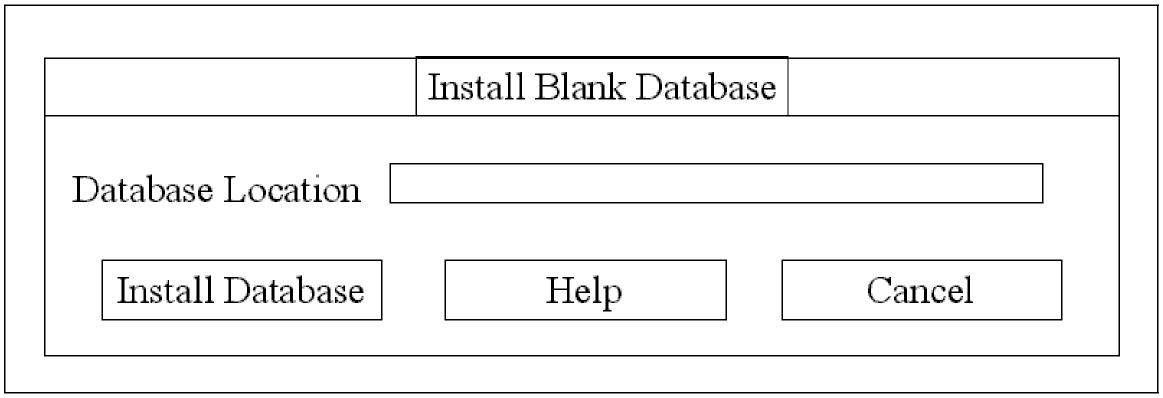
C3.1	Database Management

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Database	Yes	Created

User Interface



Input Name	Input Type	Remarks
Database Location	Text Box	• Locked • Clicking should bring about a system Save As dialogue which should be used for specifying the target database location
Install database	Button	• Create Database tables, indexes and queries as to the data modelling guidelines • Update path of database in registry • Unload component
Help	Button	• Load context sensitive help
Cancel	Button	• Unload component

C3.1.2 Reset Database Path

Rationale
To update system settings when database is moved.

Parent

C3.1	Database Management

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Registry	Yes	Location of database changed

User Interface
A system Save As dialogue. The user must supply an existing database. The supplied path will then be used to update the registry key.

C3.1.3 Repair/Compact Database

Rationale
Required to aid recovery. Also, jet databases tend fragment quite rapidly and as such require a regular data reorganisation.

Parent

C3.1	Database Management

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Database	Yes	Compacted

User Interface
None

Algorithms
Get a temporary file name
Get path of current db
Compact Database to temp file
Delete current file
Reassign the compacted db the current file name
Report success

C3.2 Manage Users

Rationale
Non-Functional Requirement 2

Parent

C3	Administrative Console

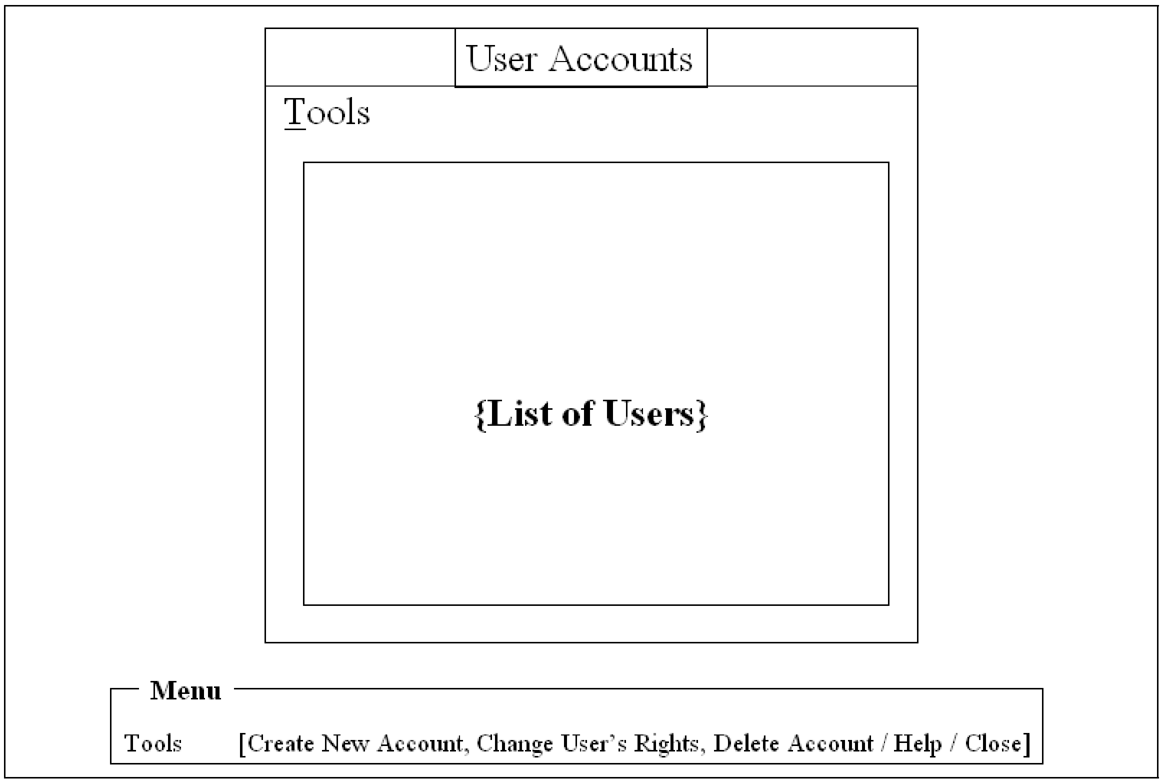
Descendants

C3.2.1	Create User Accounts
C3.2.2	Set Access Rights
C3.2.3	Delete User Account

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	No	Used to populate list of users

User Interface


User Accounts
Tools
{List of Users}
Menu
Tools [Create New Account, Change User's Rights, Delete Account / Help / Close]

Input Name	Input Type	Remarks
Create New Account	Menu	• Load C3.2.1
Change User's Rights	Menu	• Load C3.2.2 • There must be a user selected
Delete Account	Menu	• There must be a user selected • Load C3.2.3
Help	Menu	• Load context sensitive help.
Close	Menu	• Unload component

Algorithms
On Load
Populate list box with the user names of all the users recorded in the database

On Unloading A Component
Refresh the list of users.

C3.2.1 Create User Accounts

Rationale
Non-Functional Requirement 2

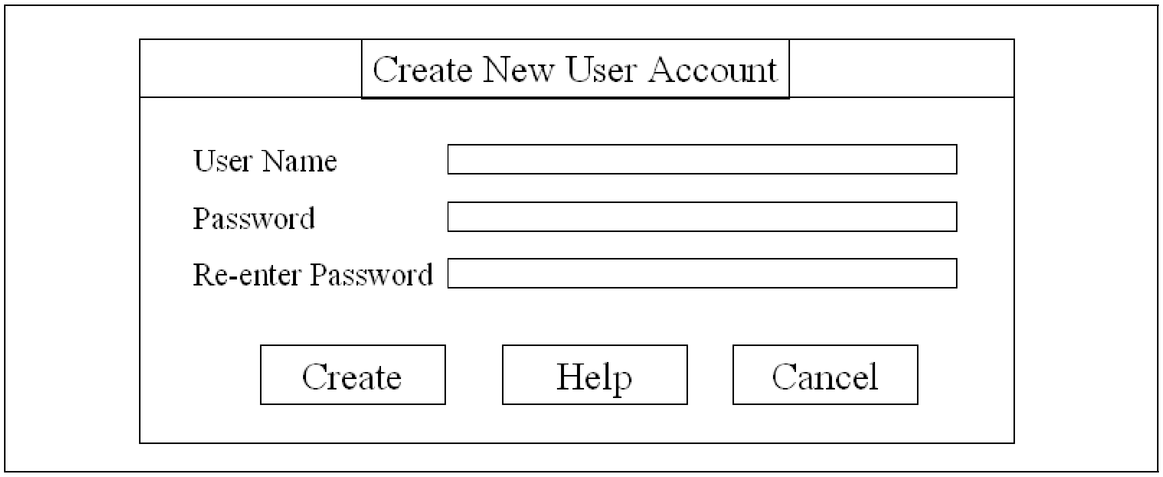
Parent
C3.2 Manage User Accounts

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	Yes	Adds new records

User Interface



Input Name	Input Type	Remarks
User Name	Text box	• Must not already be in use • Should not have leading or trailing spaces
Password	Text box	• Hidden Input • Up to 20 characters • Cannot be blank
Re-enter Password	Text Box	• Hidden input • Must match the input into Password
Create	Button	• Create a new record in the Users table with default access rights string of twenty 0s
Help	Button	• Load context sensitive help.
Cancel	Button	• Unload component

C3.2.2 Set User Rights

Rationale
Non-Functional Requirement 2

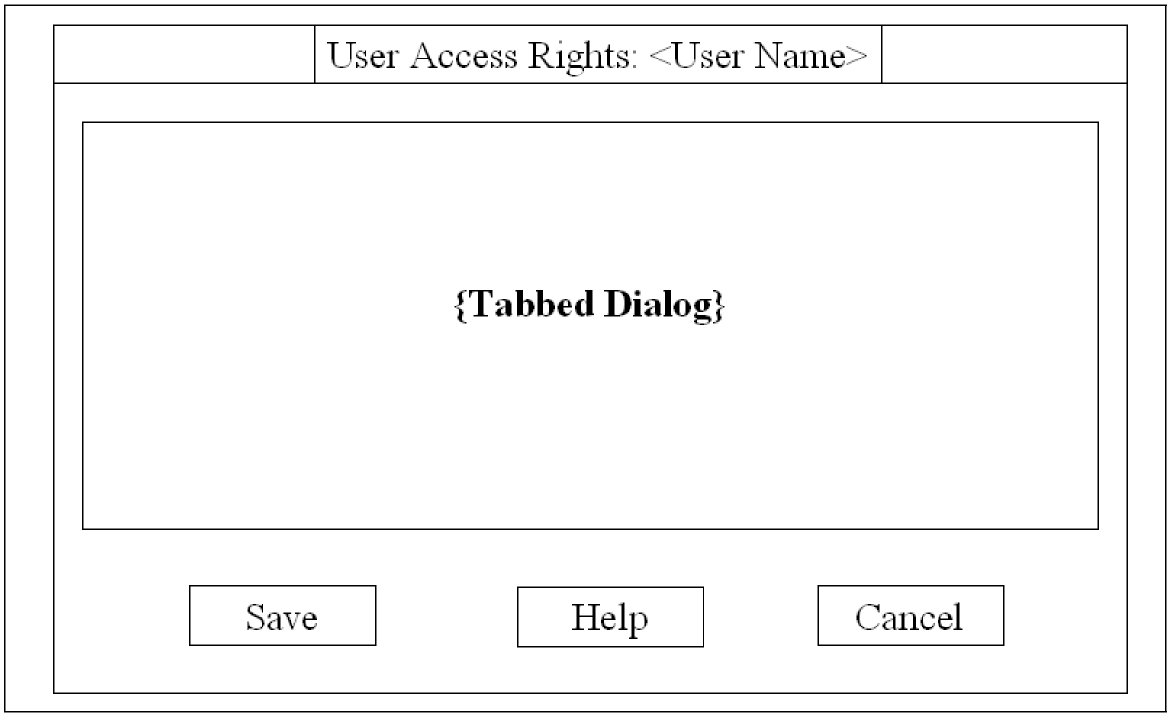
Parent
C3.2 Manage User Accounts

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	Yes	Edits the User Rights Field

User Interface



Input Name	Input Type	Remarks
Save	Button	• Save the currently displayed set of rights • Unload form
Help	Button	• Load context sensitive help.
Cancel	Button	• Unload component without saving settings

Remarks
The tabbed dialogue will be split into seven groups as detailed below for each property, a checked box will indicate that the user will be allowed to carry out the corresponding action or request the corresponding service.

Group Name	Properties
Classes	Create New Classes Change Class Details Delete classes Batch update classes
Students	Register students Change students' details Deregister students Reregister students
Levy Fees	Register new fees Deregister fees Create and modify templates Delete fees templates
Levy Accounts	Debit Accounts Credit Accounts Produce individual statements of account Batch produce statements of accounts
Deposit Accounts	Debit Accounts Credit Accounts Produce individual statements of account Batch produce statements of accounts
Balance Analysis	Fee Class Balance Analysis Analyse Deposit accounts' balances Analyse Levy accounts' balances
Utilities	Run Mail Merge Run Advanced Search

C3.2.3 Delete User Account

Rationale
Non-Functional requirement 2

Parent
C3.2 Manage User Accounts

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	Yes	Deletes records

User Interface
Confirmation of deletion dialogue

Algorithms
Locate User name
Delete Record

C4 System Services

Rationale
Abstract logical grouping of System Services components

Parent

C0	Fees Processing System

Descendants

C4.1	Start up
C4.2	Access Control
C4.3	Export
C4.3	Change Password

Data Sources Accessed And Modified
None

User Interface
None

C4.1 System Start Up

Rationale
Provides a generic system start up sequence for all the components

Parent

C4	System Services

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Database Object	No	Opened for access by all other components and subsystems. Table recordset objects instantiated.

User Interface
Splash screen specifying system and component names.

Algorithm

```
If another copy of the loading subsystem is already running
        Inform user and abort subsystem start up.
Else
        Get database path from registry
        Open the database
        Instantiate all the tables as recordsets
        Load component C4.2.1 to handle user log in
End if
```

Remarks
The following is a known dependency

C4.2	Access Control	Called to handle user authentication

C4.2 Access Control

Rationale
Abstract logical grouping of components used in access control.

Parent

C4	System Services

Descendants

C4.2.1	Log On
C4.2.2	Encryption
C4.2.3	Rights Enforcement

Data Sources Accessed And Modified
None

User Interface
None

C4.2.1 System Log On

Rationale
Non-Functional Requirement 2

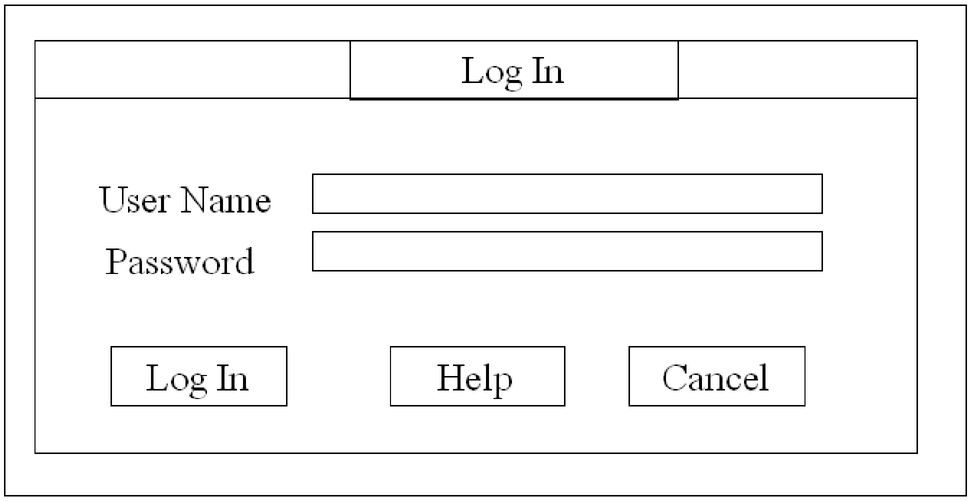
Parent
C4.2 Access Control

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	No	Queried for usernames, passwords and access rights

User Interface



Input Name	Input Type	Remarks
User Name	Text box	•
Password	Text Box	• Hidden input
Log In	Button	• Attempt to Log in user
Help	Button	• Load context sensitive help
Cancel	Button	• Halt system

Algorithms

Log In User

```
Ensure that both the username and password have been entered
If the username is Admin then
    Get admin password from registry
    If the entered password matches that of Admin then
        Enable access to all components
        Proceed to log in user
    Else
        Display log in error
        Redisplay log in form with the password text box cleared
    End if
Else
    Look up name in the users table
    If the name does not exist
        Inform user
        Refuse log in
    Else
        Get password for the user
        If password matches that entered then
            Retrieve user’s access rights
            Log In user
        Else
            Inform user
            Redisplay log in form with password text box cleared
        End if
    End if
End if
```

C4.2.2 Encryption

Rationale
Passwords are stored within a table and in the registry. Simply querying the database can therefore retrieve them. Encryption will provide a line of defence.

Parent

C4.2	Access Control

Descendants
None

Data Sources Accessed And Modified
None

Programming Interface
Method Encrypt: Accepts plain text as an argument and returns cipher-text
Method Decrypt: Accepts cipher-text as an argument and returns plain text

Encryption algorithm
© Librarius 2000 Plus, Motsi Tinovimba G., 1999 – 2000
Freely available for download at www.geocities.com/tgmotsi

C4.2.3 Rights Enforcement

Rationale
Non-Functional Requirement 2

Parent

C4.2	Access Control

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Users Table	No	Read

User Interface
None

Programming Interface
The components should read in the access rights and provide access to these through an object with a property for each right. The property will have a value of false if the user is not allowed to access the service referenced by the property, otherwise it will be true. There should also be a property storing the User Name of the currently logged on user.

C4.3 Export

Rationale
Required to facilitate the attainment of requirement 9

Parent

C4	System Services

Descendants
None

Data Sources Accessed And Modified
None

User Interface
None

Programming Interface
The components should provide two methods, one for exporting to Excel 95 and the other to Generic Comma Delaminated Text format. The methods should accept as a compulsory argument, a grid containing the body of the tabulated data to be exported. There should also be a second optional argument specifying a grid with tabulated text to be included as a header to the report. During export, a progress display should be provided.

C4.4 Change Password

Rationale
Required to facilitate the editing of user password as required by Non-Functional Requirement 2.

Parent

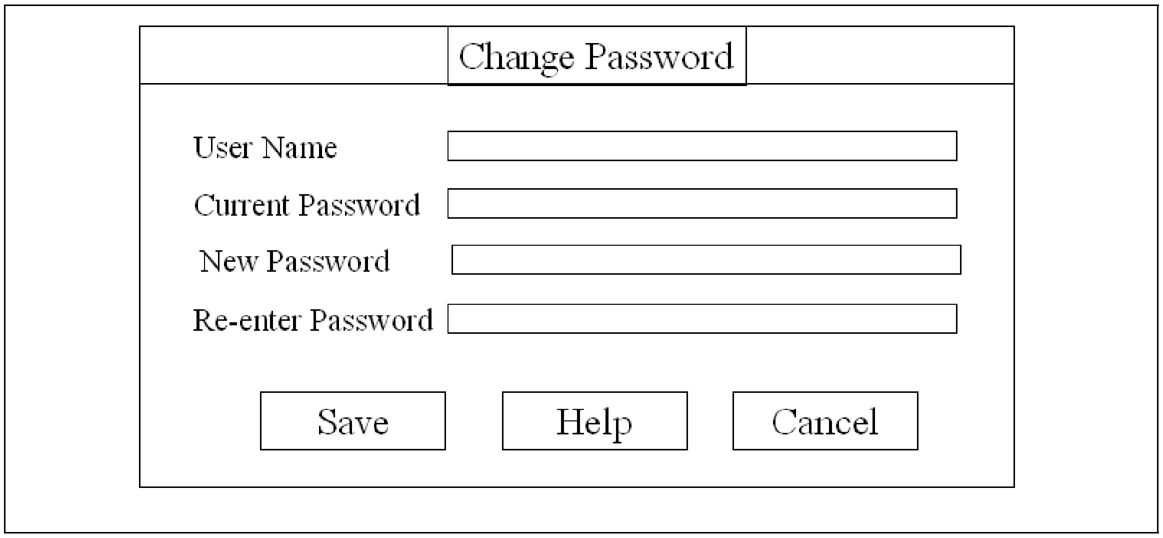
C4	System Services

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Remarks
Users Table	Yes	Password field changed

User Interface



Input Name	Input Type	Remarks
User Name	Text Box	• Should be automatically entered by the system by referencing the Current User object • Read only
Current Password	Text box	• Hidden input
New password	Text box	• Hidden Input
Re-enter Password	Text box	• Hidden Input
Save	Text Box	• Save changes to the password
Help	Text box	• Load context sensitive help
Cancel	Text box	• Unload component without committing the changes

Algorithm

```
If any of the fields has not been entered then
    Inform user and give option to have missing input entered
Else
    Get the password of the current user
    If **Entered Current Password** does not match **Stored Current Password** then
        Inform user and refuse change
        Give option to re-enter current password
    Else
        If the entered new passwords do not match then
            Inform user and refuse change
            Clear both new password text boxes
            Give option to re-enter the new password
        Else
            Locate record of current user
            Modify password field to that of the new password
        End if
    End if
End if
```

C5 Advanced Search

Rationale
Requirement 10

Parent

C0	Levy Processing System

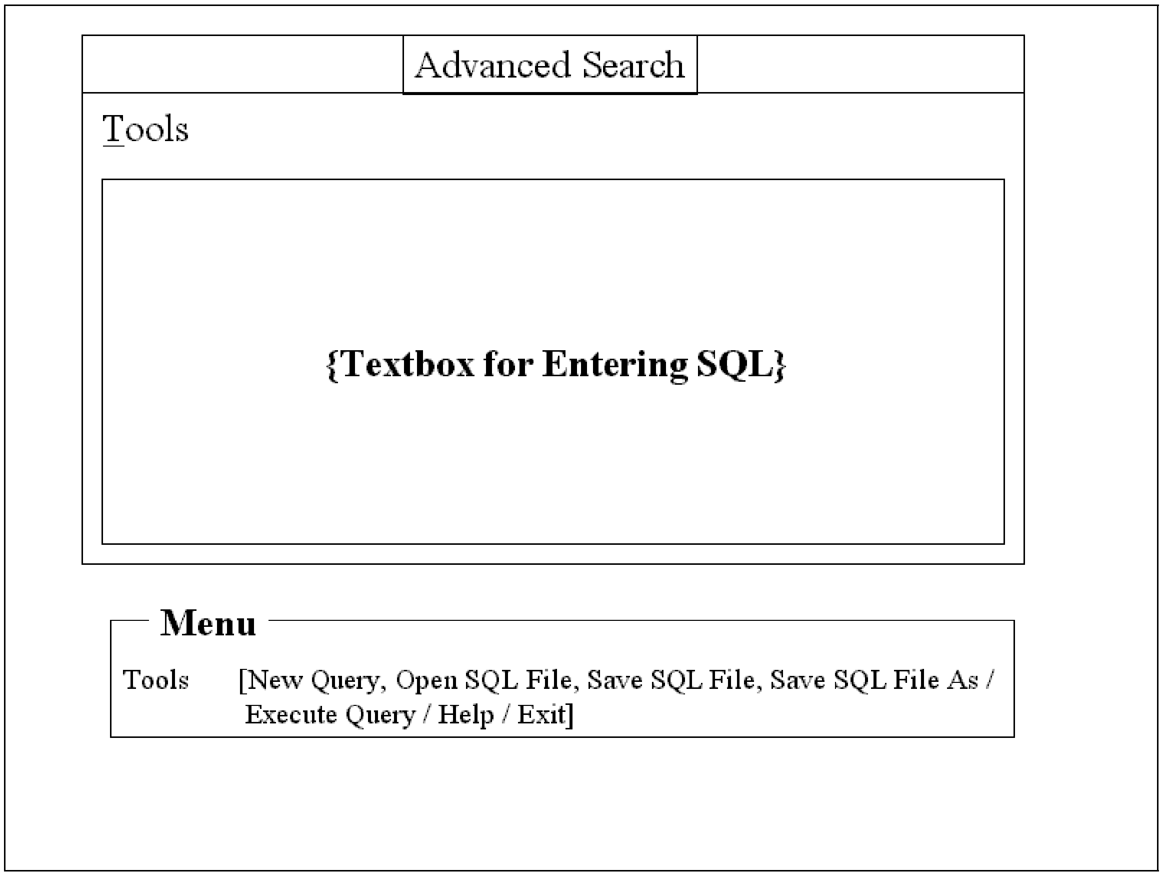
Descendants

C5.1	Advanced Search Results

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Database	No	Queried

User Interface


Advanced Search
Tools
{Textbox for Entering SQL}
Menu
Tools
[New Query, Open SQL File, Save SQL File, Save SQL File As / Execute Query / Help / Exit]

Input Name	Input Type	Remarks
SQL text box	Text box	• Should have both scroll bars
New Query	Menu	• Clear the text box
Open SQL file	Menu	• Use the system Open File dialogue to open a text file
Save SQL File	Menu	• If the loaded file already has a filename associated with it then save the loaded text under that file • Otherwise use the system Save As dialogue to save the file
Save SQL File As	Menu	• Use the system Save As dialogue to save the file
Execute Query	Menu	• Assign the text to a query object's SQL property. if the query contains any errors, an exception will be thrown. • Ensure that the query does not have the ability to modify data before execution • Load component C5.1 to display the results
Print	Menu	• Use the system Print dialogue to print the SQL text.
Help	Menu	• Load context sensitive help
Exit	Menu	• Unload the component.

C5.1 Advanced Search Results

Rationale
Requirement 10

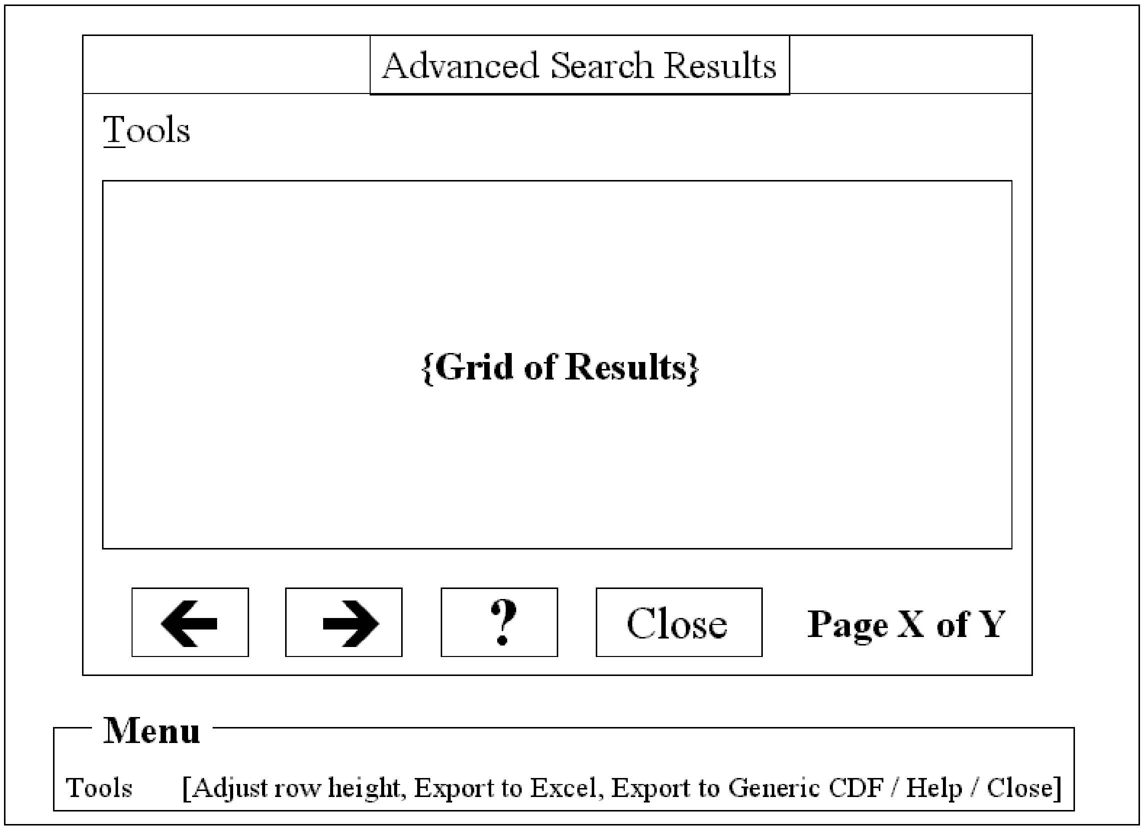
Parent
C5 Advanced Search

Descendants
None

Data Sources Accessed And Modified

Data Source	Modified	Nature of Access
Database	No	Queried

User Interface


Advanced Search Results
Tools
{Grid of Results}
?
Close
Page X of Y
Menu
Tools [Adjust row height, Export to Excel, Export to Generic CDF / Help / Close]

Display the Results
Execute the query and assign the results to a recordset
If the recordset is empty then abort
Calculate the number of pages returned by the result
Enter the column headers by iterating through the field object of the recordset
Display the first page

Adjust Row Height

```
Use input box control to get the preferred number of lines of text to display
Convert text height to scale mode units
For each row in the grid excluding the header row
        Set the new height to the calculated height
Next
```

Export To Excel

```
Get the filename of the target export
If the file exists, delete it after confirming with the user
Create an Excel Workbook with a Single Worksheet
Export the header row
Load each page from page 1 and export its contents excluding the Header page
Save and release the workbook
Redisplay the page that was being viewed just before the exporting began.
```

Export to Generic CDT

```
Get the filename of the target export
If the file exists, delete it after confirming with the user
Create a text file to export the data to
Export the header row
Load each page from page 1 and export its contents excluding the Header page
Save the text file
```

```
Redisplay the page that was being viewed just before the exporting began.
```

Validation And Evaluation

System Requirements

The meeting of the system requirements was verified by walkthroughs at various points in time from the date the requirements were specified to the end of the implementation. In most cases, the requirements were found to have been met. However, during this process, some inconsistencies were found and corrected. The following highlights the major revisions to the Functional Requirements.

1. Requirements 1a was found to be out of date after implementation since it did not include a reference to the Guardian's Telephone number. This was traced to the requirement not having been updated when the Data Modelling was revised. The inconsistency was corrected by adding the missing reference.
2. Requirement 1c, which included the wording "Facilities should exist for finding all the student details given some of the attributes listed in 1a", was found to be unverifiable during the design of the search modules. This was corrected by changing the wording to include an explicit listing of the searching fields.
3. Requirement 2c, which included the wording "The operator should be able to charge all the students in a given class by entering the fees in a template for that class" was found to be ambiguous during the design of the component for debiting Levy Accounts. The wording was subsequently revised to its current state.
4. The sorting order specified in Requirement 2cii was found to have been overlooked. The code was subsequently revised to include an implementation of the sorting order.
5. Requirement 8c was found to be unverifiable and was subsequently revised to include an explicit listing of the text editing and formatting functions that were to be included.
6. Requirement 9.a.i was found to be unverifiable and had its wording changed from "Standard Comma Delaminated Format" to "Generic comma delaminated format as implemented in Microsoft® CSV"

Test Cases

The following test cases have been proposed to test the functional validity of the implementation of the system. They are grouped as to the implementation units.

Installation Program

The installation program should be tested on the following platforms

1. Microsoft® Windows 95
2. Microsoft® Windows 98
3. Microsoft® Windows 2000
4. Microsoft® Windows XP

Care should be taken before testing to ensure that the target system does not have either Microsoft Visual Basic or Microsoft Office installed. This is to ensure that the installation program is successfully installing its required runtime libraries.

Administrative Console

The following Aspects of the Administrative Console should be tested

1. Restriction of access to the “admin” user.
2. Implementation of database management
    a. Ability to install sample database
    b. Ability to install blank database
    c. Ability to repair a corrupt database.
    d. Ability to compact database.
3. Implementation of Access control
    a. Creation of new users
    b. Changing of user access rights
    c. Deletion of user accounts

System Services

1. Encryption
    a. Ability to encrypt
    b. Ability to decrypt
2. Ability to enforce the access rights. For this test to be effective, a real effort must be made towards bypassing the security system.
3. Changing of user passwords
4. System start-up.
5. Context sensitive help loading.
6. Exportation to Excel and CSV.

Advanced Search

1. Opening, saving and printing of source SQL.
2. Execution of SQL. Executing a query in the Advanced Search Utility and then comparing the results with those obtained when the query is run in Microsoft Access can be used to test this.
3. Presentation of results with particular emphasis on adjusting row heights and exporting data.

Mail Merge

1. Opening, Saving and Printing of RTF documents.
2. Ability to “Undo” actions.
3. Clipboard functions
4. Converting case.
5. Inserting dates, rtf files and pictures.
6. Search and replace.
7. Inserting merge fields
8. Generating previews and actual letters.
9. Starting of new sessions.

Main Subsystem

1. Classes

   a. Ability to register new classes with particular emphasis on observance of the enforcement of primary key integrity.
   b. Editing of classes and ensuring that the primary key cannot be changed
   c. Ability to deregister classes. It should be ensured that the system can detect classes that are still in use and deny requests to deregister them
   d. Ability to browse through the records
   e. Batch Updating of classes

2. Registered Students
   a. Ability to register new students with particular emphasis on the generation of unique Student IDs and enforcement of the Student-Class relationship.
   b. Editing of student records and ensuring that the primary key cannot be changed.
   c. Ability to deregister students. It should be ensured that the system is capable of detecting students with non-zero balances in the accounts and denying requests to deregister them. The system should also be observed to be making accurate transfers of the data from the Registered to the Deregistered table. Deletion of transaction records should also be verified.
   d. Ability to browse through the records with particular emphasis on ability to jump to a student record and the state of the table after an unsuccessful jump.

3. Deregistered Students
   a. Ability to browse through the record of deregistered students.
   b. Ability to reregister deregistered students.

4. Search facility
   a. Ability to run searches as specified in the design.
   b. Ability to handle external service requests as generated by the components for deregistering classes and charging student accounts.
   c. Ability to present results with particular emphasis on display the printing and exporting of results.

5. Registered fees
   a. Registration of new fees with particular emphasis on the generation of a unique primary key for each fee and the enforcement of the uniqueness of each fee.
   b. Deregistration of fees
   c. Ability to detect usage of fee within templates and levy accounts and deny request to deregister fees in use.

6. Fees Templates
   a. Ability to create new blank templates with unique names
   b. Ability to delete templates
   c. Insertion of fees into templates with focus on the need for the fee to be registered.
   d. Removal of fees from templates

7. Debiting Levy Accounts
    a. Ability to specify debit targets and to have these adhered to.
    b. Making of the request to display the list of students in the target class.
    c. Ability to retrieve and load template data.
    d. Ability to add and remove individual fees from the list specified.

8. Crediting Levy Accounts

9. Generation of individual levy statements of accounts and to print & export the generated statements

10. Ability to batch produce levy statements of accounts

11. Deposit accounts
    a. Debiting
    b. Crediting
    c. Statement generation
    d. Batch statements of accounts generation

12. Balance analysis.

Testing Results

The above aspects of the system were tested leading to the discovery of errors within the implementation. Due to time pressures, the developer was not able to systematically collect the test results for presentation in this manual. Although most of the errors discovered were corrected, the following system inadequacies have been confirmed and remain largely unresolved although workarounds maybe available.

1. The installation program will fail to install bootstrap files on Windows 2000 machines. The program may also fail on other versions of Windows. For Windows 2000, the problem results from the way Windows 2000 protects its files during booting. To work around this problem on Windows 2000, the user must install Windows 2000 Service Pack 3 before running the installation program. The solution for the other versions can only be determined after analysing the registry of the target machine focusing on the flags for updating files on booting. This issue has been confirmed by Microsoft to be a known problem affecting applications designed in Visual Basic 6.0 and is documented at
    a. http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/Q191/0/96.ASP&NoWebContent=1
    b. http://support.microsoft.com/default.aspx?scid=kb;EN-US;279764

2. The extent to which the log on is secure has not been verified to the required level of security. Several attempts to bypass the form succeeded and although the underlying cases for all of these were isolated and rectified, it is not possible to state that logins are secure beyond reasonable doubt

3. The help system has been verified to be loading the required topics. However, the context sensitivity of the help file itself remains questionable due to the system used to develop it. The file was created by partitioning the User Manual which when considered in retrospect, was a poor design decision.

4. The Undo system in the mail merge utility logs all the changes made as single snapshots resulting in some changes that should be considered atomically being broken down. Most of the problems associated with this have been identified and rectified by making changes in the working area and then transferring the finished version of the text back to the parent. The developer however still has suspicions that there are other scenarios that may result in the same problems that have not been identified.

5. The mail merge clipboard function remains unreliable when it comes to images. This problem can be avoided by using the keyboard shortcuts to the inbuilt clipboard functions of the of the RTF control thus bypassing the system's custom functions.

6. Deregistration of classes and students leads to the forms displaying the first record. This is as to design but in retrospect, this was a poor design decision. The system should instead display the record occurring immediately after the deleted record.

7. Analysis functions return incorrect results when the accounts tables are empty. The cause of this error has not been identified. This problem is however not likely to be materially problematic as it is expected that the analysis will have no practical value until some records have been added.

8. The advanced search utility's paging mechanism results in the utility being unable to use the generic exportation functions written for the system. To work around this, a custom export routine had to be written for the utility. The developer acknowledges that this is a violation of the design principles.

Plans For Future Development

As stated in the preface, the scope for this project has changed dramatically from the time it was first conceptualised. Over time, the project has become more of a prototype for a larger Information Management system. As such, future development will focus on adding general IS features to the system. The question of what to add next and when remain largely unanswered. Some of the proposals are stated below.

1. The students' records management subsystem should be unbundled from the levy system and placed in a new standalone Students Records client.
2. The database should be moved to a DBMS that caters for efficient multi-user support whilst leaving the interface for the levy system largely unchanged.
3. A courses management client should be added with support for the following
    a. Association of students with marks
    b. Exams timetable analysis for clashes
    c. End of terms reports generation.
4. A form teacher client should be added and be online based. The primary purpose of this will be to enable a form teacher to keep track of his/her students and to add comments before reports are generated.
5. An online subject teacher client enabling a course teacher to log in and record examination marks should be considered.

The above proposals will have to be developed in consultation with all the stakeholders and prioritised as to their needs. It is therefore not possible to give a schedule for any of the issues stated above.

Appendix A

Department of Computing Science
University of Zimbabwe
P O Box MP167
Mount Pleasant
Harare

27 February 2003

The Permanent Secretary
Ministry of Education, Sports and Culture
Ambassador House
Harare

Dear Sir

RE: RESEARCH CLEARANCE

I would like to request clearance for the carrying out of a research into Fees Processing Automation at Mount Pleasant High School. The research, should you approve it, will focus on the management of students' accounts with respect to fees charged and paid.

Please find attached a copy of my project proposal and a letter of authorization from the University of Zimbabwe.

I sincerely hope that you will be able to assist me.

Yours faithfully

____

Motsi Tinovimba G

Project Proposal

Title
Fees Processing Automation

Terms of Reference
Authorised by the Department of Computing Science at the University of Zimbabwe as part of the programme of study for the Bachelor of Business Studies and Computing Science (Honours) degree programme. The course code is CT260.

Project Supervisor
Ms L. Bonda
Department of Computing Science
University of Zimbabwe
P. O. Box MP167
Mount Pleasant
Harare

Objective of Research
To establish an understanding of the manual fees processing system in use at Mt Pleasant High School deep enough to facilitate the design and implementation of a fully computerised alternative.

Methodology
The research will be by means of interviews with key personal involved in the processing of fees at the institution. The following is the proposed focus group

- The School Head
- The Deputy to the School Head
- The Bursar
- Assistants to the Bursar
- Secretary to the School Head
- Secretary to the School Development Association (SDA)

The interviews would be held at the earliest possible convenience of the personnel involved.

Interview Focus
The interviews will focus on the following key areas

- Types of fees charged.
- Procedures and documentation used in authorising the fees to be charged against any given student.

- Storage and uses of the information
- Identification of any problems in the system with particular focus being given to bottlenecks.
- Formulation of requirements to serve as a basis for solving the above stated problems.

Publication of Findings
The research findings will form part of the report to be submitted as to the academic regulations of the University of Zimbabwe. It will therefore be left within the public domain of the Department of Computing Science. A copy will also be published at the author's website at www.geocities.com/tgmotsi. The website is within the international public domain and already contains other projects carried out by the researcher.

Researcher Contact Details

E-mail	- tgmotsi@yahoo.com
Website	- www.geocities.com/tgmotsi
Telephone	- +263 11 754102

Compiled by

____

Motsi Tinovimba G

Interview Questions

5. **The Head / Deputy Head**
    a. **Storage of Information**
        i. What information do you store within your immediate office with regards to fees records?
        ii. What forms, if any do you use in this storage?
        iii. On average, for how long is this information stored?
        iv. On expiration of the period of storage, is the information destroyed or archived. If it is archived, is it revised in any way before this is done?

    b. **Uses of information**
        i. To what uses do you put the above-specified information?
        ii. What problems, if any, have you experienced in accessing the above specified information?
        iii. What opportunities for further utilization of the information would you wish to have explored?

    c. **Computer System**
        i. Do you have a computer system within your immediate office?
        ii. If you do have a computer system, what are its technical specifications with respect to :
            - Make
            - RAM
            - Network connectivity
            - Storage
        iii. How would you appraise your level of computer literacy?

6. **The Bursar / Assistants to the Bursar / Levy Office**
    a. **Types of fees charged**
        i. What fees do you charge?
        ii. To whom do these fees apply?
        iii. Who is responsible for which fees?
        iv. Who gives the directive to charge any given student a given fee?
        v. What documentation if any is used in (2.a.iii) above?
        vi. What documentation used in communicating the fees charged against any given student to that student?

b. Storage of information

i. What specific transaction records do you keep?
ii. What documentation is used in this storage?
iii. For how long are these records kept?
iv. What happens to the records when their period of use has expired?

c. Uses of information

i. What specific uses do you put the above specified information?
ii. Who are the consumers of this information and what specific aspects of it do they require?
iii. What problems, if any, have you experienced in storing or accessing this information?
iv. What opportunities for further utilization of the information would you want explored?

d. Computer System

i. Do you have a computer system within your immediate office?
ii. If you do have a computer system, what are its technical specifications with respect to:
    - Make
    - RAM
    - Network connectivity
    - Storage
iii. How would you appraise your level of computer literacy?

Compiled By

____

Motsi Tinovimba G.

Department of Computing Science
University of Zimbabwe
P O Box MP167
Mount Pleasant
Harare

27 February 2003

The Regional Director
Harare Region
Ministry of Education, Sports and Culture
Chester House
Harare

Dear Sir

RE: RESEARCH CLEARANCE

I am a second year Bachelor of Business Studies and Computing Science (Honours) student at the University of Zimbabwe. As part of my approved program of study, I have been tasked with the carrying out of a research into the computerisation of the Fees Processing system at Mount Pleasant High School. I am therefore hereby applying for clearance to carry out the research.

The research, should you approve it, would focus on the management of student accounts with particular focus being given to

- Types of fees charged.
- Procedures and documentation used in authorising the fees to be charged against any given student.
- Storage and uses of the information
- Identification of any problems in the system with particular focus being given to bottlenecks.
- Formulation of requirements to serve as a basis for solving the above stated problems.

It is anticipated that the information would be solicited through interviews with the administrative staff in the office of the School Head involved in the above-mentioned process at their earliest possible convenience.

I sincerely hope that you will be able to assist me.

Yours faithfully

____
Motsi Tinovimba G.